Deformable Models

Jasjit S. Suri
Aly A. Farag

Deformable Models

Biomedical and Clinical Applications

Springer

Jasjit S. Suri
Eigen LLC
Grass Valley, CA 95945
USA
jsuri@comcast.net

Aly A. Farag
Professor of Electrical
and Computer Engineering
Computer Vision and Image
Processing Laboratory
University of Louisville
Louisville, KY 40292
USA
aly.farag@louisville.edu

Series Editor

Evangelia Micheli-Tzanakou
Professor and Chair
Department of Biomedical Engineering
Rutgers University
Piscataway, NJ 08854-8014
Etzanako@rci.rutgers.edu

Library of Congress Control Number: 2007925874

ISBN-13: 978-0-387-31201-9 e-ISBN-13: 978-0-387-68413-0

Printed on acid-free paper.

9 8 7 6 5 4 3 2 1

springer.com

PREFACE

Variational or Partial-Differential Equations (PDE) approaches have proven to be powerful for biomedical image analysis. The basic ingredients of image analysis are basically comprised by the trilogy of segmentation, registration, and visualization. Variational and PDE approaches enable soft tissue modeling and detailed structural analysis of complicated topologies, and create the capability of alignment of tissues for subsequent statistical analysis. This two-volume set, *Deformable Models*: *Biomedical and Clinical Applications*, and *Theory and Biomaterial Applications* provides a wide cross-section of the methods and algorithms of variational and PDE methods in biomedical image analysis. The chapters are written by well-known researchers in this field, and the presentation style goes beyond an intricate abstraction of the theory into real application of the methods and description of the algorithms that were implemented. As such these chapters will serve the main goal of the editors of these two volumes in bringing down to earth the latest in variational and PDE methods in modeling of soft tissues. Researchers at various levels will find these chapters useful to understand the theory, algorithms, and implementation of many of these approaches.

The two volumes not only introduce application research but also novel theoretical ideas related to the problem under investigation. A list of algorithms is found in most of the chapters to provide an opportunity for those who want to implement the algorithms and repeat the results. Nearly all the chapters have a theoretical and algorithmic aspect of variational and PDE-based methods. A brief summary of the chapters in Volume 1 follows.

Chapter 1 deals with simulation of bacterial biofilms and ubiquitous life forms on the planet. More than 90 aggregates of cells attached to both biotic and abiotic surfaces. Level sets are combined with other numerical methods to model biofilms to explore the variety of their behavior.

Chapter 2 examines the distance transform and other distance measures and provides an approach to evaluate them. Interpolation and skeletonization are discussed as applications for the distance transform in image analysis and processing.

Chapter 3 deals with structural inversion for modeling structural information in medical imaging. It provides a brief introduction to some techniques that have been recently developed for solving structural inverse problems using level sets.

Chapter 4 deals with shape- and texture-based deformable models as applied to facial image analysis. The chapter examines various characteristics of active shape, appearance, and morphable models.

Chapter 5 describes a method for identification of the breast boundary in mammograms and its use in CAD systems for the breast and other applications.

Chapter 6 describes the use of statistical deformable models for cardiac segmentation and functional analysis in Gated Single Positron Emission Computer Tomography (SPECT) perfusion studies.

Chapter 7 presents an implicit formulation for dual snakes based on the level set approach. The key idea is to view the inner/outer contours as a level set of a suitable embedding function. Details of the approach are provided with applications to segmentation of cell images.

Chapter 8 describes a generalized approach for monotonically tracking advancing fronts using a multistencil fast marching (MSFM) method, which computes a solution at each grid point by solving the eikonal equation along several stencils and then picks the solution that satisfies the fast marching causality relationship.

Chapter 9 examines the use of deformable models for image segmentation and introduces an approach to reduce the dependency on initialization that utilizes object and background differentiation through watershed theories.

Chapter 10 describes the use of deformable models for detection of renal rejection as detected by Dynamic Contrast Enhanced Magnetic Resonance Images (DCE-MRI). The approach involves segmentation of the kidney from surrounding tissues and alignment of the segmented cross-sections to remove the motion artifices. The renogram describing the kidney perfusion is constructed from the graylevel distribution of the aligned cross-sections.

Chapter 11 deals with a class of physically and statistically based deformable models and their use in medical image analysis. These models are used for segmentation and shape modeling.

Chapter 12 reviews deformable organisms, a decision-making framework for medical image analysis that complements bottom–up, data-driven deformable models with top–down, knowledge-driven mode-fitting strategies in a layered fashion inspired by artificial life modeling concepts.

Chapter 13 provides a detailed description and analysis for use of PDE methods for path planning with application to virtual colonoscopy. The method works in two passes: the first identifies the important topological nodes, while the second

pass computes the flight path of organs by tracking them starting from each identified topological node.

Chapter 14 describes an approach for object tracking in a sequence of ultrasound images using the Hausdorff distance and entropy in level sets. The approach tracks the region of interest (TOI) using information in previous and current slices and accomplishes segmentation with Tsallis entropy. The Hausdorff distance is used to match candidate regions against the ROI in the previous image. This information is then used in a level set formulation to obtain the final output.

Chapter 15 describes a deformable model-based approach for image registration. A nonuniform interpolation functions is used in estimating the joint histogram between the target and reference scans. A segmentation-guided nonrigid registration framework is described.

The authors of these chapters deserve a lot of credit and have the responsibility for preparing first class manuscripts that will stand the test of time and will guarantee the long-term value of these two volumes. Several people at Springer deserve special credit for making every effort to carry out this project in such a beautiful and professional fashion. In particular, Aaron Johnson, Senior Editor, Beverly Rivero, Editorial Assistant, Tim Oliver, Project Manager, and Amy Hendrickson, LaTeX Consultant, have made every effort to make this project smooth to its superb completion.

Finally, Jasjit Suri and Aly Farag acknowledge the support of their families and express their gratitude to their collaborators and graduate students.

Jasjit Suri and Aly Farag
January 2007

CONTENTS

CONTRIBUTORS

S. Aguade
Vall d' Hebron University Hospital
Barcelona, Spain

Antonio L. Apolinário Jr.
National Laboratory for Scientific Computing
Petropolis, Brazil

Rejane A. Borges
Nucleus of Science and Technology
University of Mogi das Cruzes
São Paulo, Brazil

J. Castell
Vall d' Hebron University Hospital
Barcelona, Spain

Ruey-Feng Chang
Department of Computer Science and Information Engineering
National Chung Cheng University
Chiayi, Taiwan

David L. Chopp
Department of Engineering Sciences and Applied Mathematics
Northwestern University
Evanston, Illinois, USA

J.E. Leo Desautels
Department of Electrical and Computer Engineering
University of Calgary
Calgary, Alberta, Canada

Oliver Dorn
Departamento de Matemáticas
Universidad Carlos III de Madrid
Madrid, Spain

Ayman El-Baz
Computer Vision and Image Processing Laboratory
University of Louisville
Louisville, Kentucky, USA

Tarek A. Eldiasty
Urology and Nephrology Department
University of Mansoura
Mansoura, Egypt

Mohamed E.A. El-Ghar
Urology and Nephrology Department
University of Mansoura
Mansoura, Egypt

ROBERT FALK
Department of Medical Imaging
Jewish Hospital
Louisville, Kentucky, USA

ALY A. FARAG
Computer Vision and Image Processing Laboratory
University of Louisville
Louisville, Kentucky, USA

RICARDO J. FERRARI
Department of Electrical and Computer Engineering
University of Calgary
Calgary, Alberta, Canada

A.F. FRANGI
Computational Imaging Laboratory
Universitat Pompeu Fabra
Barcelona, Spain

ANNIE F. FRÈRE
Department of Electrical and Computer Engineering
University of Calgary
Calgary, Alberta, Canada

MOHAMED A. GHONEIM
Urology and Nephrology Department
University of Mansoura
Mansoura, Egypt

GILSON A. GIRALDI
National Laboratory for Scientific Computing
Petropolis, Brazil

GEORGE J. GREVERA
Mathematics and Computer Science Department
St. Joseph's University
Philadelphia, Pennsylvania, USA

GHASSAN HAMARNEH
School of Computing Science
Simon Fraser University
Burnaby, British Columbia, Canada

M. SABRY HASSOUNA
Computer Vision and Image Processing Laboratory
University of Louisville
Louisville, Kentucky, USA

ZHEN LEI
National Laboratory of Pattern Recognition
Institute of Automation
Chinese Academy of Sciences
Beijing, China

RONGXIN LI
BioMedIA Lab
Autonomous Systems Laboratory
CSIRO ICT Centre
Marsfield, New South Wales 2122, Australia

STAN Z. LI
National Laboratory of Pattern Recognition
Institute of Automation
Chinese Academy of Sciences
Beijing, China

JUNDONG LIU
School of Electrical Engineering and Computer Science
Ohio University
Athens, Ohio, USA

CHRIS MCINTOSH
School of Computing Science
Simon Fraser University
Burnaby, British Columbia, Canada

S. ORDAS
Computational Imaging Laboratory
Universitat Pompeu Fabra
Barcelona, Spain

SEBASTIEN OURSELIN
BioMedIA Lab
Autonomous Systems Laboratory
CSIRO ICT Centre
Marsfield, New South Wales 2122,
Australia

RANGARAJ M. RANGAYYAN
Department of Electrical and Computer
Engineering
University of Calgary
Calgary, Alberta, Canada

PAULO S.S. RODRIGUES
National Laboratory for Scientific
Computing
Petropolis, Brazil

RODRIGO L.S. SILVA
National Laboratory for Scientific
Computing
Petropolis, Brazil

JASJIT S. SURI
Biomedical Research Institute
Idaho State University
Pocatello, Idaho, USA

C. TOBON-GOMEZ
Computational Imaging Laboratory
Universitat Pompeu Fabra
Barcelona, Spain

ZENGFU WANG
Department of Automation
Institute of Information Science and
Technology
University of Science and Technology of
China
Hefei, China

SENIHA E. YUKSEL
Computer Vision and Image Processing
Laboratory
University of Louisville
Louisville, Kentucky, USA

YING ZHENG
Department of Automation
Institute of Information Science and
Technology
University of Science and Technology of
China
Hefei, China

1

SIMULATING BACTERIAL BIOFILMS

David L. Chopp

Department of Engineering Sciences and Applied Mathematics
Northwestern University, Evanston, Illinois

Biofilms are the most ubiquitous form of life on the planet. More than 90% of bacteria live in biofilms, which are aggregates of cells attached to both biotic and abiotic surfaces [6, 13]. Biofilms are responsible for nitrogen loss from agricultural fertilizers, and they deplete oxygen in streams, cause disease in humans and plants, and foul pipes, heat exchangers, and ship hulls. Biofilms are responsible for a number of human diseases, including cystic fibrosis and Legionnaire's disease, and are a potential source of nosocomial infections. According to The Biofilm Institute, biofilms cost U.S. industry billions of dollars annually in equipment and product damage, energy losses, and human infections.

On the other hand, biofilms are also exploited for their good properties. Biofilms are employed for treating sewage, industrial waste streams, and contaminated groundwater. Biofilms can also be used to improve nutrient cycling through plant roots to improve agricultural productivity, and are used to produce a wide variety of biochemicals used in medicines, food additives, and cleaning products. Because of their immense impact on society, both positively and negatively, understanding biofilms is an important goal. In this chapter, we describe how the level set method is used to simulate biofilm growth and development.

1. INTRODUCTION

Biofilms are composed of a number of different components. They may include multiple species of bacteria—for example, in activated sludge reactors, nitrifying bacteria form dense subclusters within larger colonies of heterotrophic bacteria [39]. Nitrifying bacteria are essential for nitrogen cycling in nature and

Address correspondence to David L. Chopp, Engineering Sciences and Applied Mathematics Department, Northwestern University, Evanston, IL 60208-3125, USA. chopp@northwestern.edu.

are used to treat nitrogen-contaminated water and wastewater [54]. The nitrifying bacteria are autotrophs that consume inorganic carbon sources and produce microbial products that can be consumed by heterotrophic bacteria. Heterotrophic bacteria rely on organic carbon sources such as glucose and typically grow much faster than the autotrophs. In these types of biofilms, the heterotrophs shield the nitrifiers from the external environment while the autotrophs provide an additional source of nutrients. Biofilms are also composed of inert biomass produced by dead and decaying cells, and extracellular polymeric substances (EPS), which comprise the glue that holds the biofilm together and bonds it to surfaces [35].

The biofilm structure has many important properties that give cells in biofilms an advantage over free-floating cells. For example, biofilms are well known to have a much higher resistance to antimicrobial treatment [13, 14, 60, 67]. While the precise reason is not known, several different hypotheses have been proposed for how biofilm formation enhances antimicrobial resistance [67]. For example, the EPS may have a large binding-site capacity that inactivates antimicrobials before they are able to penetrate the full depth of the biofilm. This gives bacteria deep in the biofilm protection from treatment, and they are consequently more difficult to eradicate.

In addition to the structural components of biofilms, there are other important soluble diffusing substances present in biofilms. Some substances, such as substrates, may come from external sources, while other substances, such as certain soluble microbial products, are produced by the biofilms themselves. For example, several bacterial species are known to produce specialized molecules, called signals, which allow the bacteria to monitor their local population densities [23, 24]. When the signal concentration reaches a critical threshold, the bacteria are said to be *quorum-sensing*, and upregulate certain genes, leading to a change in behavior [15, 51].

Computer simulation of biofilms have generally been done using discrete type models. The most notable among these methods is the individual based model (IbM) developed by the Delft group [32, 33, 68]. In this model, biofilms are represented by a collection of packed spheres. Each sphere is divided into mass fractions representing bacteria, EPS, and other residual biomass. Diffusing substances, such as substrates, are modeled using continuum equations. Biomass reproduction is simulated through two distinct mechanisms: growth of spheres and sphere division. Because the growth model leads to overlaps between neighboring spheres, an elastic relaxation problem is solved to allow overlapping spheres to push apart until all overlaps are resolved. While this approach has produced interesting simulations, e.g., [74], the growth rules can be viewed as arbitrary.

Other discrete type models include cellular automata based methods [5, 20, 21, 27–29, 48–50, 52, 73]. In this case, simple rules are applied to each cell, or collection of cells, for substrate consumption, soluble products production, and reproduction. As in the IbM method, the substrate and soluble products are typically modeled as a continuum, while the biomass is represented discretely.

Continuum models of biofilms, based on partial differential equations, have only recently been developed [17]. These models rely on the level set method for tracking the evolving biofilm surface, while additional variables in the domain determine the concentrations of substrates and other soluble products. In this chapter, we provide a description of how the level set method can be used to simulate biofilm growth and structure.

2. GENERAL MATHEMATICAL MODEL

Mathematical models of biofilms vary significantly in complexity. At the simplest end of the spectrum is a single-species biofilm, with a single substrate using linear kinetics. At the other end of the spectrum, multiple species, multiple substrates, inert biomass, signaling molecules, other soluble products, and nonlinear reaction kinetics can all be included. The level of detail and the characteristics of the reaction kinetics must be tailored to the particular biofilm system, as well as the questions about biofilms that are to be explored. In this section, a generic model suitable for use with the level set method is presented.

The physical structure of a bacterial biofilm consists of more than just cells. The cells are held together, and attach to solid surfaces by using extracellular polymeric substances (EPS), which are adhesive polysaccharides excreted by the cells. Other inert biomass, such as the nonbiodegradable portion of dead cells also accumulates in biofilms [34, 35].

In this model, the different components of the biofilm will be broken down into their local volume fraction within the biofilm. Let $X_1, \ldots, X_n$ be functions that represent the volume fraction of n different bacterial species and other space-occupying biofilm components, such as EPS and inert biomass. These functions may vary in space and evolve in time. If these account for all the components of the biofilm, then the volume fractions must sum to one:

$$\sum_{i=1}^{n} X_i = 1. \tag{1}$$

Each of these components will be assumed to have a constant density given by ρ_i.

In addition to the structural components of the biofilm, there are also many diffusing substances that are carried by the fluid, including various substrates, signaling molecules, and soluble products which contribute to the growth, development, and behavior of the biofilm. The variables $S_1, \ldots, S_m$ will represent the concentration of these diffusing substances in the fluid.

With these variables defined, a basic mathematical model of the system can be constructed. The system is composed of two sets of equations. The first set comes from the mass balance equations for the biofilm components, X_i, and the second set comes from the mass balance equations for the diffusing substances, S_j. These combined with appropriate boundary conditions form the mathematical description

of a biofilm and its environment. The model presented here is a multidimensional generalization of the work in [71].

2.1. Biofilm Structure Equations

The mass balance for species i is a combination of a growth/decay reaction term with the local mass flux:

$$\rho_i \frac{\partial X_i}{\partial t} = \sum_{j=1}^{n} \mu_i^j(S_1, \ldots, S_m)\rho_j X_j - \nabla \cdot (\mathbf{v}\rho_i X_i). \tag{2}$$

where μ_i^j is the rate that species i is produced by species j, and $\mathbf{v}$ is the velocity of the biomass. Note that for most cases $\mu_i^j \equiv 0$ for $i \neq j$, but not always. For example, if species i produces EPS, then $\mu_{\text{EPS}}^1 \neq 0$. Likewise, EPS is not able to reproduce itself; hence $\mu_{\text{EPS}}^{\text{EPS}} = 0$.

The velocity, $\mathbf{v}$, is generated by changes in total biomass, which are not able to go through the substratum, the surface on which the biofilm is growing. The velocity is computed from the mass balance equations. Since the sum of the volume fractions is constant, then

$$\begin{aligned}
0 &= \frac{\partial}{\partial t} \sum_{i=1}^{n} X_i, \\
&= \sum_{i=1}^{n} \left[\sum_{j=1}^{n} \mu_i^j(S_1, \ldots, S_m) \frac{\rho_j}{\rho_i} X_j - \nabla \cdot (\mathbf{v} X_i) \right], \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} \mu_i^j(S_1, \ldots, S_m) \frac{\rho_j}{\rho_i} X_j - \nabla \cdot \left(\mathbf{v} \sum_{i=1}^{n} X_i \right), \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} \mu_i^j(S_1, \ldots, S_m) \frac{\rho_j}{\rho_i} X_j - \nabla \cdot \mathbf{v}.
\end{aligned}$$

Next, we make the assumption that the velocity field is irrotational so that $\mathbf{v}$ can be derived from a potential, $\mathbf{v} = \nabla\Phi$ for some function Φ. This gives an equation to determine the velocity field:

$$\nabla^2 \Phi = \sum_{i=1}^{n} \sum_{j=1}^{n} \mu_i^j(S_1, \ldots, S_m) \frac{\rho_j}{\rho_i} X_j. \tag{3}$$

This equation is coupled with boundary conditions including no flux through solid surfaces, i.e., $\partial\Phi/\partial n = 0$, and constant $\Phi = 0$ at the biofilm/liquid interface.

Once the velocity potential is computed, then the normal speed of the biofilm/liquid interface is given by $\partial\Phi/\partial n$.

2.2. Diffusing Substances Equations

The concentrations of the various diffusing substances play an important role in determining the evolution of the biofilm. These substances are either consumed or produced by the different components of the biofilm, and are also affected by the fluid flow over the biofilm. For this discussion, we will consider only laminar fluid flow so that there is a linear boundary layer between the bulk fluid flow, which has fixed bulk concentrations of the diffusing substances, and the biofilm surface.

The mass balance equation for substance S_j is given by

$$\frac{\partial S_j}{\partial t} = \sum_{i=1}^{n} \eta_j^i(S_1, \ldots, S_m)X_i - \beta_j S_j + D_j \nabla^2 S_j, \tag{4}$$

where η_j^i is the rate of production ($\eta_j^i > 0$) or consumption ($\eta_j^i < 0$) of substance S_j by X_i, β_j is the rate of degradation of S_j due to hydrolysis or other effects, and D_j is the diffusion coefficient. The diffusion coefficient may vary by substance, and also by location, e.g., inside or outside the biofilm [59].

For the length scales at work in biofilms, the time scale for diffusion is typically on the order of seconds, while the time scale for growth is on the order of days. Consequently, the mass balance equations for the diffusing substances are taken to be quasi-steady state. Thus, the equations to be solved are

$$D_j \nabla^2 S_j = -\sum_{i=1}^{n} \eta_j^i(S_1, \ldots, S_m)X_i + \beta_j S_j. \tag{5}$$

Note that the summation in [1.5] is nonzero only inside the biofilm, and is zero outside the biofilm. Furthermore, since the diffusion coefficient is typically smaller inside the biofilm compared to outside, then [1.5] is really a combination of two elliptic equations coupled together through a shared boundary, the interface of the biofilm, along with the interface conditions:

$$\left[D_j \frac{\partial S_j}{\partial n} \right] = 0, \quad [S_j] = 0. \tag{6}$$

Here, the brackets $[\cdot]$ indicate the jump across the biofilm/liquid interface. The first condition is continuity of flux, and the second condition is continuity of concentration.

The boundary conditions for [1.5] depend on the particular system being modeled. One example of boundary conditions is given in the next section.

All the equations discussed in this section are summarized in Figure 1.

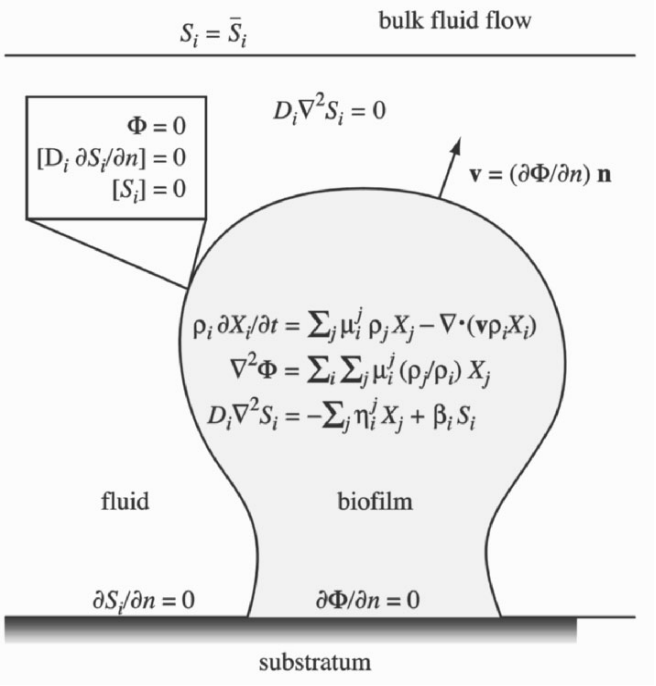


Figure 1. Summary of biofilm model equations.

3. EXAMPLE: QUORUM-SENSING IN *P. AERUGINOSA* BIOFILMS

As an example of how the mathematical framework can be used to model a specific biofilm system, consider a simple model for describing a single species *Pseudomonas aeruginosa* biofilm that exhibits quorum-sensing. A more complete description of this model can be found in [9, 10].

P. aeruginosa is a common bacterium, which is one of the leading causes of death for people suffering from cystic fibrosis [55]. It is an opportunistic bacterium, which can infect almost any human tissue which has compromised defenses. For example, it can cause a variety of infections in patients with severe burns, cancer, or AIDS.

One interesting aspect of this particular type of biofilm is that these bacteria use a signaling molecule to monitor their local population density [44, 45]. For purposes of the model, the signaling molecule is considered one of the diffusing substances. It is produced by the bacteria, degrades through hydrolysis (analogous to [56]), and diffuses through the biofilm, out into the surrounding fluid, and carried downstream by the bulk fluid flow. If sufficient bacteria are present, then the

concentration of the signal builds up. When the concentration reaches a threshold, the bacteria upregulate certain gene expressions and are said to be *quorum-sensing* [15, 51]. Quorum-sensing bacteria increase signal molecule production tenfold, and also release virulence factors which lead to tissue damage. Understanding quorum-sensing is an important goal in the effort to treat cystic fibrosis.

For this model, the biofilm is assumed to consist of two components: a single bacterial species, X_1, and EPS, X_{EPS}. In this model, inert biomass will be included in the X_{EPS} term. The substrate is assumed to be saturating, so the reaction rates are limited by availability of O_2 represented by S_1. The signal molecule concentration will be given by S_2. Following the description in the previous section and using the reactions detailed in [9, 10], the structure equations are

$$\frac{\partial X_1}{\partial t} = (Y_{x/o}\hat{q}_o - b)\frac{S_1}{K_o + S_1}X_1 - \nabla \cdot (\mathbf{v}X_1), \tag{7}$$

$$\frac{\partial X_{\text{EPS}}}{\partial t} = \frac{\rho_1}{\rho_{\text{EPS}}}((1 - f_D)b + Y_{w/o}\hat{q}_o)\frac{S_1}{K_o + S_1}X_1 - \nabla \cdot (\mathbf{v}X_{\text{EPS}}). \tag{8}$$

In (7), the first term in μ_1^1 gives the rate of reproduction and the second term gives the rate of endogenous decay. The reactions require the presence of oxygen; hence the Monod term $S_1/(K_o + S_1)$. In (8), the first term in μ_{EPS}^1 represents the accumulation of inert biomass from the nonbiodegradable portions of cells undergoing endogenous decay, and the second term represents the direct production of EPS by the bacteria. Note that both equations have been divided by the corresponding densities ρ_1 and ρ_{EPS}.

The quasi-steady state substrate and signal equations are

$$D_o\nabla^2 S_1 = \rho_1(\hat{q}_0 + \gamma f_D b)\frac{S_1}{K_o + S_1}X_1, \tag{9}$$

$$D_a\nabla^2 S_2 = -\rho_1\left(\beta_1\frac{S_1}{K_o + S_1} + \beta_2 + \beta_3 H(S_2 - A)\right)X_1 + \beta_4 S_2. \tag{10}$$

In (9), the two terms correspond to consumption of O_2 for reproduction and endogenous decay as described above. In (10), the first term corresponds to signal production by bacteria that have access to O_2. The second term corresponds to signal production independent of the environment. The third term corresponds to the increased signal production of bacteria that are quorum-sensing, the function $H(S_2 - A)$ is the Heaviside function, and A is the signal concentration threshold for quorum-sensing. The final term corresponds to hydrolysis of the signal molecules.

The boundary conditions for (9), (10) are no flow through the attachment surface, i.e., $\partial S_j/\partial n = 0$ and a fixed constant at the edge of the boundary layer, $S_j\big|_{\text{top}} = \bar{S}_j$. The O_2 concentration in the bulk flow is $\bar{S}_1$, while the signal concentration in the bulk flow is $\bar{S}_2 = 0$.

4. NUMERICAL IMPLEMENTATION

The simulations presented in this chapter use the level set method to track the location of the biofilm surface as it grows. This information is important for a number of the steps in the numerical solution of the mathematical model described in the previous section. We shall also see that certain steps in the process are sensitive and require specialized numerical tools in conjunction with the level set method. These new tools will be described in detail in Section 4.2.

4.1. Overview of the Algorithm

From the outermost viewpoint, the algorithm is relatively straightforward. In addition to the functions described in Section 2, let $\phi(\mathbf{x}, t)$ be the level set function that tracks the biofilm surface located at $\phi(\mathbf{x}, t) = 0$, where $\phi(\mathbf{x}, t) < 0$ inside the biofilm. The steps of the algorithm are:

1. Solve diffusing substances equations [1.5] for fixed values of X_i, ϕ. Note that in [1.5] $\eta_i^j = 0$ when $\phi(\mathbf{x}, t) > 0$, since this is outside the biofilm.
2. Compute the velocity potential, Φ, using (3). This is done only in the region where $\phi(\mathbf{x}, t) < 0$.
3. Update the volume fractions X_i using (2).
4. Update the level set function $\phi(\mathbf{x}, t)$ using normal speed $F = \partial\Phi/\partial n$.
5. Return to step 1.

Before proceeding, we make a few comments about the algorithm:

- Both of the first two steps in the above algorithm involve solving a set of nonlinear elliptic equations. In Step 1, the equations are solved on the whole domain, but with an interface internal to the domain that imposes additional constraints. In Step 2, the equations are solved only inside the region where $\phi < 0$, hence it is an additional nonlinear elliptic equation solved on an irregularly shaped domain. Both of these steps will require an accurate solver since there are often sharp boundary layers near the biofilm/fluid interface due to stiff reactions.
- Step 3 will use standard conservative upwind methods for propagating the volume fractions as the biofilm expands.
- In order to advance ϕ in Step 4, we will require evaluation of the normal derivative of the velocity potential, and then use velocity extensions.

In the remainder of this section, we will detail how these steps are implemented.

4.2. Solving Elliptic Equations with Irregular Boundaries

Calculating the concentrations of the diffusing substances and the velocity potential require solving a system of elliptic equations. It is very common for the velocity of a moving interface to be determined in some way by the solution of a connected elliptic equation. In some cases, the equations are relatively tame, and in other cases, such as the application presented here, the solutions have narrow boundary layers near the interface that must be adequately resolved. Most importantly, the interface speed often depends on the normal derivative of the solution to the elliptic equations evaluated at the interface. Proper evaluation of this expression requires special care.

There are a number of options available to accurately solve elliptic equations with arbitrary boundaries. Popular methods include the finite-element method with a conforming mesh [4, 7], boundary integral methods [25], the immersed interface method [36, 37], and the immersed boundary method [46, 47]. Each of these methods has its advantages and disadvantages. Traditional finite-element formulations can offer excellent accuracy, and can easily handle internal interfaces, but a moving interface will force a re-meshing every time step adding a significant expense. Boundary integral methods offer excellent accuracy but can be difficult to formulate for complicated nonlinear equations. Both the immersed interface method and the immersed boundary method are attractive because they are designed to work with moving interfaces on fixed meshes. The immersed boundary method uses extra delta-function approximations near the boundary to account for interface conditions, then uses a standard finite-difference stencil. The immersed interface method uses modified finite-difference stencils near the boundary to incorporate the interface conditions. The immersed boundary method is attractive for its simplicity, but it is only first order accurate, and not able to produce reliable gradients at the interface. The immersed interface method is globally second order accurate but can still have difficulty with approximating gradients at the interface.

A recently developed alternative to these methods, which works well with the level set method, is the eXtended Finite Element Method (X-FEM) [19, 40]. The method uses a fixed finite-element mesh with the standard finite-element basis functions, coupled with additional specialized basis functions that capture interface conditions and other localized behavior. The method offers the accuracy of the finite-element method without requiring a conforming mesh, hence it does not require remeshing. Furthermore, the additional basis functions can be used to obtain sub-grid accurate solutions, reducing the need for adaptive mesh refinement. This is particularly important in the context of the level set method, where the accuracy of the solution and its gradient *at the interface* is critical.

4.2.1. The eXtended Finite Element Method

For simplicity, the version of X-FEM for solving linear elliptic equations in two dimensions is presented here. The X-FEM has been successfully applied to

a growing number of applications, including crack growth [11, 18, 40, 62, 65], material interfaces and voids [64], solidification [22, 30], and moving solids in fluid flow [70].

Consider the elliptic equation

$$\nabla \cdot (\beta \nabla u) + \kappa u = f \tag{11}$$

in a domain Ω. Within Ω is an interface Γ. The coefficients β, κ, and f may be discontinuous across Γ, and jump conditions for solution u must also be specified to close the system of equations. This type of problem appears across a broad spectrum of applications, and in particular, in [1.3] and [1.5].

The numerical solution of (11) is represented as a linear combination of basis functions and so-called enrichment functions:

$$u(x,y) \approx \sum_{n_i \in N} \varphi_i(x,y) u_i + \sum_{n_j \in N_E} \varphi_j(x,y) \psi(\phi) a_j, \tag{12}$$

where N is the set of all nodes in the domain, N_E is the set of enriched nodes, φ_i is a standard finite-element basis function corresponding to node n_i, ψ is an enrichment function, and ϕ is the signed distance function from Γ. Coefficients u_i and a_j are the unenriched and enriched degrees of freedom, respectively. In practice, set N_E is only a small subset of N, consisting of those nodes bounding elements that are cut by Γ.

Following the ordinary transformation of (11) into its equivalent weak formulation, coefficients u_i, a_j are computed by solving system $\mathbf{Ax} = \mathbf{b}$, where matrix $\mathbf{A}$ can be broken down into four submatrices:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}^{UU} & \mathbf{A}^{UE} \\ \mathbf{A}^{EU} & \mathbf{A}^{EE} \end{bmatrix}, \tag{13}$$

where $\mathbf{A}^{UU}$ is the standard finite-element matrix obtained without enrichment, and the remaining matrices arise from the addition of the enrichment function. Matrices $\mathbf{A}^{UE}$, $\mathbf{A}^{EU}$, and $\mathbf{A}^{EE}$ are much smaller than $\mathbf{A}^{UU}$ because N_E is much smaller than N. Furthermore, all the matrices are sparse. A detailed description of the contents of these matrices are beyond the scope of this book, but a description can be found in [69].

Before demonstrating the effectiveness of this approach, we make a few comments:

- Contrary to the usual finite-element approximation, it is no longer true that $u(x_i, y_i) = u_i$, where n_i is located at (x_i, y_i), due to the inclusion of the enrichment function. This does not introduce any additional complications, but is important to remember.

Table 1. Comparison of system sizes and non-zero density of the matrices generated by the X-FEM and the IIM

	X-FEM		IIM	
n	System size	Density	System size	Density
19	520	2.07396%	400	1.09250%
39	1,840	0.54277%	1,600	0.29234%
79	6,880	0.13840%	6,400	0.07558%
159	26,560	0.03490%	25,600	0.01921%
319	104,320	0.00876%	102,400	0.00484%

- Multiple enrichment functions may be added without difficulty. This is done by adding additional sums, one for each extra enrichment function, to (12). For most applications, only one or perhaps two enrichment functions are used.

- The most common, and simplest, enrichment functions are a step function [66] and a linear ramp function [31]. However, enrichment functions specific to a particular application can also be used, for example a square root singularity function around crack tips [3]. In each case, the enrichment functions are given as functions of the signed distance to the interface. This is easily evaluated when the method is coupled with the level set method.

- The resulting matrix that must be inverted for the X-FEM is a little larger, and not quite as sparse as the one generated by the IIM. In Table 1, the dimensions and sparsity for a sample problem are compared. The computational cost of solving the two systems are comparable.

To illustrate the comparative accuracy of the X-FEM, we compared the X-FEM with the immersed interface method (IIM) on several problems [69]. We present one of those examples here for which an exact solution is known.

Consider the differential equation

$$\nabla^2 u = 0 \tag{14}$$

in the domain $[-1,1] \times [-1,1]$, with Γ consisting of a circle of radius $\frac{1}{2}$ in the center. At the interface, the following jump conditions are specified:

$$[u] = e^x \cos y, \tag{15}$$

$$\left[\frac{\partial u}{\partial n}\right] = 2e^x(x\cos y - y\sin y). \tag{16}$$

Table 2. Global max-norm errors for the X-FEM and the IIM solving system (14)–(16)

n	X-FEM	IIM
19	1.7648×10^{-4}	3.6253×10^{-3}
39	6.0109×10^{-5}	4.6278×10^{-4}
79	1.7769×10^{-5}	3.0920×10^{-4}
159	4.8626×10^{-6}	1.1963×10^{-4}
319	1.2362×10^{-6}	4.5535×10^{-5}

Table 3. Max-norm errors measured on the interface for the X-FEM and the IIM solving system (14)–(16)

n	X-FEM	IIM
19	4.7842×10^{-4}	4.0230×10^{-3}
39	1.0659×10^{-4}	5.7563×10^{-4}
79	2.8361×10^{-5}	3.1617×10^{-4}
159	7.3603×10^{-6}	1.2004×10^{-4}
319	2.0634×10^{-6}	4.5526×10^{-5}

The exact solution for this system is

$$u(x,y) = \begin{cases} e^x \cos y & r \leq \frac{1}{2} \\ 0 & r > \frac{1}{2} \end{cases}. \tag{17}$$

Both the X-FEM and the IIM were used to solve system (14)–(16). In Table 2, the global accuracy of the two methods are compared. This measures the accuracy over the whole domain Ω. While the convergence rates are the same, Table 2 shows how the X-FEM produces an order of magnitude better solution than the IIM. If the error is measured only around the interface, the X-FEM again outperforms the IIM as shown in Table 3.

As noted earlier, while the global error is important, it is critical that the gradient of the solution be accurately computed from the solution. In this regard, the X-FEM performs dramatically better, as shown in Table 4. Taking the derivative of the numerical solution is expected to produce lower accuracy as is observed. However, while the X-FEM produces two or three digits of accuracy, the IIM is unable to produce any digits of accuracy for this problem. It is worth noting that this problem is typical of the difficulty faced in (11). One explanation for the dramatic difference in the results is that the immersed interface method is only computed at the grid points, so interpolation is required to obtain values at the interface, which does not generally pass directly through grid points. On the other

Table 4. Max-norm errors of the normal derivative to the interface, measured on the interface for the X-FEM and the IIM solving system (14)–(16)

n	X-FEM	IIM
19	5.6520×10^{-2}	$3.0009 \times 10^{+1}$
39	2.4190×10^{-2}	$5.5185 \times 10^{+1}$
79	9.4512×10^{-3}	$1.2034 \times 10^{+2}$
159	7.1671×10^{-3}	$2.6466 \times 10^{+2}$
319	2.6865×10^{-3}	$5.2870 \times 10^{+2}$

Table 5. Max-norm errors of the normal derivative to the interface, measured on the interface for the X-FEM and the IIM solving the system (18)

n	X-FEM	IIM
19	1.9×10^{-1}	$2.8 \times 10^{+2}$
39	4.1×10^{-2}	$6.9 \times 10^{+2}$
79	1.1×10^{-2}	$8.3 \times 10^{+2}$
159	3.6×10^{-3}	$8.0 \times 10^{+2}$
319	2.3×10^{-3}	$8.6 \times 10^{+2}$

hand, the X-FEM, through the use of suitable enrichment functions, can provide more accurate solutions between the grid points.

Comparable results are obtained for a problem much more similar to (11). In this example, the two methods are compared solving

$$\nabla^2 u - \lambda^2 u = 0, \tag{18}$$

where $\lambda >> 1$ inside a circle, and $\lambda = 0$ outside the circle. Table 5 shows the errors in computing the normal derivative to the interface. This example is significantly more difficult, because of the large value of λ, and hence an additional enrichment function was used to improve the accuracy of the X-FEM. In this case, an exponential function was used, since it is a reasonable approximation to the solution in a neighborhood of the interface. A side view of the solution for a circle of radius $1/4$ is shown in Figure 2. With the inclusion of this extra enrichment function, the X-FEM is able to produce at least two digits of accuracy, while the IIM is not.

While the X-FEM does outperform the IIM in terms of accuracy and solvability of the resulting linear system, this does come at a price. The IIM is significantly easier to program than the X-FEM. In particular, generating the entries in the

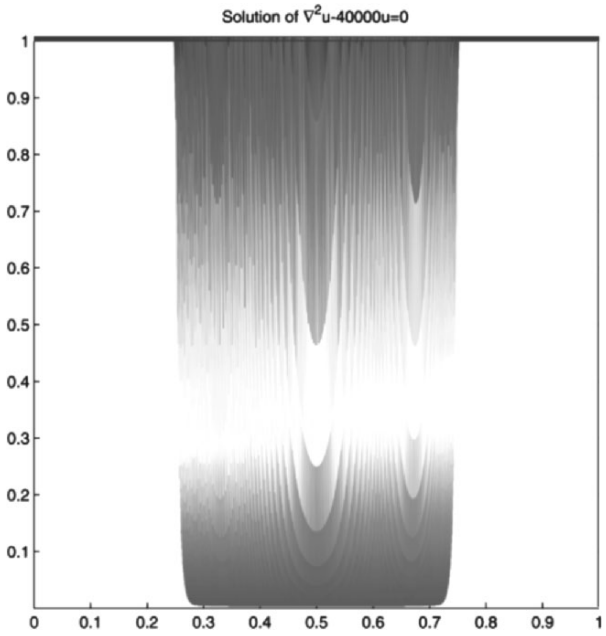


Figure 2. Sample solution of the Helmholtz equation on a circle using the X-FEM with exponential enrichment. See attached CD for color version.

matrices that include enrichment functions requires careful integrations to be carried out on the corresponding elements. The matrix for the IIM, on the other hand, is much easier to generate. Consequently, the X-FEM can be somewhat more expensive to generate the linear system. However, on the whole, the most expensive step in this procedure is the actual inversion of the linear system, and that is essentially the same for both methods.

4.2.2. Coupling the X-FEM to the Level Set Method

The merits of coupling level sets to the extended finite-element method was first explored in [64], and subsequently its advantages further realized in [12, 26, 31, 41, 61, 63, 66]. The two methods make a natural pair of methods where:

- Level sets provide greater ease and simplification in the representation of geometric interfaces.
- The X-FEM, given the right enrichment functions, can accurately compute solutions of elliptic equations that are often required for computing the interface velocity.

- Geometric computations required for evaluating the enrichment functions (such as the normal or the distance to the interface) are readily computed from the level set function [66].

- The nodes to be enriched are easily identified using the signed distance construction of the level set function [61, 63, 64, 66].

4.2.3. Examples of Solving the Substrate and Velocity Potential Equations

In Figure 3 a sample graph of the solution of (9) is shown. Note how the solution is essentially a linear ramp matched with almost constant zero solution in the interior of the biofilm. The sharp transition between the two regions is close to the biofilm surface, and plays a critical role in determining the velocity potential, and hence the growth rate of the biofilm.

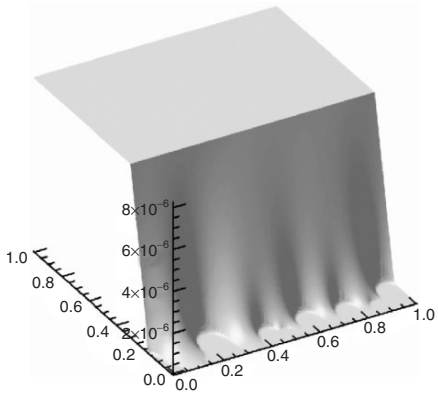


Figure 3. Sample solution of substrate equation (9).

The velocity potential equation that is obtained from (7), (8) is given by

$$\nabla^2\Phi = \left(Y_{x/o}\hat{q}_0 - b + \frac{\rho_1}{\rho_{\text{EPS}}}((1-f_D)b + Y_{w/o}\hat{q}_0)\right)\frac{S_1}{K_o + S_1}X_1. \qquad (19)$$

Figure 4 shows a sample graph of the solution of (19) for the substrate concentration computed in Figure 3. Note that this equation is solved only in the interior of the biofilm, and is constant zero outside the biofilm. Larger negative values are indicative of greater local growth rates.

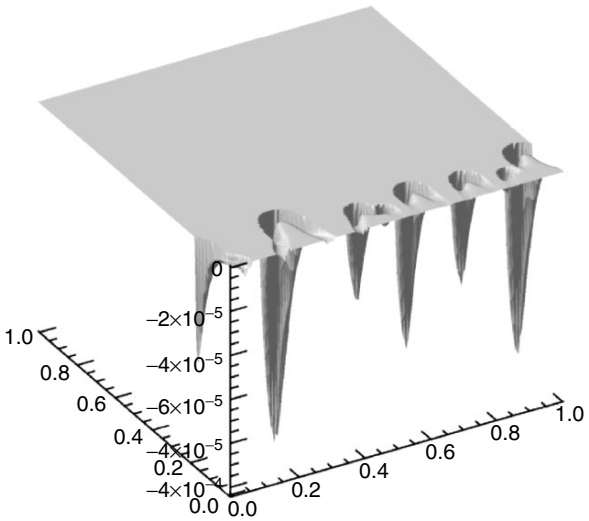


Figure 4. Sample solution of velocity potential equation (19).

4.3. Advancing the Volume Fractions

Once the substrates and velocity potential are computed, volume fractions X_i must be updated inside the biofilm using (2). It is sufficient to track all but one of the volume fractions, with the last one easily recovered from (1). Solving (2) is straightforward using upwind finite differences.

First, the velocity is determined from the velocity potential, $\mathbf{v} = \nabla\Phi$, by using central difference approximations at the grid points. Special care must be taken near the biofilm/fluid interface to ensure that Φ is evaluated only inside the biofilm, since Φ is undefined otherwise.

To advance in time, (2) is rewritten using the velocity potential:

$$\begin{aligned}
\frac{\partial X_i}{\partial t} &= \sum_{j=1}^{n} \mu_i^j \frac{\rho_j}{\rho_i} X_j - \nabla \cdot (\mathbf{v} X_i). \\
\frac{\partial X_i}{\partial t} &= \sum_{j=1}^{n} \mu_i^j \frac{\rho_j}{\rho_i} X_j - X_i \nabla \cdot \mathbf{v} - \mathbf{v} \cdot \nabla X_i \\
\frac{\partial X_i}{\partial t} &= \sum_{j=1}^{n} \mu_i^j \frac{\rho_j}{\rho_i} X_j - X_i \nabla^2 \Phi - \mathbf{v} \cdot \nabla X_i \\
\frac{\partial X_i}{\partial t} &= \sum_{j=1}^{n} \mu_i^j \frac{\rho_j}{\rho_i} X_j - X_i \sum_{k=1}^{n} \sum_{j=1}^{n} \mu_k^j \frac{\rho_j}{\rho_k} X_j - \mathbf{v} \cdot \nabla X_i. \qquad (20)
\end{aligned}$$

The spatial derivative of X_i is computed using upwind finite-differences according to the direction of the velocity vector, $\mathbf{v} = (u, v)$:

$$\begin{aligned}\mathbf{v} \cdot \nabla X_i \approx \min(u,0) &\left(\frac{X_i(x_j + \Delta x, y_k) - X_i(x_j, y_k)}{\Delta x} \right) \\ + \max(u,0) &\left(\frac{X_i(x_j, y_k) - X_i(x_j - \Delta x, y_k)}{\Delta x} \right) \\ + \min(v,0) &\left(\frac{X_i(x_j, y_k + \Delta y) - X_i(x_j, y_k)}{\Delta x} \right) \\ + \max(v,0) &\left(\frac{X_i(x_j, y_k) - X_i(x_j, y_k - \Delta y)}{\Delta x} \right). \end{aligned} \quad (21)$$

Equation (20) is only valid inside the biofilm, but as described below, the values of X_i are extended outside the domain. Thus, the finite-difference approximation in (21) is valid for all points inside the biofilm.

Note that the volume fractions are updated before the interface is advanced according to the velocity, $\mathbf{v}$. Thus, while X_i are updated inside the biofilm here, there may be points that are currently outside the biofilm, which will subsequently be inside the biofilm after the interface is advanced. To account for this, the values of X_i are extended outside the biofilm. To do this, it is assumed that $\partial X_i / \partial n = 0$. In terms of the level set function, ϕ, which encapsulates the biofilm/liquid interface, the unit normal to the interface is given by

$$\mathbf{n} = \frac{\nabla \phi}{\|\nabla \phi\|}. \quad (22)$$

Therefore, to extend the values of X_i, we want

$$0 = \nabla X_i \cdot \mathbf{n} = \nabla X_i \cdot \frac{\nabla \phi}{\|\nabla \phi\|}. \quad (23)$$

This equation is easily recognized as the equation required for computing velocity extensions in the level set method [1]. However, in this instance, the X_i data need only be extended one layer of additional grid points outside the biofilm, as illustrated in Figure 5.

The extended value of X_i is computed by discretizing ((23)) in the same manner as is done for velocity extensions in the level set method. For example, consider point (x_j, y_k) indicated in Figure 5. This grid point has two neighboring interior grid points. For this point, the discretized version of (23) will be

$$\left(\frac{X_{j+1,k} - X_{j,k}}{\Delta x} \right) \left(\frac{\phi_{j+1,k} - \phi_{j,k}}{\Delta x} \right) + \left(\frac{X_{j,k} - X_{j,k-1}}{\Delta y} \right) \left(\frac{\phi_{j,k} - \phi_{j,k-1}}{\Delta y} \right) = 0, \quad (24)$$

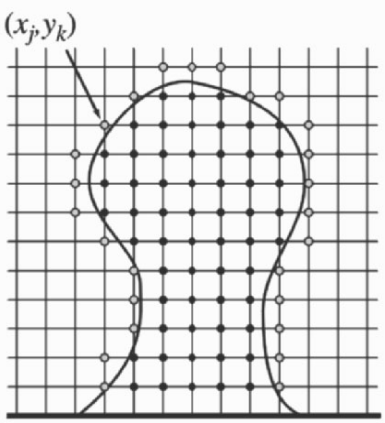


Figure 5. Illustration of where X_i values must be extended. The black dots are inside the biofilm, and the gray dots are where the extension values must be computed.

where here we have dropped the subscript i from X_i for clarity. Equation [1.24] is easily solved for the extension value $X_{j,k}$:

$$X_{j,k} = \frac{X_{j+1,k}\Delta y^2(\phi_{j,k} - \phi_{j+1,k}) + X_{j,k-1}\Delta x^2(\phi_{j,k} - \phi_{j,k-1})}{\Delta y^2(\phi_{j,k} - \phi_{j+1,k}) + \Delta x^2(\phi_{j,k} - \phi_{j,k-1})}. \tag{25}$$

Obviously, $X_{j,k}$ is a weighted average of the values inside the biofilm.

Once the X_i values have been extended, the volume fractions update is complete, and the location of the biofilm/fluid interface can now be updated using the level set method.

4.4. Advancing the Biofilm/Fluid Interface

As noted earlier, the location of the biofilm/fluid interface is maintained by the level set method [42], using level set function ϕ, where, by convention, the interior of the biofilm is indicated by $\phi < 0$, and the biofilm/fluid interface is given by $\Gamma = \phi^{-1}(0)$. Function ϕ is updated using the level set evolution equation [42]:

$$\phi_t + F\|\nabla\phi\| = 0. \tag{26}$$

The key to all level set method applications is the generation of speed function F. Once F is determined, ϕ is easily updated via (26) using upwind finite-difference methods borrowed from techniques used to solve hyperbolic conservation laws.

In the case of the biofilm application presented here, speed function F may only be determined on the interface, and is given by

$$F\big|_{\Gamma} = \frac{\partial \Phi}{\partial n}, \tag{27}$$

where Φ is the velocity potential discussed in Section 4.2. Once $F\big|_{\Gamma}$ is computed, it must be extended off the interface to the rest of the domain so that we can use (26). This is accomplished through a velocity extension, discussed later in this section.

Since Φ is only defined on the inside of the biofilm, then $\partial\Phi/\partial n$ must be computed using a one-sided finite-difference approximation at the interface. Suppose $\mathbf{x} \in \Gamma$ is where $\partial\Phi/\partial n$ is to be computed; then it is approximated by

$$\frac{\partial \Phi}{\partial n} \approx \frac{\Phi(\mathbf{x}) - \Phi(\mathbf{x} - \epsilon\nabla\phi)}{\epsilon\|\nabla\phi\|} = \frac{-\Phi(\mathbf{x} - \epsilon\nabla\phi)}{\epsilon\|\nabla\phi\|}, \tag{28}$$

where $\epsilon = \sqrt{\Delta x^2 + \Delta y^2}$ is a small constant, and we are using the boundary condition on the velocity potential, $\Phi(\mathbf{x}) = 0$. Evaluation of $\Phi(\mathbf{x} - \epsilon\nabla\phi)$ is done through (12).

Now that the speed function can be evaluated at an arbitrary location on the interface, the speed function must be extended to the rest of the domain. This will require using a bicubic interpolation procedure to locate the interface and evaluate the velocity function, followed by the fast marching method to extend the velocity to the rest of the domain.

4.4.1. Bicubic interpolation

It was shown in [8] that a more accurate method is available, which can drive higher-order fast marching method solutions.

The underpinning of this higher degree of accuracy around the initial front is the use of a bicubic interpolation function, p, which is a second-order accurate local representation of level set function ϕ (i.e., $p(\mathbf{x}) \approx \phi(\mathbf{x})$). Interpolation function $p(\mathbf{x})$ can serve many purposes, including second-order accuracy for the distance to the zero level set, sub-grid resolution of the shape of the interface, as well as sub-grid resolution of the level set function $\phi(\mathbf{x})$ itself.

We begin with a description of the bicubic interpolation for a level set function given on a rectangular mesh. The approximation is done locally in a box of the mesh bounded by grid points, and we call them (x_i, y_j), (x_{i+1}, y_j), (x_i, y_{j+1}), and (x_{i+1}, y_{j+1}) as in Figure 6.

A bicubic interpolation, $p(\mathbf{x})$, of function $\phi(\mathbf{x})$ is a function

$$p(\mathbf{x}) = p(x, y) = \sum_{m=0}^{3} \sum_{n=0}^{3} a_{m,n} x^m y^n, \tag{29}$$

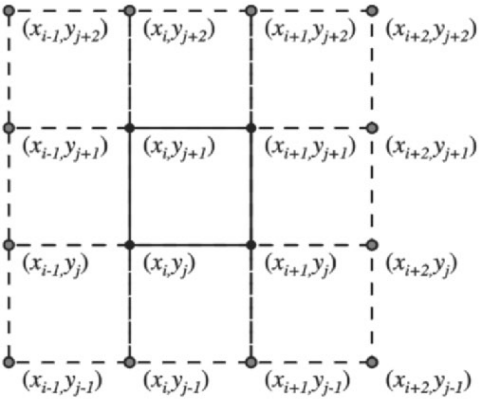


Figure 6. Sample portion of the mesh where a bicubic interpolation is used.

which solves the following set of equations:

$$\begin{aligned} p(x_k, y_\ell) &= \phi(x_k, y_\ell) \\ \frac{\partial p}{\partial x}(x_k, y_\ell) &= \frac{\partial \phi}{\partial x}(x_k, y_\ell) \\ \frac{\partial p}{\partial y}(x_k, y_\ell) &= \frac{\partial \phi}{\partial y}(x_k, y_\ell) \\ \frac{\partial^2 p}{\partial x \partial y}(x_k, y_\ell) &= \frac{\partial^2 \phi}{\partial x \partial y}(x_k, y_\ell) \end{aligned}$$

for $k = i,\ i+1;\ \ell = j,\ j+1$. This gives sixteen equations for the sixteen unknown coefficients $a_{m,n}$. Solving for the $a_{m,n}$ makes $p(x, y)$ a bicubic interpolating function of $\phi(x, y)$ on the rectangle bounded by corners (x_i, y_j), (x_{i+1}, y_j), (x_i, y_{j+1}), and (x_{i+1}, y_{j+1}). The derivatives on the right are approximated at the grid points using central finite differences.

Now, given interpolating function $p(x, y)$ in domain $[x_i, x_{i+1}] \times [y_j, y_{j+1}]$, and given point (x_0, y_0) in that domain, we compute the distance between (x_0, y_0) and the zero level curve of $p(x, y)$. Point (x_1, y_1) on the zero level curve closest to (x_0, y_0) must satisfy two conditions:

$$p(x_1, y_1) = 0, \tag{30}$$

$$\nabla p(x_1, y_1) \times ((x_0, y_0) - (x_1, y_1)) = 0. \tag{31}$$

Equation (30) is a requirement that (x_1, y_1) must be on the interface. Equation (31) is a requirement that the interface normal, given by $\nabla p(x_1, y_1)$, must be aligned with the line through points (x_0, y_0) and (x_1, y_1). Equations (30), (31) are solved

simultaneously using Newton's method. Typically, less than five iterations are necessary in order to achieve sufficient accuracy.

To compute the speed at grid point (x_i, y_j) off the interface, a bicubic interpolant is generated for the box containing the interface and grid point (x_i, y_j). The point on the interface nearest to (x_i, y_j), labeled (x_1, y_1) in the discussion above, is determined by solving (30), (31). Given the interface speed $F(x_1, y_1)$ from (28), the speed function value is then $F(x_i, y_j) = F(x_1, y_1)$. These points are labeled as accepted points for purposes of the Fast Marching Method described below. Once all the points near the interface have been given initial speed function values, the remainder of the grid points have their speed function values computed using the velocity extension method described next.

4.4.2. The Fast Marching Method and Velocity Extensions

Velocity extensions for the level set method were introduced in [1] and are computed using the fast marching method [57, 58]. The fast marching method is an optimal method for solving the following equation:

$$G\|\nabla\phi\| = 1, \tag{32}$$

where G is the speed of the interface. What makes the fast marching method fast is the fact that (32) can be solved with one pass over the mesh. For the purposes of the biofilm application, we only require the fast marching method to compute the velocity extension, and hence (32) will be solved with $G \equiv 1$.

The key to solving (32) in one pass is to traverse the mesh in the proper order. The grid points must be evaluated in the order of increasing ϕ. This is accomplished by using a sorted heap that always keeps track of which grid point is to be evaluated next. To begin, the set of grid points is divided into three disjoint sets, *accepted* points A, *tentative* points T, and *distant* points D. The accepted points in A are points (x_i, y_j) for which the computed value of $\phi_{i,j}$ is already determined. The tentative points in T are points the (x_i, y_j) for which a tentative value for $\phi_{i,j}$ is computed. The remainder of the points are in set D. One by one, points in T are taken, in order of increasing value of $\phi_{i,j}$, from set T into A. Each time, points (x_i, y_j) in D that become adjacent to points in set A are moved into set T and a tentative value for ϕ_{ij} is computed using a finite-difference approximation for (32). The algorithm terminates when all points have migrated into set A. See (7) for an illustration of sets A, T, and D.

The implementation of the fast marching method uses upwind finite differences, where the direction of upwind differences is taken toward smaller values of ϕ. For example, suppose points (x_{i-1}, y_j), (x_i, y_{j+1}) are in set A, then $\phi_{i-1,j}$, $\phi_{i,j+1}$ are already determined, and we wish to compute a tentative value for $\phi_{i,j}$. Equation (32) is discretized using one-sided differences to obtain

$$(D_{-x}\phi_{i,j})^2 + (D_{+y}\phi_{i,j})^2 = 1. \tag{33}$$

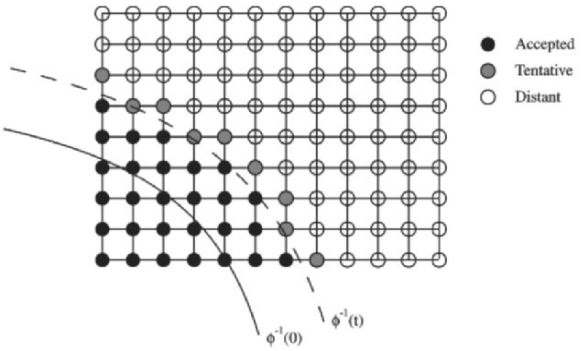


Figure 7. Illustration of the sets A, T, D, associated with the fast marching method. This figure reprinted from [8].

This equation can be rewritten as a quadratic in terms of the unknown $\phi_{i,j}$:

$$\left(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}\right)\phi_{i,j}^2-2\left(\frac{\phi_{i-1,j}}{\Delta x^2}+\frac{\phi_{i,j+1}}{\Delta y^2}\right)\phi_{i,j}+\frac{\phi_{i-1,j}^2}{\Delta x^2}+\frac{\phi_{i,j+1}^2}{\Delta y^2}-1=0. \tag{34}$$

In most cases, solving (34) will produce two solutions, one which is less than the values of $\phi_{i-1,j}$, $\phi_{i,j+1}$, and one which is greater. The larger of the two values is always chosen because of the causality assumption made by this method; values that are unknown are always greater than the known values.

The full algorithm for the fast marching method becomes:

1. Initialize all the points adjacent to the initial interface with an initial value using the bicubic interpolation, and put those points in set A. All points $(x_i, y_j) \notin A$, adjacent to a point in A are given initial estimates for $\phi_{i,j}$ by solving (32). These points are tentative points and are put in the set T. All remaining points unaccounted for are placed in D and given initial values of $\phi_{i,j} = +\infty$.

2. Choose point $(x_i, y_j) \in T$ that has the smallest value of $\phi_{i,j}$ and move it into A.

3. Any point that is adjacent to (x_i, y_j) (i.e., points (x_{i-1}, y_j), (x_i, y_{j-1}), (x_{i+1}, y_j), (x_i, y_{j+1})) and that is in T has its value $\phi_{i,j}$ recalculated using (32). Any point adjacent to (x_i, y_j) and in D has its value $\phi_{i,j}$ computed using (32) and is moved into set T.

4. If $T \neq \emptyset$, go to step 2.

As observed in [1], if the level set speed function is F, then the velocity that preserves the signed distance function solves

$$\nabla F \cdot \nabla \phi = 0, \tag{35}$$

where ϕ is the level set function. If we assume $F_{i,j}$ is given for all points (x_i, y_j) initially in set A, then the remainder of the values for $F_{i,j}$ are computed using the same upwind direction finite difference as the fast marching method. Thus, considering again the example where (x_{i-1}, y_j) and $(x_i, y_{j+1}) \in A$, (35) is discretized to become

$$\left(\frac{F_{i,j} - F_{i-1,j}}{\Delta x}\right)\left(\frac{\phi_{i,j} - \phi_{i-1,j}}{\Delta x}\right) + \left(\frac{F_{i,j+1} - F_{i,j}}{\Delta x}\right)\left(\frac{\phi_{i,j+1} - \phi_{i,j}}{\Delta x}\right) = 0. \tag{36}$$

In this case, $\phi_{i,j}$ is known, so [1.36] is easily solved for $F_{i,j}$.

4.5. Final Detailed Algorithm

Now that the pieces have all been described, the full algorithm can be assembled:

1. Initialize the location of the biofilm with the surface at $\phi = 0$, the interior of the biofilm indicated by $\phi < 0$, where ϕ is the signed distance function to the interface. Also initialize all volume fractions, $0 \leq X_i \leq 1$, inside the biofilm.

2. Solve [1.5] for diffusing substances, S_j. Use the X-FEM with exponential and step enrichment functions at the interface to ensure accuracy.

3. Solve velocity potential equation (3) again using the X-FEM with step enrichment.

4. Update the volume fractions inside the biofilm using 20. Use the velocity extension equation, (23), to extend the volume fraction values outside the biofilm.

5. Use a bicubic interpolation to extend the velocity computed at the interface, given by (28) onto neighboring grid points. Then use the fast marching method to extend the velocity out to the rest of the grid points.

6. Advance the interface using the level set evolution equation, (26).

7. Go to step 2.

5. EXAMPLES

To illustrate the use of this algorithm, we compute the example described in Section 3. The biofilm is inoculated with seven hemispherical colonies of radius 10 microns at random locations along the substratum. The parameter values used are listed in Table 6.

In Figure 8, the growth of the biofilms is shown over a course of 100 days. Note how the initial colony that is the most isolated is also the most successful. Initially, there is both horizontal and vertical growth. However, once the colonies grow large enough, the taller colonies continue to grow, choking out the smaller ones.

The growth also suggests there is a tip-splitting instability in this flow. There are similarities between this type of diffusion-limited growth and instabilities observed in combustion of solid fuels [75]. This connection has so far been unexplored and is the subject of current research.

The observations about the relationship between substrate availability and the growth pattern is illustrated in Figure 9. In this graph each contour represents half the concentration of the contour above it (see labels). Several important observations can be made from this plot. First, the sharp decrease in substrate concentration at the tops of the biofilms is evident by the packed contour lines. Also, extremely low substrate concentrations are observed in the regions between the colonies, which leads to the arrested horizontal growth.

The concentration of growth at the tops of the colonies is made even more evident when looking at the contours of the velocity potential, which also shows the main areas where the substrate is being consumed. In Figure 10 a contour plot of the velocity potential is shown.

For this system, the biofilm is very uniform in its composition, with the bacteria occupying a volume fraction in range $0.5261 \leq X_1 \leq 0.5283$. A graph of the volume fraction is shown in Figure 11, where it is even more clear how uniform the composition of the biofilm is. This is consistent with the observations made in [9, 10].

One of the reasons *P. aeruginosa* is of interest is that it is a bacteria that has exhibited quorum-sensing ability. Signal molecules are produced by the bacteria, which are then diffused out into the fluid. As the biofilm grows, a concentration gradient of the signal is generated, with the maximum occurring at the bases of the biofilms. This behavior is consistent with experimental observations. For the present example, quorum-sensing occurs when the biofilm is approximately 200 microns thick and is initiated at the bases of the colonies. This is also consistent with experimental observations [16, 43]. In Figure 12, the signal concentration in the biofilm is shown just immediately prior to the onset of quorum-sensing. Once the biofilm starts quorum-sensing, signal production increases tenfold and the entire domain is above the quorum-sensing threshold.

Table 6. Table of model parameters

Name	Description	Value	Reference
ρ_x	Biomass concentration	$1.0250 \frac{\text{mg VS}}{\text{mm}^3}$	[53]
ρ_w	Inactive material concentration	$1.0125 \frac{\text{mg VS}}{\text{mm}^3}$	[53]
$Y_{x/o}$	Yield of active biomass due to substrate consumption	$0.583 \frac{\text{mg VS}}{\text{mg O}_2}$	[2, 54]
$Y_{w/o}$	Yield of EPS due to substrate consumption	$0.477 \frac{\text{mg VS}}{\text{mg O}_2}$	[2, 54]
$\hat{q}_0$	Maximum specific substrate utilization rate	$8 \frac{\text{mg O}_2}{\text{mg VS day}}$	[54]
K_o	Half-maximum-rate concentration for utilization of substrate	$5 \times 10^{-7} \frac{\text{mg O}_2}{\text{mm}^3}$	[54]
b	Endogenous decay rate coefficient	$0.3/\text{day}$	[54]
β_1	Nutrient dependent signal production rate	$10^{-4} \frac{\text{mg acyl-HSL}}{\text{mg VS day}}$	Estimated
β_2	Additional signal production rate in quorum sensing cells	$10^{-3} \frac{\text{mg acyl-HSL}}{\text{mg VS day}}$	Estimated
β_3	Nutrient independent signal production rate	$10^{-4} \frac{\text{mg acyl-HSL}}{\text{mg VS day}}$	Estimated
β_4	Signal hydrolysis rate	$10^{pH-7} \ln(2)/\text{day}$	[56]
pH		7	
D_o	Substrate diffusion coefficient in the biofilm	$146.88 \frac{\text{mm}^2}{\text{day}}$	[38, 72]
D_a	Signal diffusion coefficient in the biofilm	$146.88 \frac{\text{mm}^2}{\text{day}}$	Assumed
a_0	Signal threshold inducing quorum0sensing	$6.7 \times 10^{-9} \frac{\text{mg acyl-HSL}}{\text{mm}^3}$	Estimated
o_L	Substrate concentration in bulk liquid	$8.3 \times 10^{-6} \frac{\text{mg O}_2}{\text{mm}^3}$	[38]
f_D	Biodegradable fraction of active biomass	0.8	[54]
γ	Chemical oxygen demand of VS	$1.42 \frac{\text{mg O}_2}{\text{mg VS}}$	[54]

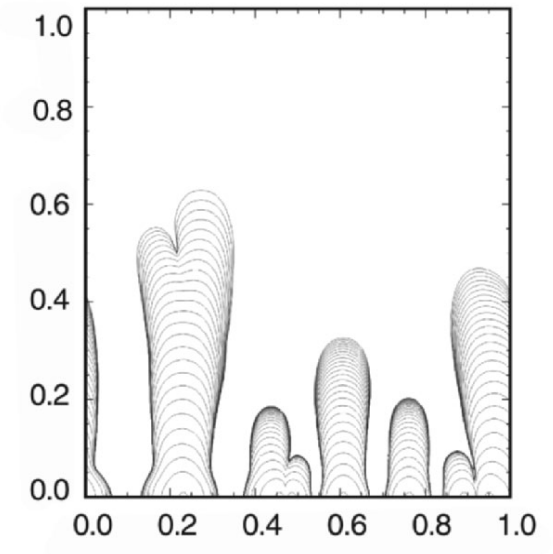


Figure 8. Sample biofilm growth.

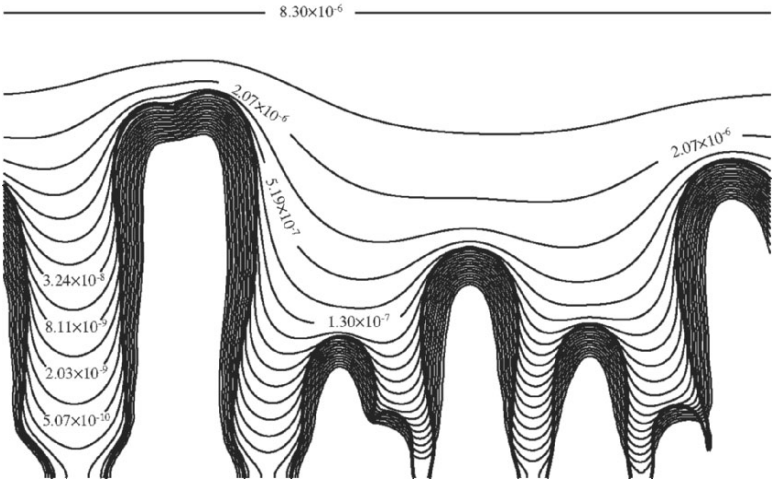


Figure 9. Substrate concentration contours for the given biofilm shown with bold lines. Each contour line indicates half the concentration of the one above it as labeled.

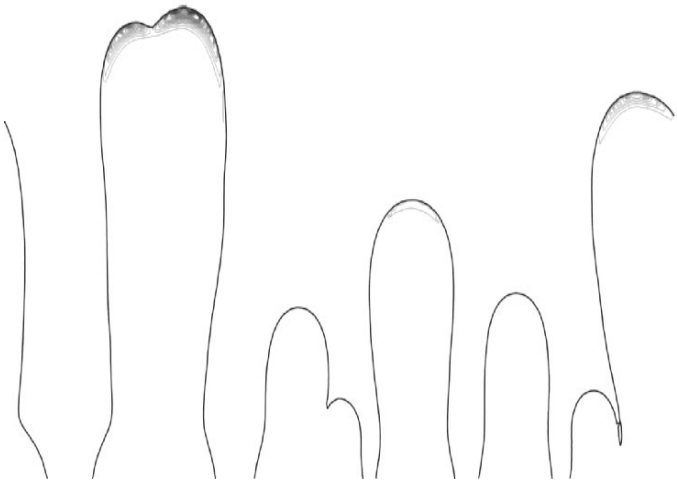


Figure 10. Contours of the velocity potential, Φ, which also indicates the rate of substrate consumption. Consumption is concentrated at the tops of the tallest colonies.

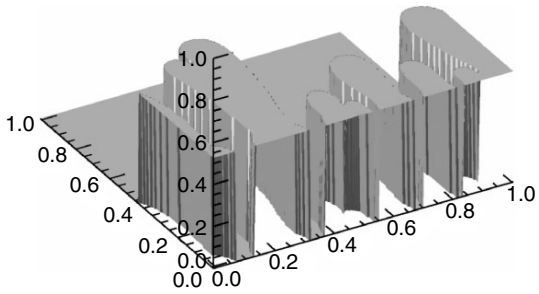


Figure 11. Graph of the volume fraction, X_1, of the bacteria in the biofilm. The composition is very uniform.

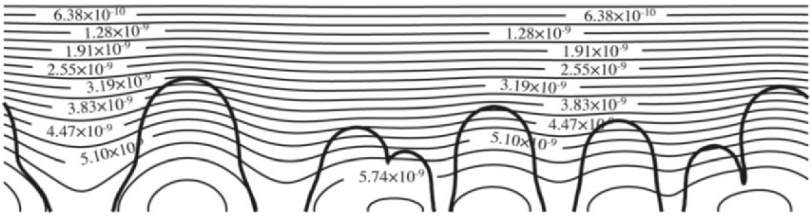


Figure 12. Signal concentration contours in and around the biofilm just prior to quorum-sensing.

6. CONCLUSION

In this chapter we showed how the level set method can be coupled with other numerical methods to model bacterial biofilms. One of the key methods that was coupled to the level set method is the eXtended Finite Element Method (X-FEM). The combination of methods is a powerful tool extending the range of problems to which the level set method can be applied.

The simulated biofilms generated by the level set method algorithm behave qualitatively very similar to real biofilms observed in experiments. Models such as this will be used to explore a variety of important biofilm phenomena. Considering that biofilms are so critically intertwined in both the environment, industry, and society, tools such as the one presented here may have applications across a wide spectrum of natural and manmade processes.

7. ACKNOWLEDGMENTS

This research was supported by a grant from the National Institutes of Health through contract R01-GM067248. Thanks to the many people who contributed to this work, including Mary Jo Kirisits, Brian Moran, Matt Parsek, Bryan Smith, and Benjamin Vaughan.

8. REFERENCES

1. Adalsteinsson D, Sethian JA. 1999. The fast construction of extension velocities in level set methods. *J Comput Phys* **48**(1):2–22.
2. Bakke R, Characklis WG, Turakhia MH, Yeh A. 1990. Modeling a monopopulation biofilm system: *pseudomonas aeruginosa*. In *Biofilms*. New York: John Wiley & Sons.
3. Belytschko T, Black T. 1999. Elastic crack growth in finite element with minimal remeshing. *Int J Num Meth Eng* **45**:601–620.
4. Bramble J, King J. 1996. A finite element method for interface problems in domains with smooth boundaries and interfaces. *Adv Comput Math* **6**:109–138.
5. Chang I, Gilber ES, Eliashberg N, Keasling JD. 2003. A three-dimensional, stochastic simulation of biofilm growth and transport-related factors that affect structure. *Microbiol–SGM* **149**(10):2859–2871.
6. Characklis WG, Marshall KC. 1990. *Biofilms*. New York: John Wiley & Sons.
7. Chen Z, Zou J. 1998. Finite element methods and their convergence for elliptic and parabolic interface problems. *J Num Math* **79**:175–202.
8. Chopp DL. 2001. Some improvements of the fast marching method. *SIAM J Sci Comp* **23**(1):230–244.
9. Chopp DL, Kirisits MJ, Moran B, Parsek M. 2002. A mathematical model of quorum sensing in a growing *P. aeruginosa* biofilm. *J Ind Microbiol Biotechnol* **29**(6):339–346.
10. Chopp DL, Kirisits MJ, Parsek MR, Moran B. 2003. The dependence of quorum sensing on the depth of a growing biofilm. *Bull Math Biol*. To appear.
11. Chopp DL, Sukumar N. 2003. Fatigue crack propagation of multiple coplanar cracks with the coupled extended finite element/fast marching method. *Int J Eng Sci* **41**:845–869, 2003.

12. Chopp DL, Sukumar N. 2003. Fatigue crack propagation of multiple coplanar cracks with the coupled extended finite element/fast marching method. *Int J Eng Sci* **41**(8):845–869.
13. Costerton JW, Lewandowski Z, Caldwell DE, Korber DR, Lappin-Scott HM. 1995. Microbial biofilms. *Annu Rev Microbiol* **49**:711–745.
14. Costerton JW, Stewart PS, Greenberg EP. 1999. Bacterial biofilms: A common cause of persistent infections. *Science* **284**:1318–1322.
15. Davies DG, Parsek MR, Pearson JP, Iglewski BH, Costerton JW, Greenberg EP. 1998. The involvement of cell-to-cell signals in the development of a bacterial biofilm. *Science* **280**:295–298.
16. De Kievit TR, Gillis R, Marx S, Brown C, Iglewski BH. 2001. Quorum-sensing genes in *Pseudomonas aeruginosa* biofilms: their role and expression patterns. *Appl Environ Microbiol* **67**:1865–1873.
17. Dockery J, Klapper I. 2001. Finger formation in biofilm layers. *SIAM J Appl Math* **62**(3):853–869.
18. Dolbow JE, Moës N, Belytschko T. 2000. Discontinuous enrichment in finite elements with a partition of unity method. *Finite Elem Anal Des* **36**:235–260.
19. Dolbow JE, Moës N, Belytschko T. 2001. An extended finite element method for modeling crack growth with frictional contact. *Comput Meth Appl Mech Eng* **190**:6825–6846.
20. Eberl HJ, Parker DF, van Loosdrecht MCM. 2001. A new deterministic spatiotemporal continuum model for biofilm development. *J Theor Med* **3**:161–175.
21. Eberl HJ, Picioreanu C, Heijnen JJ, van Loosdrecht MCM. 2000. A three-dimensional numerical study on the correlation of spatial structure, hydrodynamic conditions, and mass transfer and conversion in biofilms. *Chem Eng Sci* **55**:6209–6222.
22. Chessa J, et. al. 2002. The extended finite element method (xfem) for solidification problems. *Int J Num Meth Eng* **53**:1959–1977.
23. Fuqua C, Greenberg EP. 1995. Self perception in bacteria: quorum sensing with acylated homoserine lactones. *Curr Opin Microbiol* **118**(2):269–277.
24. Fuqua C, Parsek MR, Greenberg EP. 2001. Regulation of gene expression by cell-to-cell communication: acyl-homoserine lactone quorum sensing. *Ann Rev Genet* **35**:439–468.
25. Gaul L, Kögl M, Wagner M. 2003. *Boundary element methods for engineers and scientists*. New York: Springer.
26. Gravouil A, Moës N, Belytschko T. 2002. Non-planar 3d crack growth by the extended finite element and the level sets, II: level set update. *Int J Num Meth Eng* **53**(11):2569–2586.
27. Hermanowicz SW. 1999. Two-dimensional simulations of biofilm development: effect of external environmental conditions. *Water Sci Technol* **39**(7): 107–114.
28. Hermanowicz SW. 2001. A simple 2D biofilm model yields a variety of morphological features. *Math Biosci* **169**:1–14.
29. Indekeu JO, Giuraniuc CV. 2004. Cellular automaton for bacterial towers. *Phys A* **336**(1–2):14–26.
30. Ji H, Chopp D, Dolbow JE. 2002. A hybrid extended finite element/level set method for modeling phase transformations. *Int J Num Meth Eng* **54**:1209–1233.
31. Ji H, Chopp D, Dolbow JE. 2002. A hybrid extended finite element/level set method for modeling phase transformations. *Int J Num Meth Eng* **54**(8):1209–1233.
32. Kreft JU, Booth G, Wimpenny JWT. 1998. BacSim, a simulator for individual-based modeling of bacterial colony growth. *Microbiology* **144**:3275–3287.
33. Kreft JU, Picioreanu C, Wimpenny JWT, van Loosdrecht MCM. 2001. Individual-based modeling of biofilms. *Microbiology–SGM* **147**:2897–2912.
34. Laspidou CS, Rittmann BE. 2002. Non-steady state modeling of extracellular polymeric substances, soluble microbial products, and active and inert biomass. *Water Res* **36**:1983–1992.
35. Laspidou CS, Rittmann BE. 2002. Non-steady state modeling of microbial products and active and inert biomass. *Water Res* **36**:1983–1992.
36. LeVeque R, Li Z. 1994. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM J Num Anal* **31**:1019–1044.
37. Li Z. 2003. An overview of the immersed interface method and its applications. *Taiwan J Math* **7**:1–49.

38. Lide DR, ed. 1990. *CRC handbook of chemistry and physics*. Boca Raton, FL: CRC Press.
39. Mobarry BK, Wagner M, Urbain V, Rittmann BE, Stahl DA. 1996. Phylogenetic probes for analyzing abundance and spatial organization of nitrifying bacteria. *Appl Environ Microb* **62**(6):2156–2162.
40. Moës N, Dolbow J, Belytschko T. 1999. A finite element method for crack growth without remeshing. *Int J Num Meth Eng* **46**(1):131–150.
41. Moës N, Gravouil A, Belytschko T. 2002. Non-planar 3d crack growth by the extended finite element and the level sets, I: mechanical model. *Int J Num Meth Eng* **53**(11):2549–2568.
42. Osher S, Sethian JS. 1988. Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulation. *J Comput Phys* **79**:12–49.
43. Parsek MR. Unpublished data.
44. Pesci EC, Iglewski BH. 1997. The chain of command in *Pseudomonas* quorum sensing. *Trends Microbiol* **5**(4):132–135.
45. Pesci EC, Pearson JP, Seed PC, Iglewski BH. 1997. Regulation of *las* and *rhl* quorum sensing in *Pseudomonas aeruginosa*. *J Bacteriol* **179**(10):3127–3132.
46. Peskin CS. 1977. Numerical analysis of blood flow in the heart. *J Comput Phys* **25**:220–252.
47. Peskin CS. 1981. Lecures on mathematical aspects of physiology. *Lectures Appl Math* **19**:69–107.
48. Picioreanu C, van Loosdrecht MCM, Heijnen JJ. 1998. Mathematical modeling of biofilm structure with a hybrid differential-discrete cellular automaton approach. *Biotechnol Bioeng* **58**(1):101–116.
49. Picioreanu C, van Loosdrecht MCM, Heijnen JJ. 1999. Discrete-differential modeling of biofilm structure. *Water Sci Technol* **39**(7):115–122.
50. Picioreanu C, van Loosdrecht MCM, Heijnen JJ. 2000. A theoretical study on the effect of surface roughness on mass transport and transformation in biofilms. *Biotechnol Bioeng* **68**(4):355–369.
51. Piper KR, Beck von Bodman S, Farrand SK. 1993. Conjugation factor of *Agrobacterium tumefaciens* regulates Ti plasmid transfer by autoinduction. *Nature* **362**:448–450.
52. Pizarro G, Griffeath D, Noguera DR. 2001. Quantitative cellular automaton model for biofilms. *J Environ Eng* **127**(9):782–789.
53. Rittmann BE. 2002. Personal communication.
54. Rittmann BE, McCarty P. 2001. *Environmental Biotechnology*. New York: McGraw Hill.
55. Rosenfeld M, Ramsey B. 1992. Evolution of airway microbiology in the infant with cystic fibrosis: role of nonpseudomonal and pseudomonal pathogens. *Semin Respir Infect* **7**:158–167.
56. Schaefer AL, Hanzelka BL, Parsek MR, Greenberg EP. 2000. Detection, purification and structural elucidation of acylhomoserine lactone inducer of *Vibrio fischeri* luminescence and other related molecules. *Meth Enzymol* **305**:288–301.
57. Sethian JA. 1996. A marching level set method for monotonically advancing fronts. *Proc Natl Acad Sci USA* **93**(4):1591–1595.
58. Sethia JA. 1999. Fast marching methods. *SIAM Rev* **41**(2):199–235.
59. Stewart PS. 2003. Diffusion in biofilms. *J Bacteriol* **185**(5):1485–1491.
60. Stewart PS, Costerton JW. 2001. Antibiotic resistance of bacteria in biofilms. *Lancet* **358**(9276):135–138.
61. Stolarska M, Chopp DL. 2003. Modeling spiral cracking due to thermal cycling in integrated circuits. *Int J Eng Sci* **41**(20)2381–2410.
62. Stolarska M, Chopp DL, Möes N, Belytschko T. 2001. Modelling crack growth by level sets in the extended finite element method. *Int J Num Meth Eng* **51**:943–960.
63. Stolarska M, Chopp DL, Moës N, Belytschko T. 2001. Modelling crack growth by level sets in the extended finite element method. *Int J Num Meth Eng* **51**(8):943–960.
64. Sukumar N, Chopp DL, Moës N, Belytschko T. 2001. Modeling holes and inclusions by level sets in the extended finite element method. *Comput Meth Appl Mech Eng* **90**(46–47):6183–6200.
65. Sukumar N, Chopp DL, Moran B. 2003. Extended finite element method and fast marching method for three-dimensional fatigue crack propagation. *Eng Fracture Mech* **70**:29–48.

66. Sukumar N, Chopp DL, Moran B. 2003. Extended finite element method and fast marching method for three-dimensional fatigue crack propagation. *Eng Fracture Mech* **70**(1):29–48.
67. Szomolay B, Klapper I, Dockery J, Stewart PS. 2005. Adaptive response to antimicrobial agents in biofilms. *Environ Microbiol* **7**(8):1186–1191.
68. van Loosdrecht MCM, Heijnen JJ, Eberl HJ, Kreft JU, Picioreanu C. 2002. Mathematical modeling of biofilm structures. *Antonie van Leeuwenhoek* **81**:245–256.
69. Vaughan BL, Smith BG, Chopp DL. 2005. A comparison of the extended finite element method and the immersed interface method for elliptic equations with discontinuous coefficients and singular sources. Preprint available at `http://www.esam.northwestern.edu/chopp`.
70. Wagner GJ, Moës, N, Liu WK, Belytschko T. 2001. The extended finite element method for rigid particles in Stokes flow. *Int J Num Meth Eng* **51**:293–313.
71. Wanner O, Gujer W. 1986. A multispecies biofilm model. *Biotechnol Bioeng* **28**:314–328.
72. Williamson KJ, McCarty PL. 1976. Verification studies of the biofilm model for bacterial substrate utilization. *J Water Pol Control Fed* **48**:281–289.
73. Wimpenny JWT, Colasanti R. 1997. A unifying hypothesis for the structure of microbial biofilms based on cellular automaton models. *FEMS Microbiol Ecol* **22**(1):1–16.
74. Xavier JB, Picioreanu C, van Loosdrecht MCM. 2004. Assessment of three-dimensional biofilm models through direct comparison with confocal microscopy imaging. *Water Sci Technol* **49**(11–12):177–185.
75. Zik O, Moses E. 1999. Fingering instability in combustion: an extended view. *Phys. Rev. E* **60**(1):518–530.

2

DISTANCE TRANSFORM ALGORITHMS AND THEIR IMPLEMENTATION AND EVALUATION

George J. Grevera

Saint Joseph's University
Philadelphia, Pennsylvania, USA

Consider an n-dimensional binary image consisting of one or more objects. A value of 1 indicates a point within some object and a value of 0 indicates that that point is part of the background (i.e., is not part of any object). For every point in some object, a distance transform assigns a value indicating the distance from that point within the object to the nearest background point. Similarly for every point in the background, a distance transform assigns a value indicating the minimum distance from that background point to the nearest point in any object. By convention, positive values indicate points within some object and negative values indicate background points. A number of elegant and efficient distance transform algorithms have been proposed, with Danielsson being one of the earliest in 1980 and Borgefors in 1986 being a notable yet simple improvement. In 2004 Grevera proposed a further improvement of this family of distance transform algorithms that maintains their elegance but increases accuracy and extends them to n-dimensional space as well. In this paper, we describe this family of algorithms and compare and contrast them with other distance transform algorithms. We also present a novel framework for evaluating distance transform algorithms and discuss applications of distance transforms to other areas of image processing and analysis such as interpolation and skeletonization.

1. INTRODUCTION

Consider a binary image, I, consisting of one or more objects. Since this is a binary image, each point is either within the bounds of some object (interior point) or is part of the background and is not part of any object (exterior point).

George J. Grevera, BL 215, Mathematics and Computer Science, 5600 City Avenue, Saint Joseph's University, Philadelphia, PA 19131, USA. Phone: (610) 660-1535; Fax: (610) 660-3082. ggrevera@sju.edu.

We note that some points within objects are noteworthy in that they are positioned on the border of the object with at least one of the outside (background) points. Adopting terminology from digital topology [1], we call these points elements of the set of points that form the immediate interior (II) of some object. Similarly, some background points are notable in that they are positioned on the border or interface of some object as well. Once again adopting terminology from digital topology, we call these background points elements of the set of points that form the immediate exterior (IE). Together, the union of the sets II and IE form a set of points called border points (or boundary elements), B. We may now define a distance transform as an algorithm that given I produces a transformed image, I', by assigning to each point in I the minimum distance from that point to all border points.

A number of issues arise when dealing with distance transform algorithms. The first and probably the most important issue is one of accuracy. Does the distance transform produce results with minimal errors (or is the distance transform error free)? If that is the case, it is important to develop a methodology to verify this claim (and we will do so in this chapter). Even for those algorithms that are theoretically proven to be error free, from a software engineering standpoint it is important to be able to validate the implementation of the algorithm. A second important issue concerns the computation time required by the method. It is relatively straightforward to develop an exhaustive method that requires a great deal of processing time. It is important to evaluate processing time as well. And yet another issue is with regard to the dimensionality of I. Since medical imagery is of three or four dimensions, it is important in the medical arena for a distance transform algorithm to generalize to dimensions higher than two. Another issue that arises when dealing with medical images is that of anisotropic sampling. Medical images are typically acquired as three-dimensional volumes of data (stacks of slices) with the sampling within each plane or slice at a higher rate than the sampling across slices. This yields data with finer spacing between neighboring pixels (or more generally, voxels) within a slice (e.g., 0.5 mm) than between neighboring pixels between slices (e.g., 1.0 mm).

Distance transform algorithms are, like most other computationally intensive algorithms, of interest in and by themselves and have been the subject of at least one PhD dissertation [2]. Many distance transform algorithms have been proposed, with [3] and [4] most likely being the earliest. In general, distance transform algorithms exhibit varying degrees of accuracy of the result, computational complexity, hardware requirements (such as parallel processors), and conceptual complexity of the algorithms themselves. In [5], the author proposed an algorithm that produces extremely accurate results by propagating vectors that approximate the distance in 2D images by sweeping through the data a number of times by propagating a local mask in a manner similar to convolution. In [6] the author presented the Chamfer distance algorithm (CDA), which propagates scalar, integer values to efficiently and accurately calculate the distance transform of 2D and 3D images (again in a

manner similar to convolution). Borgefors [6] also presented an error analysis for the CDA for various neighborhood sizes and integer values. More recently in [7] an analysis of 3D distance transforms employing 3x3x3 neighborhoods of local distances was presented. In [8] an analysis of the 2D Chamfer distance algorithm using 3x3, 5x5, and larger neighborhoods employing both integer and real values was presented. Marchand-Maillet and Sharaiha [9] also present an analysis of Chamfer distance using topological order as opposed to the approximation to the Euclidean distance as the evaluation criteria. Because of the conceptual elegance of the CDA and because of its widespread popularity, we feel that the CDA family of algorithms is important and worthy of further study.

Of course, distance transforms outside of the Chamfer family also have been presented. A technique from Artificial Intelligence, namely $A*$ heuristic search [10], has been used as the basis for a distance transform algorithm [11]. A multiple-pass algorithm using windows of various configurations (along the lines of [5] and other raster scanning algorithms such as the CDA) was presented in [12] and [13]. A method of distance assignment called ordered propagation was presented in [14]. The basis of that algorithm and others such as $A*$ (used in [11]) is to propagate distance between pixels, which can be represented as nodes in a graph. These algorithms typically employ sorted lists to order the propagation among the graph nodes. Guan and Ma [15] and Eggers [16] employ lists as well. In [17] the authors present four algorithms to perform the exact, Euclidean, n-dimensional distance transform via the serial composition of n-dimensional filters. Algorithms for the efficient computation of distance transforms using parallel architectures are presented in [18] and [19]. In [19] the authors present an algorithm that consists of two phases, with each phase consisting of both a forward scan and a backward scan. In the first phase columns are scanned; in the second phase rows are scanned. They note that since the scanning of a particular column (or row) is independent of the scanning of the other columns (or rows), each column (row) may be scanned independently (i.e., in parallel). A distance transform employing a graph search algorithm is also presented in [20].

Since the early formulation of distance transform algorithms [3, 4], applications employing distance transforms have also become widespread. For example, distance transforms have been used for skeletonization of images [21, 22, 23, 24]. Distance transforms are also useful for the (shape-based) interpolation of both binary images [25, 26] as well as gray image data [27]. In [28] the authors employ distance transform information in multidimensional image registration. An efficient ray tracing algorithm also employs distance transform information [29]. Distance transforms have also been shown to be useful in calculating the medial axis transform, with [30, 31] employing the Chamfer distance algorithm specifically. In addition to the usefulness of distance transforms for the interpolation of 3D gray medical image data [32, 33], they have also been used for the automatic classification of plant cells [34] and for measuring cell walls [35]. The Chamfer distance was also employed in a method to characterize spinal cord atrophy [36].

Because distance transforms are applicable to such a wide variety of problems, it is important to develop accurate and efficient distance transform algorithms.

2. DISTANCE TRANSFORM ALGORITHMS

Before we begin our discussion of algorithms in detail, we note that our descriptions will be limited to the two-dimensional case (i.e., where the input image, I, is two dimensional). Some algorithms readily generalize (or have been generalized) to higher dimensions. We will endeavor to point out those more general algorithms. Furthermore, all of the algorithms that will be described do not require any special purpose hardware such as parallel processors.

All of the distance transform algorithms that will be described can be said to rely on the determination of border points, so we begin with a method to determine them. Recall that we define the border points, B, as the union of the sets *II* and *IE*. To determine the set of border points, we must first determine these sets. A point $p = (x, y)$ is an element of an object iff $I(p)$ = 1. Similarly, a point $q = (x, y)$ is an element of the background iff $I(q) = 0$. But not all object points are elements of *II* (nor are all background points elements of *IE*). Only those object points that are on the border of an object are elements of *II* (and similarly for *IE*). To determine if an object point, $p = (x, y)$, is an element of *II*, we consider the neighborhood of p to be the set of all points $N(p) = \{(x + dx, y + dy) \mid -1 <= dx <= 1$ and $-1 <= dy <= 1\}$. In practice, we typically restrict the definition of $N(p)$ to include only those nearby elements with the same x or y coordinates as p (the so-called 4-adjacency (connectedness or connectivity) versus the less restrictive 8-adjacency) as follows: $N(p) = \{(x + dx, y + dy)| - 1 <= dx <= 1$ and $-1 <= dy <= 1$ and $|dx + dy| = 1\}$. If there exists at least one point q in $N(p)$ such that q is an element of the background, then p is an element of *II*. Similarly, to determine if a background point, $q = (x, y)$, is an element of *IE*, we consider the neighborhood of q. If there exists at least one point p in $N(q)$ such that p is an element of an object, then q is an element of the *IE*. The algorithm for this determination follows:

```
for (y=1; y<ySize-1; y++)
   for (x=1; x<xSize-1; x++)
        if ( I(x-1,y) != I(x,y) or I(x+1,y)] != I(x,y) or
             I(x,y-1) != I(x,y) or I(x,y+1)  != I(x,y))
            then (x,y) is a 4-adjacent border element.
        if ( I(x-1,y-1) != I(x,y) or I(x+1,y-1)] !=I(x,y) or
             I(x-1,y+1) != I(x,y) or I(x+1,y+1)  !=I(x,y) )
           then (x,y) is a remaining 8-adjacent border element.
```

where xSize is the number of columns in I and I', and ySize is the number of rows. We note that some distance transform algorithms including [5] restrict the definition of border points to elements of *II* only. Our framework easily accommodates this

via a simple change to the algorithm above as illustrated below. We point out, however, that this definition will not preserve the property of symmetry under complement [37]. Consider the complement C of the binary image I such that $C(p) = 1$ if $I(p) = 0$ and $C(p) = 0$ otherwise. A distance transform that preserves symmetry under complement produces the same result given either C or I (i.e., $C'(p) = I'(p)$ for all p, although the sign may be opposite by convention. In that case, $|C'(p)| = |I'(p)|$.

```
for (y=1; y<ySize-1; y++)
    for (x=1; x<xSize-1; x++)
        if (I(x,y)==1)    //restrict border points to II only
            if ( I(x-1,y) != I(x,y) or I(x+1,y)] !=I(x,y) or
                 I(x,y-1) != I(x,y) or I(x,y+1)  !=I(x,y) )
                then (x,y) is a 4-adjacent border element.
            if ( I(x-1,y-1) != I(x,y) or I(x+1,y-1)] != I(x,y) or
                 I(x-1,y+1) != I(x,y) or I(x+1,y+1) != I(x,y) )
                then (x,y) is a remaining 8-adjacent border element.
```

Note that, without loss of generality, we assume that no object extends to the edge of the discrete matrix in which it is represented. Otherwise, the description of the algorithms would be unnecessarily complicated by additional boundary condition checks. If it is the case that an object extends to the edge of the matrix, one may simply embed that matrix and the objects that are represented within it in a larger matrix with an additional layer of surrounding background elements.

2.1. A Simple Distance Transform Algorithm (Simple)

Arguably the simplest distance transform follows. First, we assign each border element a distance value of 0: $I'(s) = 0$, where s is in B. Then for each t not in B, we assign $I'(t) = \min\{d(s,t) | s \text{ in } B \text{ and } t \text{ not in } B\}$, where $d(s,t)$ is the Euclidean distance from s to t. This algorithm is very simple, both conceptually and computationally. It is also error free. Furthermore, it is also very easy to extend this algorithm to higher dimensions as well as to anisotropic data. Unfortunately, if for each t we must search I to determine every s in B, we have an algorithm that is the least computationally efficient of those that will be discussed. We subsequently refer to this algorithm as Simple. Pseudo code for this algorithm follows.

```
//iterate over all (non border element) points
 for (y=1; y<ySize-1; y++) {
     for (x=1; x<xSize-1; x++) {
         if (I'(x,y)!=0) {  //only consider non border elements
             //t=(x,y)
             //now iterate over all border elements
             for (y1=0; y1<ySize; y1++) {
                 for (x1=0; x1<xSize; x1++) {
```

```
                    if (I'(x1,y1)==0) {
                        //s=(x',y')
                //calculate the distance to this border element
                        d = sqrt( (x-x1)*(x-x1) + (y-y1)*(y-y1) );
                //is it better than what's already been assigned?
                        if (d < I'(x,y)) {
                //yes, then update the distance from this
                //point, t, to the border element, s
                            I'(x,y) = d;
                        }
                    }  //end if
                  }  //end for x1
              }  //end for y1
          }  //end if
      }  //end for x
  }  //end for y
```

2.2. A Simple Distance Transform Algorithm Employing a List (SimpleList)

A straightforward modification to the simple algorithm yields a surprisingly effective method. Instead of exhaustively and repeatedly searching I to determine every s in B for every t not in B, we employ an additional data structure, a list (actually using the vector class which can be indexed as implemented in the generics in the standard C++ library) to represent B. The C++ vector class is used because it may be indexed like an array, but unlike an array it can grow in size dynamically. One does not need to know a priori the number of points to allocate for the array. Then for each t not in B, we search the list $L = B$ and assign $I'(t) = \min\{d(s,t)|s \text{ in } L \text{ and } t \text{ not in } B\}$. Like the simple algorithm that does not employ a list, this algorithm is also very simple, both conceptually and computationally. It too is also error free. Furthermore, it is also very easy to extend this algorithm to higher dimensions as well as to anisotropic data. Unlike the simple algorithm that does not employ a list, this algorithm is very efficient when the size of the list is small. We subsequently refer to this algorithm as SimpleList. Pseudo code for this algorithm follows.

```
//iterate over all (non border element) points
for (y=1; y<ySize-1; y++) {
    for (x=1; x<xSize-1; x++) {
        if (I'(x,y)!=0) {  //only consider non border elements
        //at this stage, we have a point that is not an element of
        //the border. iterate over all border elements in the list.
            for (i=0; i<list.size(); i++) {
                x1 = list[i]->x;
```

```
                y1 = list[i]->y;
                //calculate the distance to this border element
                d = sqrt( (x-x1)*(x-x1) + (y-y1)*(y-y1));
                //is it better than what's already been assigned?
                if (d < I'(x,y)) {
                    //yes, then change to this border element
                    I'(x,y) = d;
                }
            }
        }  //end if
    }  //end for x
}  //end for y
```

This method is also important for testing other methods as it will form the basis of our testing procedure.

2.3. Danielsson's [5] 4SED and 8SED Algorithms (and Grevera's Improved 8SED Algorithm)

Although not completely error free, Danielsson's distance transform algorithms were early contributions and are important steps in the development of subsequent algorithms such as the Chamfer distance and Dead Reckoning. 8SED is efficient, reasonably accurate, conceptually easy to understand, and is still in widespread use today. These algorithms begin by initially assigning a distance value of 0 for all p in B and a value of infinity for all p not in B. Then the algorithms sweep through I' using a number of passes and various local "windows" in a manner somewhat similar to convolution [38] from digital signal processing. The current distance assignment to each point under consideration, u, is compared to the current assignments to its neighbors plus the distance, a, from the specific neighbor, n, to u. If the current distance assignment, $I'(u)$, is greater than $I'(n) + a$, then $I'(u)$ is updated to $I'(n) + a$, which results in minimizing the distance to u. The difference between 4SED, 8SED, and Grevera's improved 8SED algorithms are in the number of sweeps and in the neighborhood windows that are checked during the minimization process. 4SED is the least accurate but is the fastest. Grevera's improved 8SED produces more accurate results at the expense of increased processing time although the increase in time is not significant. Pseudo code for the 4SED algorithm follows.

```
//perform the first pass ("first picture scan")
for (y=1; y<=ySize-1; y++) {
    for (x=0; x<=xSize-1; x++)  check( x,   y-1, dy  );
    for (x=1; x<=xSize-1; x++)  check( x-1, y,   dx  );
    for (x=xSize-2; x>=0; x--)  check( x+1, y,   dx  );
}
```

```
//perform the final pass ("second picture scan")
for (y=ySize-2; y>=0; y--) {
    for (x=0; x<=xSize-1; x++)  check( x,   y+1, dy  );
    for (x=1; x<=xSize-1; x++)  check( x-1, y,   dx  );
    for (x=xSize-2; x>=0; x--)  check( x+1, y,   dx  );
}
```

where *check* compares, and updates if necessary, the current distance $I'(u)$, $u = (x, y)$ to the specified neighbor and offset from neighbor to u, $I'(n) + a$. Pseudo code for the 8SED algorithm follows.

```
//perform the first pass ("first picture scan")
for (y=1; y<=ySize-1; y++) {
    for (x=0; x<=xSize-1; x++) {
        if (x>0) {  //** boundary condition not checked in original
                    //   but needed
            check( x-1, y-1, dxy );
        }
        check( x,   y-1, dy  );
        if (x<xSize-1) {  //** not checked in original but needed
            check( x+1, y-1, dxy );
        }
    }
    for (x=1; x<=xSize-1; x++)  check( x-1, y,   dx  );
    for (x=xSize-2; x>=0; x--)  check( x+1, y,   dx  );
}

//perform the final pass ("second picture scan")
for (y=ySize-2; y>=0; y--) {
    for (x=0; x<=xSize-1; x++) {
        if (x>0) {  //** not checked in original but needed
            check( x-1, y+1, dxy );
        }
        check( x,   y+1, dy  );
        if (x<xSize-1) {  //** not checked in original but needed
            check( x+1, y+1, dxy );
        }
    }
    for (x=1; x<=xSize-1; x++)  check( x-1, y, dx );
    for (x=xSize-2; x>=0; x--)  check( x+1, y, dx );
}
```

Pseudo code for Grevera's improved 8SED algorithm follows. Note: * indicates a difference from the original 8SED algorithm.

```
//perform the first pass ("first picture scan").
for (y=1; y<ySize-1; y++) {
    for (x=0; x<=xSize-1; x++) {  //* from 4sed
        check( x,   y-1, dy  );
    }
    for (x=1; x<=xSize-1; x++) {
        check( x-1, y,   dx  );
        check( x-1, y-1, dxy );   //*
    }
    for (x=xSize-2; x>=0; x--) {
        check( x+1, y,   dx  );
        check( x+1, y-1, dxy );   //*
    }
}

//perform the final pass ("second picture scan")
for (y=ySize-2; y>=0; y--) {
for (x=0; x<=xSize-1; x++) {  //* from 4sed
        check( x,   y+1, dy  );
    }
    for (x=1; x<=xSize-1; x++) {
        check( x-1, y,   dx  );
        check( x-1, y+1, dxy );   //*
    }
    for (x=xSize-2; x>=0; x--) {
        check( x+1, y,   dx  );
        check( x+1, y+1, dxy );   //*
    }
}
```

2.4. Borgefors' [6] CDA (Including Chessboard, Cityblock, and Euclidean 3x3 Window)

Borgefors' Chamfer distance algorithm (CDA) is arguably the most popular distance transform method. Like Danielsson's algorithms, it is not completely error free, but it is efficient, reasonably accurate, and conceptually even easier to understand than Danielsson's algorithms. Furthermore, it has also been extended from 2D to 3D [7, 40], and simple modifications produce other distance transforms such as chessboard, cityblock, and Euclidean with a 3x3 window. Like Danielsson's algorithm, the CDA (including chessboard, cityblock, and Euclidean) begins by initially assigning a distance value of 0 for all p in B and a value of infinity for all q not in B. Then the CDA sweeps through using two passes. The first pass is from top to bottom and left to right and the second pass is from bottom to top and right to left. Again, various local "windows" are used in a manner similar to convolution

[38] from digital signal processing using 3x3 windows for CDA 3x3, chessboard, cityblock, and Euclidean, 5x5 windows for CDA 5x5, and 7x7 windows for CDA 7x7. The current distance assignment to each point under consideration, u, is compared to the current assignments to its neighbors plus the distance, a_n, for that specific neighbor n taken from Figure 1 from the specific neighbor, n, to u. If the current distance assignment, $I'(u)$, is greater than $I'(n)+a_n$, then $I'(u)$ is updated to $I'(n) + a_n$, which results in minimizing the distance to u. Resulting distance transform errors diminish with increasing window size while computation cost increases with increasing window size. Note that different window configurations (but of the same size) are employed for the forward and backward passes.

Pseudo code for CDA 3x3, cityblock, chessboard, and Euclidean 3x3 appears below. *dx*, *dy*, and *dxy* are assigned a_n values according to the table entries in Figure 1 corresponding to the desired method. CDA 5x5 and CDA 7x7 are analogous.

```
//perform the first (forward) pass
for (y=1; y<ySize-1; y++) {
    for (x=1; x<xSize-1; x++) {
        check( x-1, y-1, dxy );
        check( x,   y-1, dy  );
        check( x+1, y-1, dxy );
        check( x-1, y,   dx  );
    }
}
//perform the final (backward) pass
for (y=ySize-2; y>=1; y--) {
    for (x=xSize-2; x>=1; x--) {
        check( x+1, y,   dx  );
        check( x-1, y+1, dxy );
        check( x,   y+1, dy  );
        check( x+1, y+1, dxy );
    }
}
```

where *check* compares, and updates if necessary, the current distance $I'(u)$, $u = (x, y)$ to the specified neighbor and offset from neighbor to u, $I'(n) + a_n$.

Borgefors cleverly demonstrated: (i) using a small window and propagating distance in this manner introduces errors in the assigned distance values even if double precision floating point is used to represent distance values, (ii) these errors may be minimized by using values other than 1 and $\sqrt{2}$ for the distances between neighboring pixels, and, surprisingly, (iii) using integer window values such as 3 and 4 yields more accurate results than using window values of 1 and $\sqrt{2}$ and does

	forward pass	backward pass
CDA 3×3	4 3 4 3 u - - - -	- - - - u 3 4 3 4
city block	- 1 - 1 u - - - -	- - - - u 1 - 1 -
chessboard	1 1 1 1 u - - - -	- - - - u 1 1 1 1
CDA 5x5	- 11 - 11 - 11 7 5 7 11 - 5 u - - - - - - - - - - - -	- - - - - - - - - - - - u 5 - 11 7 5 7 11 - 11 - 11 -
CDA 7×7	- 43 38 - 38 43 - 43 - 27 - 27 - 43 38 27 17 12 17 27 38 - - 12 u -	- u 12 - - 38 27 17 12 17 27 38 43 - 27 - 27 - 43 - 43 38 - 38 43 -
Euclidean 3x3	$\sqrt{2}$ 1 $\sqrt{2}$ 1 u - - - -	- - - - u 1 $\sqrt{2}$ 1 $\sqrt{2}$

Figure 1. Various windows used by the Chamfer distance algorithm. 'u' indicates the center of the window. '-' indicates that the point is not used (considered) during that pass of the algorithm.

so with much better performance (when implemented using integer arithmetic), and (iv) larger windows with appropriate values minimize errors even further at increased computational cost.

2.5. Grevera's [37] Dead Reckoning Algorithm

The Dead Reckoning Algorithm (DRA) is a straightforward modification to the CDA that, employing equal-sized windows, produces more accurate results at a slightly increased computational cost. Furthermore, it has been demonstrated [37] that DRA using only a 3x3 window typically produces more accurate results than CDA with a 7x7 window with similar execution times.

In addition to I', which for a given point, (x, y), is the minimum distance from (x, y) to the nearest border point, the DRA introduces an additional data structure, $P(x, y) = (x', y')$, which is used to indicate the actual border point (x', y') in B such that $I'(x, y)$ is minimum. This is similar to the method employed by Danielsson [5], where 4SED employs three minimization iterations in both the forward and backward passes. Our method as in the CDA employs only one iteration in each pass. Note that as the CDA progresses, $I'(x, y)$ may be updated many times. In the DRA, each time that $I'(x, y)$ is updated, $P(x, y)$ is updated as well. We note that the order in which the 'if' statements in the pseudo code for this algorithm are evaluated may influence the assignment of $P(x, y)$ and subsequently, the value assigned to $I'(x, y)$. Regardless, our results demonstrate that our algorithm remains more accurate using only a 3x3 neighborhood than CDA using a 7x7 neighborhood. Although the DRA employs a 3x3 (or larger) window to guide the update/minimization of distance process as does CDA, the actual values assigned to I' are not the same as CDA. DRA uses instead the actual Euclidean distance from the border to the point (x, y) at the center of the window. Using only a 3x3 window, the DRA typically determines a more accurate estimation of the exact Euclidean distance within the framework of the CDA. Pseudo code for the DRA is the same as CDA except for a modification to *check*. *check* compares, and updates if necessary, the current distance $I'(u)$, $u = (x, y)$ to the specified neighbor and offset from neighbor to u, $I'(n) + a_n$ as before in the CDA but if $I'(u) >= I'(n) + a_n$, $I'(u)$ is not assigned $I'(n) + a_n$ but is assigned distance from u to $P(n)$ and $\underline{P}(u) = P(n)$.

2.6. Dijkstra's Graph Algorithm

Dijkstra's shortest path algorithm [39] for determining minimum cost paths in graphs can be adapted to distance transform algorithms as well. To accomplish this we simply map the input binary image, I, to a graph, $G = (V, E)$, where V is the set of vertices (the set of discrete (x, y) locations in I and E is the set of edges defined as follows. Consider some point, $p = (x, y)$, in V and the (8-connected) neighborhood $N(p) = \{(x-1, y), (x+1, y), (x, y-1), (x, y+1), (x-1, y-1), (x+1, y-1), (x-1, y+1), (x+1, y+1)\} = \{(x+dx, y+dy) | -1 <= dx <= 1$

and $-1 <= dy <= 1$ and $|dx + dy| <= 2\}$. E consists of all edges from points p to each of its neighbors and define the cost (distance) associated as either 1 or $\sqrt{2}$ depending on the Euclidean distance from p to the particular neighbor. As in previous algorithms, this method begins as many of the previous ones with initially assigning a distance value of 0 for all p in B and a value of infinity for all p not in B. Those points p for which $I'(p) = 0$ are also initially placed on an ordered list L that is sorted from smallest to largest according to the distance assignment, $I'(p)$. The algorithm then proceeds as follows:

```
while (L is not empty) {
    remove from L, p such that I'(p) is minimal
      among all elements of L;
    consider each neighbor n in N(p);
    if (I'(p)+d(p,n) < I'(n)) {
        I'(n) = I'(p)+d(p,n);
        put n in L according to I'(n);
    }
}
```

where $d(p, n)$ is the Euclidean distance from p to n. Because of the regularly discretized nature of I, $d(p, n)$ is either 1 or $\sqrt{2}$ since n is in $N(p)$. When this algorithm terminates, $I'(p)$ will contain the minimal distance from p to an element of B in terms of the minimal summation of edge costs associated with a path from any element of B to p. In that respect, this method is similar to DRA. Subsequently, we will refer to Dijkstra's algorithm adapted and applied to the distance transform problem as ModifiedDijkstra (MD).

We will now describe a variant of the MD algorithm. Recall that DRA is the same as CDA except for a modification to the *check* procedure where *check* compares the current distance assignment $I'(u)$, $u = (x, y)$ to the specified neighbor and offset from neighbor to u, $I'(n) + a_n$ as before, but if $I'(u) >= I'(n) + a_n$, $I'(u)$ is not assigned $I'(n) + a_n$ but is assigned the distance from u to $P(n)$. We can also modify MD to perform in this manner. We call this method ModifiedDijkstraDeadReckoning (MDDR) using 8-connected neighborhoods.

None of these algorithms (based on Dijkstra's graph algorithm) are error free. To develop an error-free graph-based algorithm, we note that Dijkstra's algorithm determines cost (distance) as a discrete sequence of edges in the graph between two vertices. Since we are using this discrete space as a model of an underlying continuous space, small errors may be introduced. Therefore, the first time that a vertex is encountered may not be the optimal distance assignment for that point. We must allow for a vertex to be revisited (as in $A*$ heuristic search [10]) by maintaining a list (vector) of border element assignments (instead of a single, first assignment). We call this method DijkstraVectors (DV), and experimental results have shown this algorithm to be error free.

2.7. Ragnemalm's [14] CSED Algorithm

Ragnemalm's CSED algorithm is similar to Dijkstra's algorithm except that instead of employing a single ordered list, L, it avoids maintaining L in sorted order by using two lists, $L1$ and $L2$, and a limit or threshold, l , on the current p in $L1$ such that $I'(p) < l$. In this manner, propagation of distance values is ordered along approximate isocontours by repeatedly sweeping through $L1$ by examining each p in $L1$ in turn. If the current $I'(p) < l$, then we consider the neighborhood of p. Otherwise, we move p from $L1$ to $L2$ for future consideration. Initially, all elements of $N(b)$, b in B, are placed on $L1$. Similar to the DRA, which introduced an additional data structure, $P(x, y) = (x', y')$, which is used to indicate the actual border point (x', y') in B such that $I'(x, y)$ is minimum, CSED uses an additional data structure $\underline{P}(x, y) = (d\underline{x}, d\underline{y})$ such that $(x', y') = (x, y) + (d\underline{x}, d\underline{y})$. ($P(x, y)$ is an element of B where $\underline{P}(x, y)$ is the displacement from (x, y) to an element of B). Additionally, we will refer to the x component of p in $\underline{P}$ as $\underline{p}x$ and similarly for y. Ragnemalm's algorithm also performs a more intelligent propagation among the 8-connected neighbors than Dijkstra's algorithm as well.

Pseudo code for this algorithm follows. Like the algorithms based on Dijkstra's algorithm (except for DV), CSED is not error free.

```
for all p {
    if I'(p)=0 then
        consider each neighbor n in N(p)
        check( p,  0,  1 );
        check( p,  1,  1 );
        check( p,  1,  0 );
        check( p,  1, -1 );
        check( p,  0, -1 );
        check( p, -1, -1 );
        check( p, -1,  0 );
        check( p, -1,  1 );
}
swap( l1, l2 );
l = 1;
while L1 is not empty
    for each p in L1
        remove p from L1
        if I'(p) > l then
            put p on L2;
        else if px(p)=0 then
            check( p, 0,sgn(py(p)) );
                                          //vertical
        else if py(p)=0 then
            check( p, sgn(px(p)), 0 );
```

```
                                        //horizontal
        else if |px(p)| = |py(p)| then
            check( p, sgn(px(p)), sgn(py(p)) );
                                        //diagonal
            check( p, sgn(px(p)), 0          );
                                        //horizontal
            check( p, 0,          sgn(py(p)) );
                                        //vertical
        else if |Px(p)| > |Py(p)| then
            check( p, sgn(px(p)), sgn(py(p)) );
                                        //diagonal
            check( p, sgn(px(p)), 0          );
                                        //horizontal
        else
            check( p, sgn(px(p)), sgn(py(p)) );
                                        //diagonal
            check( p, 0,          sgn(py(p)) );
                                        //vertical
    L1 = L2;
```

where $sgn(k) = -1$ if $k < 0$, 0 if $k = 0$, and 1 if $k > 0$ and *check* is defined as follows:

```
check( p, dx, dy )
    let n = (px+dx, py+dy);
    let d = sqrt( dx*dx + dy*dy );
    if (I'(n) > I'(p)+d) {
        I'(n) = I'(p)+d;
        put n in L2;
    }
```

Ragnemalm also presents an error free version of the CSED algorithm. Unfortunately, our implementation of that algorithm, which we believe is faithful to the description in their paper, allows a few points to remain initialized at infinity in our tests. This severely skews the results. Therefore, the software that accompanies this article includes our implementation of the error free version but the results of executing that implementation will not be included in this paper.

3. EVALUATING DISTANCE TRANSFORM ALGORITHMS

Distance transforms may be evaluated according to a variety of criteria. As mentioned previously, the accuracy of the result is arguably the most important measure. Even algorithms that purport to be error free should be evaluated to ensure that the implementation is indeed error free. To evaluate accuracy we

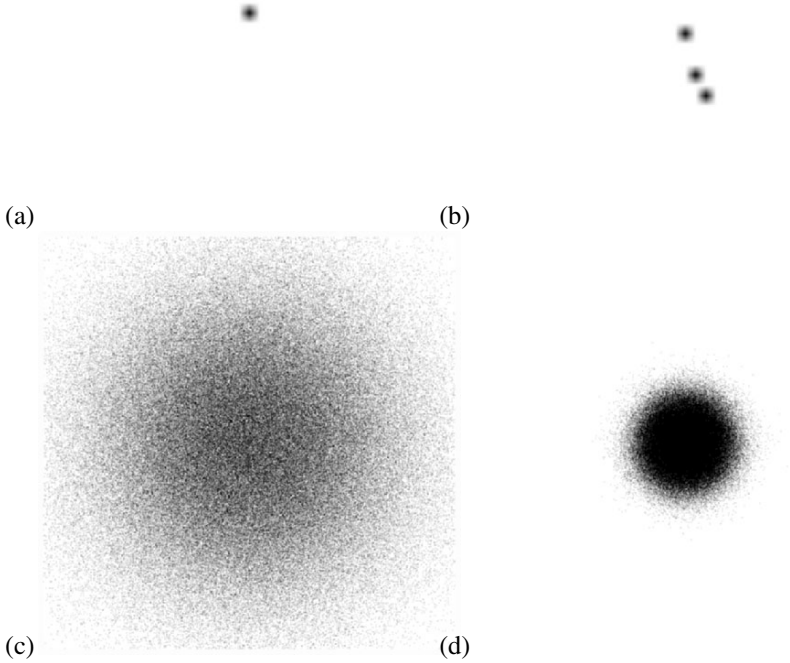


Figure 2. Sample test images consisting of (a) a single, solitary point-object, (b) a configuration of three single point-objects that is a known problematic configuration, (c) and (d) randomly generated test images by sampling from a normal distribution (with different standard deviations).

need to (i) choose a suite of test cases (input binary images), (ii) develop a "gold standard" or "ground truth" for each of the test cases, (iii) choose a set of metrics to compare the result of a distance transform method with ground truth, and then (iv) compare the results of a method under test with the gold standard using the metrics.

The simplest test case consists of an image that contains a solitary object consisting of a single point at the center of the image as shown in Figure 2a. Another test case has been described [2] as being extremely problematic for algorithms that sweep through the data using local windows (such as 4SED, 8SED, CDA,

DRA, and others). It consists of the three single point-objects as shown in Figure 2b. Although input images consisting of a few solitary point-objects are useful for understanding algorithms, they are not reflective of real-world objects. To simulate real-world objects, we also include images consisting of randomly generated objects by sampling from a normal distribution, as shown in Figures 2c and 2d.

With regard to a gold standard, we chose the SimpleList algorithm because it is straightforward, easy to verify, and is exhaustive in its determination of the correct distance assignments. The Simple algorithm could be used instead of SimpleList but in practice, Simple is too slow to be useful. (For example, for a rather small image of 300x300 consisting of a single center point object, SimpleList required 0.03 s of CPU time on a 2-GHz Pentium 4 under Linux. Simple required 71.98 s. The remaining methods required less then 1 s.)

The result of the distance transform, I', may be regarded as a grey image where the grey value at each location is the distance value assigned by the particular algorithm. Given I', the result of some distance transform algorithm, and $I'_{\mathrm{SimpleList}}$ (the result of applying SimpleList to I), we can compute the magnitude of the differences between I' and $I'_{\mathrm{SimpleList}}$ and determine the RMS (root mean squared) error as well as the location of the (magnitude of the) single largest difference between I' and $I'_{\mathrm{SimpleList}}$. Additionally, we also calculate the number of pixels that exhibit any difference whatsoever (regardless of the magnitude of the difference) and express this as a percentage of the whole. More qualitative insights can also be gained by viewing difference images ($|I' - I'_{\mathrm{SimpleList}}|$) or by simply thresholding I' to create a binary image and viewing the result as shown in Figure 7 as applied to the input binary image consisting of a single center point. The expected thresholded result should appear as a circular isocontour with radius equal to the distance from the center point. Which isocontour is observed depends upon the selected threshold value. Note that the thresholded results of CDA 3x3, CDA 5x5, Chessboard, Cityblock, Euclidean 3x3, and MD exhibit significant visible errors in the form of polygonal approximations to the circular isocontour. The more accurate of these methods exhibit polygons with more sides (while the least have less sides). Chessboard has only four sides, while the thresholded results of CDA 3x3, Cityblock, Euclidean 3x3, and MD have eight sides. Careful examination of the thresholded results of CDA 5x5 and CDA 7x7 yields 16- and 20-sided polygons for the selected threshold, respectively. The remaining, most accurate methods do not have any noticeable artifacts. In addition to accuracy, it is also important to report the CPU time required to perform the distance transform as well.

4. RESULTS OF EVALUATION

All experiments were performed on a Dell 3.6-GHz Pentium 4 system with 2 GB of RAM running Redhat Linux version 2.6.9 and using g++ version 3.4.2.

The times reported are user mode CPU time plus kernel mode CPU time. We feel that this is a better measure than simple elapsed time, especially on modern, multiprogrammed operating systems. No other users were logged onto the system during the tests. Four input test images were employed to evaluate the various distance transform algorithms: (1) a solitary object consisting of a single solitary point at the center of the image (Figure 2a), (2) the extremely problematic image consisting of 3 point-objects (Figure 2b), (3) a randomly generated set of objects created by sampling from a normal distribution with a mean of the center of the image an a standard deviation of 0.20 (Figures 2c), and (4) another randomly generated set of objects created by sampling from a normal distribution with a different standard deviation of 0.05 (Figure 2d). Each of the input test images were 1000x1000 pixels in size. As previously mentioned, the SimpleList algorithm was used as the gold standard. RMS error as well as the magnitude of the single largest difference are reported as well. The results of the evaluation are shown in Table 1 for the central single-point object and three single-point objects, and 2 for two sets of randomly generated objects.

5. CONCLUDING REMARKS

Although the results in Table 1 appear promising for the gold standard method, SimpleList, with regards to CPU time, Table 2 demonstrates that SimpleList is not practical for most applications because its time is two to three orders of magnitude worse than other methods. The best-performing methods with regard to CPU time took as little as 0.1 seconds. Of these fastest methods, DRA 3x3 exhibited minimal error for the randomly generated images.

With regard to accuracy, DV and SimpleList were the only methods that exhibited 0 errors. The performance of SimpleList precludes it from being used in practice but the performance of DV is quite good for practical use. For applications that can tolerate small errors, the modified 8SED algorithm had a very low error rate and excellent performance.

6. ACKNOWLEDGMENTS

The authors gratefully acknowledge NIH Grant R01-EB004395-01 for support of this work.

Table 1. Results of various distance transform algorithms applied to an image consisting of a solitary object consisting of a single point at the center of the image (left) and 3 single-point objects (right). RMSE is the root mean squared error, max err is the value of the magnitude of the largest difference, diff is the percentage of the total number of points that are different, and time is the CPU time in seconds

	Central single-point object				3 single-point objects			
	RMSE	max err	diff	time	RMSE	max err	diff	time
CDA 3×3	14.7	39.6	95.7	0.1	14.7	39.6	95.8	0.1
CDA 5×5	3.9	10.0	95.7	0.1	3.9	9.9	95.7	0.1
CDA 7×7	2.5	6.8	95.7	0.3	2.4	6.9	95.7	0.3
Chessboard	67.5	202.6	95.8	0.1	67.4	203.0	95.8	0.1
Cityblock	135.0	286.7	95.8	0.1	134.2	287.3	95.8	0.1
CSED	0.0	0.0	0.0	0.2	4E-05	0.0	1E-04	0.4
DRA 3×3	0.0	0.0	0.0	0.1	0.5	2.8	13.7	0.1
DRA 7×7	0.0	0.0	0.0	0.5	0.1	0.9	2.7	0.5
DV	0.0	0.0	0.0	1.8	0.0	0.0	0.0	2.8
8SED	0.0	0.0	0.0	0.2	1E-02	0.2	1.1	0.2
8SED modified	0.0	0.0	0.0	0.2	1E-02	0.1	1.1	0.1
Euclidean 3×3	22.5	43.9	95.8	0.1	22.4	44.1	95.8	0.1
4SED	0.4	1.0	47.8	0.2	0.5	3.0	48.3	0.2
MD	22.5	43.9	95.8	1.4	22.4	44.1	95.8	1.4
MDDR	0.0	0.0	0.0	1.4	1E-02	0.1	1.1	1.4
SimpleList	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.4

Table 2. Results of various distance transform algorithms applied to an image (left) consisting of a randomly generated set of objects created by sampling from a normal distribution with a mean of the center of the image an a standard deviation of 0.20, and (right) another randomly generated set of objects created by sampling from a normal distribution with a different standard deviation of 0.05

	First randomly generated objects				Second randomly generated objects			
	RMSE	max err	diff	time	RMSE	max err	diff	time
CDA 3×3	0.1	1.0	20.3	0.1	7.0	26.6	87.6	0.1
CDA 5×5	2E-02	0.3	20.3	0.1	1.7	5.9	87.5	0.1
CDA 7×7	1E-02	0.2	20.3	0.3	1.1	4.1	87.6	0.3
Chessboard	0.3	5.0	20.9	0.1	42.3	136.3	88.3	0.1
Cityblock	0.5	5.8	20.7	0.1	65.8	192.7	88.1	0.1
CSED	0.0	0.0	0.0	0.5	2E-05	2E-02	4E-04	0.4
DRA 3×3	4E-03	0.6	3E-02	0.1	1.0	18.2	1.7	0.1
DRA 7×7	4E-05	4E-02	1E-04	0.5	1E-02	1.6	0.1	0.5
DV	0.0	0.0	0.0	4.0	0.0	0.0	0.0	3.0
8SED	3E-04	0.1	3E-03	0.2	5E-03	0.3	0.2	0.2
8SED modified	3E-04	0.1	2E-03	0.2	5E-03	0.3	0.2	0.2
Euclidean 3×3	0.1	1.3	10.9	0.1	9.6	28.0	86.5	0.1
4SED	0.2	2.8	10.2	0.2	0.4	3.4	48.4	0.2
MD	0.1	1.3	10.9	3.1	9.6	28.0	86.5	2.0
MDDR	3E-04	0.1	3E-03	3.2	7E-03	0.3	0.4	1.9
SimpleList	0.0	0.0	0.0	5498.2	0.0	0.0	0.0	854.6

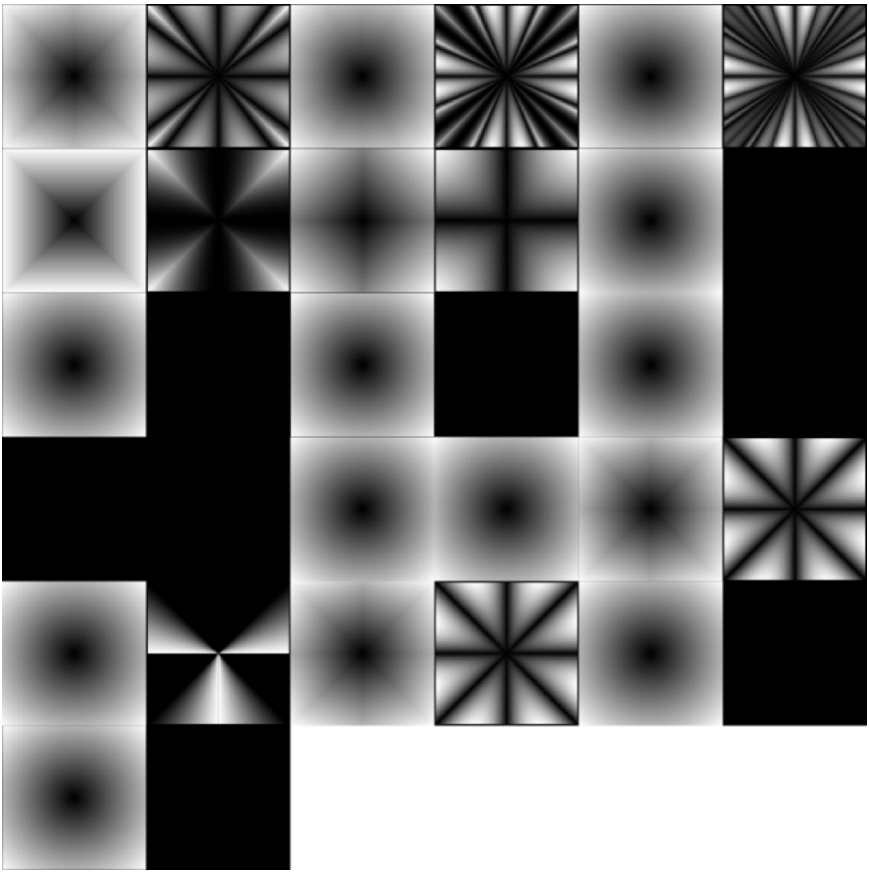


Figure 3. Results of various distance transform algorithms applied to an image consisting of a solitary object consisting of a single point at the center of the image, and the magnitude of the differences for the method. Top row, left to right: CDA 3x3 and differences, CDA 5x5 and differences, CDA 7x7 and differences. Second row, left to right: Chessboard and differences, Cityblock and differences, CSED and difference. Third row, left to right: DRA 3x3 and differences, DRA 7x7 and differences, DV and differences. Fourth row, left to right: 8SED and differences, 8SED modified and differences, Euclidean 3x3 and difference. Fifth row, left to right: 4SED and differences, MD and differences, MDDR and differences. Sixth row: SimpleList and differences.

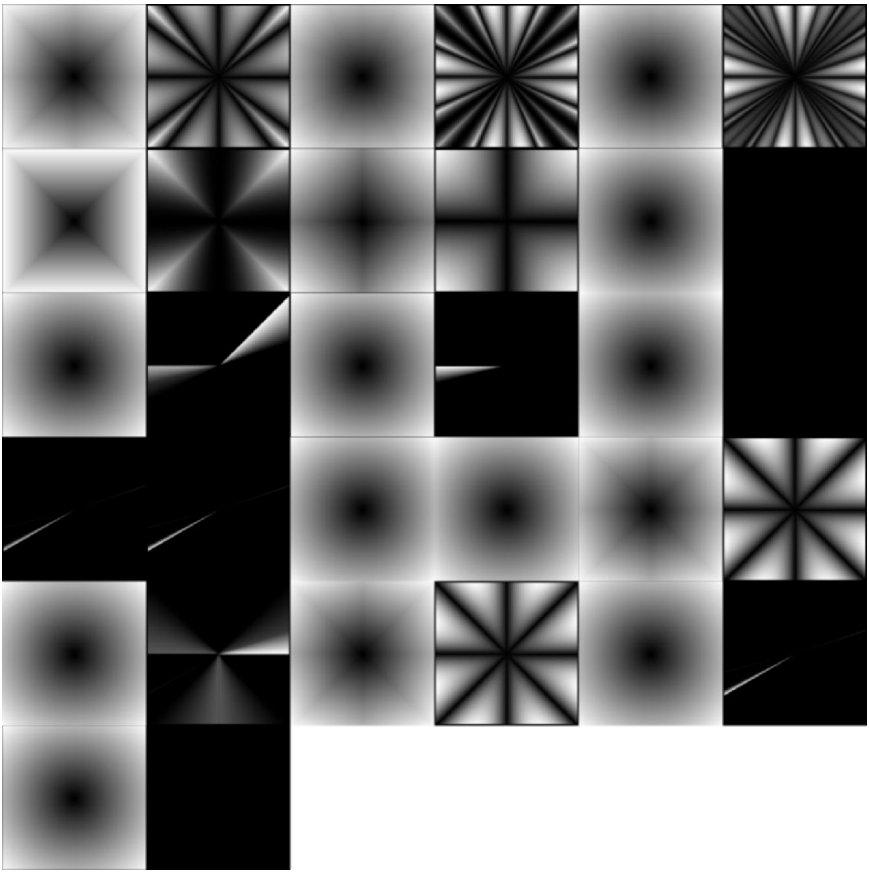


Figure 4. Results of various distance transform algorithms applied to an image consisting of 3 single-point objects. Top row, left to right: CDA 3x3, CDA 5x5, CDA 7x7. Second row, left to right: Chessboard, Cityblock, CSED. Third row, left to right: DRA 3x3, DRA 7x7, DV. Fourth row, left to right: 8SED, 8SED modified, Euclidean 3x3. Fifth row, left to right: 4SED, MD, MDDR. Sixth row: SimpleList.

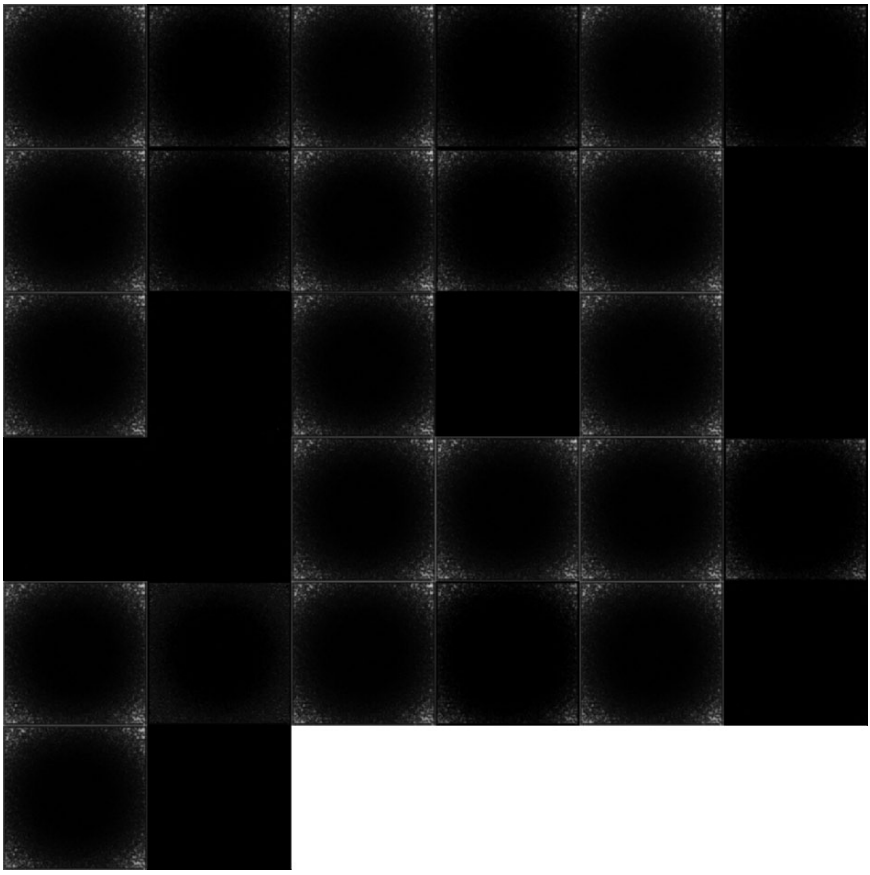


Figure 5. Results of various distance transform algorithms applied to an image consisting of a randomly generated set of objects created by sampling from a normal distribution with a mean of the center of the image an a standard deviation of 0.20. Top row, left to right: CDA 3x3, CDA 5x5, CDA 7x7. Second row, left to right: Chessboard, Cityblock, CSED. Third row, left to right: DRA 3x3, DRA 7x7, DV. Fourth row, left to right: 8SED, 8SED modified, Euclidean 3x3. Fifth row, left to right: 4SED, MD, MDDR. Sixth row: SimpleList.

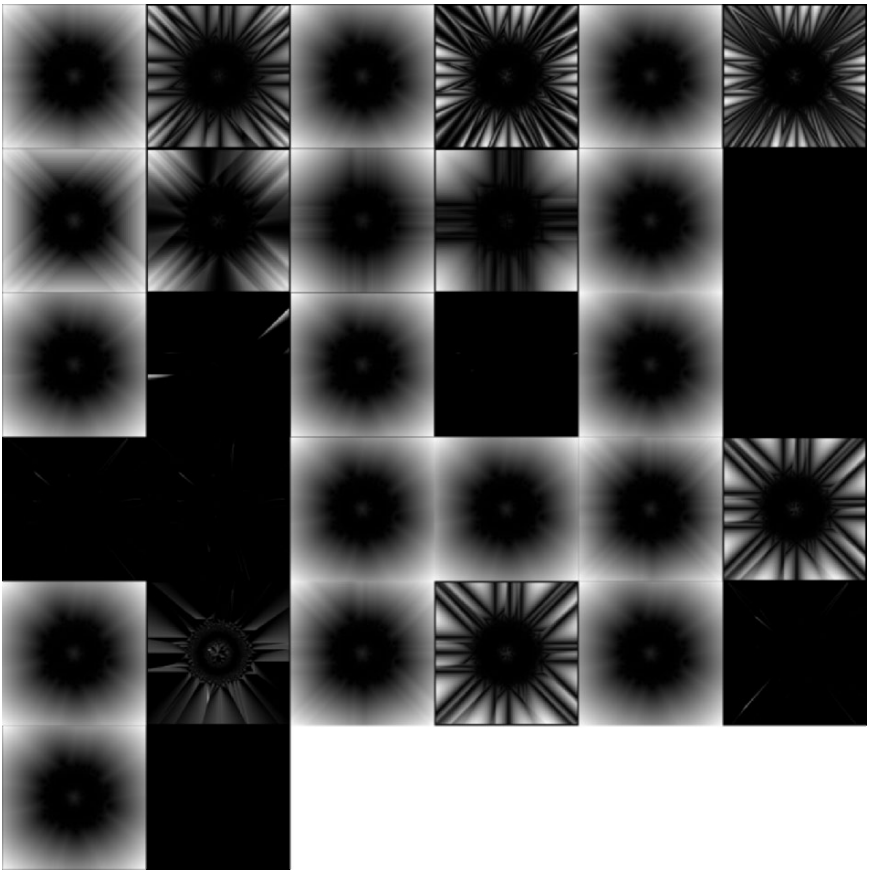


Figure 6. Results of various distance transform algorithms applied to an image consisting of a randomly generated set of objects created by sampling from a normal distribution with a mean of the center of the image an a standard deviation of 0.05. Top row, left to right: CDA 3x3, CDA 5x5, CDA 7x7. Second row, left to right: Chessboard, Cityblock, CSED. Third row, left to right: DRA 3x3, DRA 7x7, DV. Fourth row, left to right: 8SED, 8SED modified, Euclidean 3x3. Fifth row, left to right: 4SED, MD, MDDR. Sixth row: SimpleList.

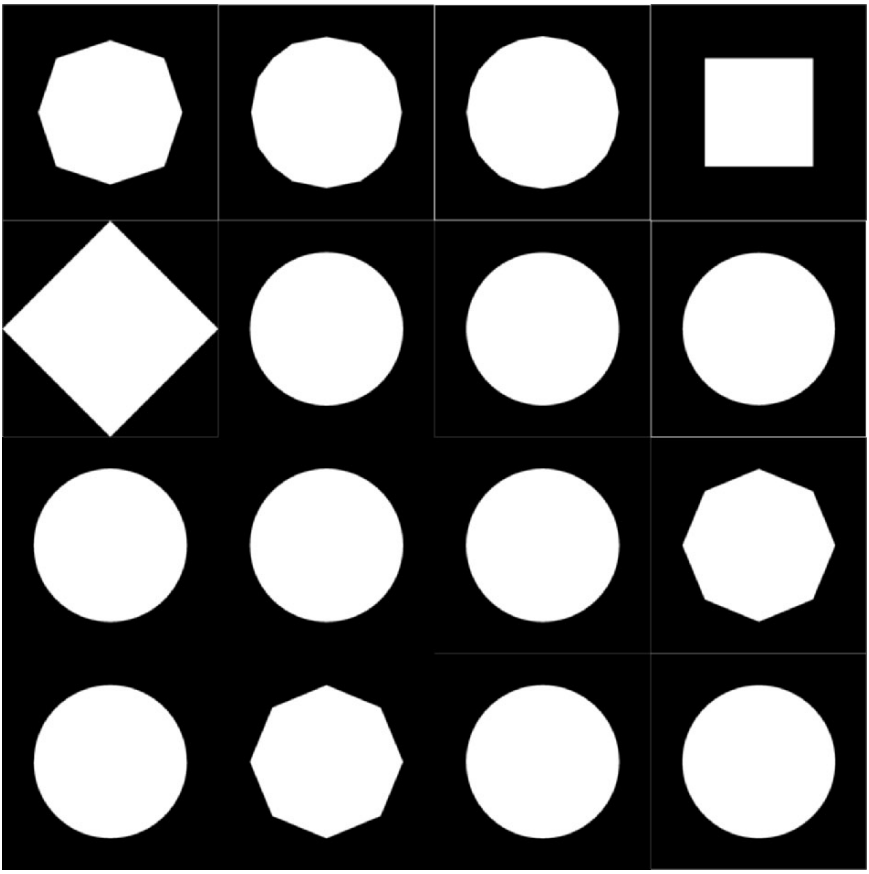


Figure 7. Thresholded results of various distance transform algorithms applied to an image consisting of a solitary object consisting of a single point at the center of the image. Top row, left to right: CDA 3x3, CDA 5x5, CDA 7x7, Chessboard. Second row, left to right: Cityblock, CSED, DRA 3x3, DRA 7x7. Third row, left to right: DV, 8SED, 8SED modified, Euclidean 3x3. Fourth row, left to right: 4SED, MD, MDDR, SimpleList.

7. APPENDIX

The accompanying CD contains implementations (C++ classes that compile and run under Windows with VC++ 6, and Linux and Solaris with g++) of the following distance transform algorithms in an extensible framework. Updates to this software can be found at http://www.sju.edu/∼ggrevera.

- Chamfer2D_3x3
- Chamfer2D_5x5
- Chamfer2D_7x7
- Chessboard2D
- Cityblock2D
- CSED
- DeadReckoning_3x3
- DeadReckoning_7x7
- DijkstraVectors
- EightSED
- EightSED_modified
- errorfreeCSED
- Euclidean2D
- FourSED
- ModifiedDijkstra
- Simple
- SimpleList

Other support classes include:

- **CLUT —** CLUT (Color LookUp Table) class for writing some color TIFF image files
- **Timer —** Timer class for reporting elapsed time and CPU time
- **Normal —** Normal class which samples random numbers from a normal distribution using the Box-Muller transform

- **TIFFWriter —** This class contains methods that write 8-bit color rgb images or float, double, 8-bit, or 16-bit grey images
- **DistanceTransform —** an abstract base class from which all distance transform classes inherit

A VC++ 6 workspace is included along with Windows executables. The Makefile works under both Linux and Solaris. The main.cpp file creates binary test images, applies each of the distance transforms in turn to the test images, evaluates the results, and creates TIFF images of the results.

8. REFERENCES

1. Udupa JK. 1994. Multidimensional digital boundaries. *Comput Vision Graphics Image Process: Graphical Models Image Process* **56**(4):311–323.
2. Cuisenaire O. 1999. Distance transformations: fast algorithms and applications to medical image processing. PhD thesis. Université Catholique de Louvian.
3. Rosenfeld A, Pfaltz JL. 1968. Distance functions on digital pictures. *Pattern Recognit* **1**(1):33–61.
4. Montanari U. 1968. A method for obtaining skeletons using a quasi-Euclidean distance. *J Assoc Comput Machin* **15**:600–624.
5. Danielsson P-E. 1980. Euclidean distance mapping. *Comput Graphics Image Process* **14**:227–248.
6. Borgefors G. 1986. Distance transformations in digital images. *Comput Vision Graphics Image Process* **34**:344–371.
7. Borgefors G. 1996. On digital distance transforms in three dimensions. *Comput Vision Image Understand* **64**(3):368–376.
8. Butt MA, Maragos P. 1998. Optimum design of chamfer distance transforms. *IEEE Trans Image Process* **7**(10):1477–1484.
9. Marchand-Maillet S, Sharaiha YM. 1999. Euclidean ordering via Chamfer distance calculations. *Comput Vision Image Understand* **73**(3):404–413.
10. Nilsson NJ. *Artificial intelligence: a new synthesis*. San Francisco: Morgan Kaufmann, 1998.
11. Verwer BJH, Verbeek PW, Dekker ST. 1989. An efficient uniform cost algorithm applied to distance transforms. *IEEE Trans Pattern Anal Machine Intell* **11**(4):425–429.
12. Leymarie F, Levine MD. 1992. Fast raster scan distance propagation on the discrete rectangular lattice. *Comput Vision Graphics Image Process: Image Understand* **55**(1):84–94.
13. Satherley R, Jones MW. 2001. Vector-city vector distance transform. *Comput Vision Image Understand* **82**:238–254.
14. Ragnemalm I. 1992. Neighborhoods for distance transformations using ordered propagation. *Comput Vision Graphics Image Process: Image Understand* **56**(3):399–409.
15. Guan W, Ma S. 1998. A list-processing approach to compute Voronoi diagrams and the Euclidean distance transform. *IEEE Trans Pattern Anal Machine Intell* **20**(7):757–761.
16. Eggers H. 1998. Two fast Euclidean distance transformations in Z2 based on sufficient propagation. *Comput Vision Image Understand* **69**(1):106–116.
17. Saito T, Toriwaki J-I. 1994. New algorithms for euclidean distance transformation of an n-dimensional digitized picture with application. *Pattern Recognit* **27**(11):1551–1565.
18. Boxer L, Miller R. 2000. Efficient computation of the Euclidean distance transform. *Comput Vision Image Understand* **80**:379–383.
19. Meijster A, Roerdink JBTM, Hesselink WH. 2000. A general algorithm for computing distance transforms in linear time. In *Mathematical morphology and its applications to image and signal processing*, pp. 331–340. Ed. J Goutsias, L Vincent, DS Bloombers. New York: Kluwer.

20. Lotufo RA, Falcao AA, Zampirolli FA. 2000. Fast Euclidean distance transform using a graph-search algorithm. *SIBGRAPI* **2000**:269–275.
21. da Fontoura Costa L. 2000. Robust skeletonization through exact Euclidean distance transform and its application to neuromorphometry. *Real-Time Imaging* **6**:415–431.
22. Pudney C. 1998. Distance-ordered homotopic thinning: a skeletonization algorithm for 3D digital images. *Comput Vision Image Understand* **72**(3):404–413.
23. Sanniti di Baja G. 1994. Well-shaped, stable, and reversible skeletons from the (3,4)-distance transform. *J Visual Commun Image Represent* **5**(1):107–115.
24. Svensson S, Borgefors G. 1999. On reversible skeletonization using anchor-points from distance transforms. *J Visual Commun Image Represent* **10**:379–397.
25. Herman GT, Zheng J, Bucholtz CA. 1992. Shape-based interpolation, *IEEE Comput Graphics Appl* **12**(3):69–79.
26. Raya SP, Udupa JK. 1990. Shape-based interpolation of multidimensional objects. *IEEE Trans Med Imaging* **9**(1):32–42.
27. Grevera GJ, Udupa JK. 1996. Shape-based interpolation of multidimensional grey-level images. *IEEE Trans Med Imaging* **15**(6):881–892.
28. Kozinska D. 1997. Multidimensional alignment using the Euclidean distance transform. *Graphical Models Image Process* **59**(6):373–387.
29. Paglieroni DW. 1997. Directional distance transforms and height field preprocessing for efficient ray tracing. *Graphical Models Image Process* **59**(4):253–264.
30. Remy E, Thiel E. 2000. Computing 3D medial axis for Chamfer distances. *Discrete Geom Comput Imagery* pp. 418–430.
31. Remy E, Thiel E. 2002. Medial axis for chamfer distances: computing look-up tables and neighbourhoods in 2D or 3D. *Pattern Recognit Lett* **23**(6):649–662.
32. Grevera GJ, Udupa JK. 1998. An objective comparison of 3D image interpolation methods. *IEEE Trans Med Imaging* 17(4):642–652.
33. Grevera GJ, Udupa JK, Miki Y. 1999. A task-specific evaluation of three-dimensional image interpolation techniques. *IEEE Trans Med Imaging* **18**(2):137–143.
34. Travis AJ, Hirst DJ, Chesson A. 1996. Automatic classification of plant cells according to tissue type using anatomical features obtained by the distance transform. *Ann Botany* **78**:325–331.
35. Van Der Heijden GWAM, Van De Vooren JG, Van De Wiel CCM. 1995. Measuring cell wall dimensions using the distance transform. *Ann Botany* **75**:545–552.
36. Schnabel JA, Wang L, Arridge SR. 1996. Shape description of spinal cord atrophy in patients with MS. Comput Assist Radiol ICS **1124**:286–291.
37. Grevera GJ. 2004. The “dead reckoning” signed distance transform. *Comput Vision Image Understand* **95**:317-333.
38. Oppenheim AV, Schafer RW, Buck JR. 1999. *Discrete-time signal processing*, 2d ed. Englewood Cliffs: Prentice Hall.
39. Cormen TH, Leiserson CE, Rivest RL, Stein C. 2001. *Introduction to algorithms*, 2d ed. Cambridge: MIT Press.
40. Svensson S, Borgefors G. 2002. Digital distance transforms in 3D images using information from neighbourhoods up to 5x5x5. *Comput Vision Image Understand* **88**:24-53.

3

LEVEL SET TECHNIQUES FOR STRUCTURAL INVERSION IN MEDICAL IMAGING

Oliver Dorn

Universidad Carlos III de Madrid, España

Dominique Lesselier

Laboratoire des Signaux et Systèmes
Gif sur Yvette, France

Most biological bodies are structured in the sense that they contain quite well-defined interfaces between regions of different types of tissue or anatomical material. Extracting structural information from medical or biological images has been an important research topic for a long time. Recently, much attention has been devoted to quite novel techniques for the direct recovery of structural information from physically measured data. These techniques differ from more traditional image processing and image segmentation techniques by the fact that they try to recover structured images not from already given pixel or voxel-based reconstructions (obtained, e.g., using traditional medical inversion techniques), but directly from the given raw data. This has the advantage that the final result is guaranteed to satisfy the imposed criteria of data fitness as well as those of the given structural prior information. The 'level-set-technique' [1–3] plays an important role in many of these novel structural inversion approaches, due to its capability of modeling topological changes during the typically iterative inversion process. In this text we will provide a brief introduction into some techniques that have been developed recently for solving structural inverse problems using a level set technique.

Address all correspondence to: Oliver Dorn, Departamento de Matemáticas, Universidad Carlos III de Madrid, Avenida de la Universidad, 30, 28911 Leganés, Madrid, España. Phone: (+34)91-6248825. Fax: (+34)91-6249129. odorn@math.uc3m.es. http://www.athena.uc3m.es/~dorn/.

1. INTRODUCTION

Level set techniques for solving inverse problems were first proposed by Santosa [4]. Further examples for early contributions are (without claim of completeness of this list) [5–11]. By now, many more results in a variety of applications can be found in the literature. We refer to the recent overview articles [12–14], each of them illuminating the recent progress in this exciting research topic with a different viewpoint. An overview of level set techniques in medical imaging can be found in [15]. In the present text, we intend to give a general introduction into level set techniques in medical imaging in the above-described sense of direct reconstruction of structured images from raw data. We will follow two typical examples for this purpose, namely X-ray computerized tomography (CT) and diffuse optical tomography (DOT). Both are representative for broader classes of inverse problems, the first one representing linear tomography problems, and the second nonlinear ones. The theory for both can be developed fairly in parallel, with some characteristic differences. We will point out these analogies as well as the differences.

This chapter is organized as follows. In Section 2 we introduce the use of level set techniques for linear inverse problems, in particular X-ray CT. First, we describe the more traditional pixel-based filtered backprojection technique, which admits an interesting geometric interpretation, to be compared then with an alternative gradient-based scheme for pixel-based inversion. We then extend this gradient-based scheme to the situation of structural inversion using a level set technique. We first concentrate on the search of descent directions for finding only unknown shapes from given data, which will then be generalized to joint inversion for interfaces and structural information in each individual region. Then, some popular geometric regularization techniques are described for shape inversion using level sets. At the end of the section we give a few hints for work in the literature addressing linear inverse problems using level sets. The second part of the text is concerned with nonlinear inverse problems, which is discussed in Section 3. It starts with generalizing the concepts already seen for the linear case to this more general situation. The main example is diffuse optical tomography. First, the mathematical and physical background of this novel imaging technique is explained briefly. An important component of nonlinear inverse problems by iterative gradient-based schemes is efficient calculation of linearized parameter-to-data mappings in each step. One powerful way of doing so (the so-called adjoint scheme) will be described for our specific application. Then, some numerical examples are presented that illustrate the general behavior of shape reconstruction schemes using level sets. Finally, some more literature references will be given for level set techniques in nonlinear medical imaging problems, which concludes the second part of the text.

2. LEVEL SET TECHNIQUES FOR LINEAR INVERSE PROBLEMS

In this section we will focus on linear inverse problems in medical imaging. We will concentrate on x-ray computerized tomography, which will serve as an example for a broader class of linear inverse problems. Further examples will be described briefly in Section 2.10.

2.1. Example: X-Ray CT

We start with a classical example, namely x-ray computerized tomography (CT). Here the propagation of x-rays through tissue is modeled mathematically by a stationary linear transport equation of the form

$$\theta \cdot \nabla u(\mathbf{x}, \theta) + \mu(\mathbf{x}) u(\mathbf{x}, \theta) = q(\mathbf{x}, \theta) \quad \text{in} \quad \Omega, \tag{1}$$

with boundary condition

$$u(\mathbf{x}, \theta) = 0 \quad \text{for} \quad (\mathbf{x}, \theta) \in \tilde{\Gamma}_-. \tag{2}$$

Here we use the notation

$$\tilde{\Gamma}_\pm = \Big\{ (\mathbf{x}, \theta) \in \partial\Omega \times S^1, \quad \pm\theta \cdot \mathbf{n}(\mathbf{x}) > 0 \Big\}, \tag{3}$$

where $\mathbf{n}(\mathbf{x}) \in S^1$ (S^1 being the unit circle in $\mathbb{R}^2$) denotes the outward unit normal to $\partial\Omega$ in the point $\mathbf{x} \in \partial\Omega$. In this model, $u(\mathbf{x}, \theta)$ denotes the density of X-ray photons propagating at position $\mathbf{x} \in \Omega$ in direction $\theta \in S^1$. Boundary condition (2) indicates that there are no x-ray photons entering the domain. Instead, in our model the photons are created by the source term, $q(\mathbf{x}, \theta)$, in (1) which can be located at the boundary of the domain. In fact, we have a duality between boundary sources and incoming boundary conditions, as pointed out in [16], so we can choose between putting either an inhomogeneity in (2) or using a source term in (1) for creating x-ray photons at the boundary of the domain. Usually, there is an additional energy variable involved when modeling x-ray photons, which is neglected in (1), (2) for simplicity.

For x-ray CT, the quantity of interest is the attenuation, $\mu(\mathbf{x})$, in (1), which models loss of particles propagating in direction θ due to various physical processes. We will assume here that this quantity does not depend on the direction θ in which the x-rays propagate at position $\mathbf{x}$, but it will depend on position $\mathbf{x}$. Knowing the spatial distribution of $\mu(\mathbf{x})$ in the human body, the physician can gain important information about the structure of the body. Usually, in clinical applications of this imaging technique, a large amount of high-quality x-ray data is available, such that reconstruction of function $\mu(\mathbf{x})$ in form of a pixel-based array

representing the attenuation distribution in the body is possible, whose resolution primarily depends on the measurement geometry and the sampling rate [17–19]. We will briefly mention below how this is achieved in practice. In situations where the gathered data are sparse and/or have a limited geometry, the situation changes. These so-called 'limited-view' situations still cause problems when aiming at reconstructing density variations in the body from the x-ray data. This situation might be, however, a good candidate for applying level set techniques, if some additional prior information is available. This will be pointed out in more detail in the remainder of this text.

2.2. Filtered Backprojection in CT

Let us consider a measurement geometry with sources

$$q_{jk} \;=\; I_0 \delta(\mathbf{x} - \mathbf{s}_j)\delta(\theta - \theta_k) \tag{4}$$

for $j = 1, \ldots, \underline{j}$, $k = 1, \ldots, \underline{k}$. This can be, for example, a fan beam or a parallel beam geometry [17, 18]. Each source q_{jk} defines a line L_{jk} along which the x-ray photons are propagating through the tissue, until they reach the corresponding detector position which is denoted by $\mathbf{d}_{jk}$. Integration of (1) along this line (which at the same time is a characteristic for (1)) yields the field u_{jk} along this line. The measured intensity at detector position $\mathbf{d}_{jk}$ is then given by

$$I_{jk} \;=\; u_{jk}(\mathbf{d}_{jk}, \theta_k) \;=\; I_0\, e^{-\int_{L_{jk}} \mu(\mathbf{x}) d\mathbf{x}}. \tag{5}$$

What actually is used in CT as data are the quantities

$$g_{jk} \;=\; -\log\left(\frac{I_{jk}}{I_0}\right) \;=\; \int_{L_{jk}} \mu(\mathbf{x}) d\mathbf{x}, \tag{6}$$

which are the line integrals of the attenuation over the lines L_{jk}. In the continuous setting, when obtaining the line integrals over all possible lines, $L(s, \theta) = \{\mathbf{x} : \mathbf{x} \cdot \theta = s\}$, through Ω for $s \in \mathbb{R}^1$ and $\theta \in S^1$, the data $g(s, \theta)$ are given by the Radon transform of the attenuation $\mu(\mathbf{x})$:

$$(T\mu)(s, \theta) \;=\; g(s, \theta) \;=\; \int_{\Omega} \delta_{L(s,\theta)}(\mathbf{x}) \mu(\mathbf{x}) d\mathbf{x}, \tag{7}$$

where $\delta_{L(s,\theta)}(\mathbf{x})$ is the two-dimensional Dirac delta distribution concentrated on the line $L(s, \theta)$. The operator T maps functions on $\mathbb{R}^2$ to functions on $\mathbb{R}^1 \times S^1$. An analytical inversion formula is available [20] for this transform. In commercial CT scanners, its implementation is done in form of the *filtered backprojection*

technique, which takes into account the sampling rate of the data, the measurement geometry that is used, and other details of the experimental setup. The *backprojection operator* T^* is just the adjoint of T and is defined by [18, 19]

$$(T^*g)(\mathbf{x}) = \int_{S^1} g(\mathbf{x}\cdot\theta,\theta)d\theta. \tag{8}$$

It maps functions on $\mathbb{R}^1 \times S^1$ back to functions on $\mathbb{R}^2$. More precisely, let us denote by Z the data space equipped with the inner product

$$\big\langle g_1(\theta,s)\,,\, g_2(\theta,s)\big\rangle_Z \;=\; \int_{S^1}\int_{R^1} g_1(\theta,s)g_2(\theta,s)\,d\theta ds, \tag{9}$$

and by P the parameter space (of attenuation functions) equipped with the inner product

$$\big\langle \mu_1(\mathbf{x})\,,\, \mu_2(\mathbf{x})\big\rangle_P \;=\; \int_{\mathbf{R}^2} \mu_1(\mathbf{x})\mu_2(\mathbf{x})\,d\mathbf{x}. \tag{10}$$

Then we have

$$\big\langle (T\mu)(\theta,s)\,,\, g(\theta,s)\big\rangle_Z \;=\; \big\langle \mu(\mathbf{x})\,,\, (T^*g)(\mathbf{x})\big\rangle_P\,. \tag{11}$$

The backprojection operator can be interpreted physically as a process that is taking all gathered data and 'projects them back' uniformly over the lines that have contributed to them. Summing up all these contributions yields T^*g. The filtered backprojection reconstruction scheme can then be described as an implementation of the formula

$$(V*\mu)(\mathbf{x}) \;=\; \Big(T^*(v*g)\Big)(\mathbf{x}). \tag{12}$$

Here, v is a filtering operator that is chosen such that $V = T^*v$ approximates the two-dimensional Dirac delta distribution concentrated in $\mathbf{x}$. This will have the effect that $V*\mu$ is an approximation to μ. In physical terms, the filtered data $v*g$ are backprojected by application of T^* such that the result $V*\mu$ approximates as much as possible the sought attenuation μ.

Notice that the filtered backprojection scheme calculates for each pixel in the domain an individual value of the attenuation μ from the given data g using formula (12). This usually yields good results if sufficient data of high quality are available. In Section 2.4 we will introduce into the shape-based reconstruction idea, which takes into account additional prior information about the structure of the tissue. As said already, this is useful in cases where the quality or quantity of the given data are not sufficient for a successful application of formula (12). Adding prior information to a reconstruction is often called 'regularization.' In this sense, the shape-based approach can be considered as a specific form of regularization, which is practically achieved by using a level set technique. Before introducing the shape-based approach, we want to derive in the following section an iterative gradient scheme for data inversion in CT, which will then lead us directly to the level set based strategies.

2.3. Least-Square Cost Functionals and Gradient Directions

We define the least-squares cost functional

$$\mathcal{J}(\mu) \;=\; \frac{1}{2}\big\|T\mu - g\big\|_Z^2 \;=\; \frac{1}{2}\Big\langle\, T\mu - g\,,\, T\mu - g\,\Big\rangle_Z, \tag{13}$$

where $\langle\,,\,\rangle_Z$ denotes the canonical inner product in data space Z as defined in (9). Let us perturb the attenuation $\mu(\mathbf{x})$ by a small amount $\delta\mu(\mathbf{x})$. Then, the cost changes according to

$$\begin{aligned}\mathcal{J}(\mu+\delta\mu) &= \mathcal{J}(\mu) \,+\, \Big\langle T\mu - g\,,\, T\delta\mu\Big\rangle_Z \,+\, \frac{1}{2}\|T\delta\mu\|_Z^2 \\ &= \mathcal{J}(\mu) \,+\, \Big\langle T^*\,(T\mu - g)\;,\, \delta\mu\Big\rangle_P \,+\, O(\|T\delta\mu\|_Z^2), \end{aligned}\tag{14}$$

where we have used the fact that the backprojection operator T^*, as defined in (8), is the adjoint operator of T. We call

$$\mathbf{grad}_{\mathcal{J}}(\mu) \;=\; T^*\,(T\mu - g) \tag{15}$$

the *gradient direction* of $\mathcal{J}$ in μ. A gradient-type method for reconstructing the attenuation μ aims at calculating the minimizer

$$\min_{\mu}\; \mathcal{J}(\mu) \tag{16}$$

by starting with a guess, $\mu^{(0)}$, and updating it successively by applying in step n of the iteration the correction

$$\mu^{(n+1)} \;=\; \mu^{(n)} - \lambda \mathbf{grad}_{\mathcal{J}}(\mu^{(n)}) \tag{17}$$

for a sufficiently small step-size $\lambda > 0$. Doing so, the cost $\mathcal{J}(\mu)$ is reduced in each step as can be seen by plugging $\delta\mu = -\lambda T^*(T\mu - g)$ into (14). Certainly, such an iterative gradient approach would be much slower than applying the filtered backprojection scheme when sufficient high-quality data can be used for the reconstruction. However, as mentioned previously, there are situations where only few, noisy, and irregularly sampled data are available. In these situations, iterative algorithms become more interesting due to their capability of incorporating a-priori information in a very simple and flexible way. Certainly, our interest in the gradient scheme is not motivated so much by its applicability to classical pixel-based attenuation reconstruction from x-ray data, but by the fact that it directly leads us to the most basic level set reconstruction technique for finding attenuation distributions of the form (18) from the data in an iterative fashion. How to do this, will be demonstrated in the following.

2.4. Shape-Based Reconstruction in CT

In the shape-based approach to image reconstruction in CT we make the assumption that

$$\mu(\mathbf{x}) = \begin{cases} \mu_i(\mathbf{x}) & \text{in} \quad D \\ \mu_e(\mathbf{x}) & \text{in} \quad \Omega \backslash D. \end{cases} \tag{18}$$

In other words, the attenuation is assumed to be equal to some interior function $\mu_i(\mathbf{x})$ inside the shape $D \subset \Omega$, and equal to some exterior function $\mu_e(\mathbf{x})$ outside this shape. Along the shape boundary $\Gamma = \partial D$ the attenuation will typically be discontinuous. Here, $\mu_i(\mathbf{x})$ and $\mu_e(\mathbf{x})$ can in general be arbitrary functions. However, the most typical application of level set techniques assumes so far that $\mu_i(\mathbf{x})$ is equal to a constant value inside some sought anomaly (e.g., a tracer distribution) in the body, and $\mu_e(\mathbf{x})$ is a predefined homogeneous or inhomogeneous background attenuation distribution. For simplicity, we start our discussion with this simple but already quite useful model.

In many applications, the unknown region D has a complicated structure. We will assume that it can be written as

$$D = \bigcup_{m=1}^{\underline{m}} D_m, \quad D_m \cap D_{m'} = \emptyset \quad \text{for} \quad m \neq m', \tag{19}$$

where the number of components $\underline{m}$ might be arbitrary (but finite). Moreover, this number $\underline{m}$ typically is not known a priori, such that we need to be able to easily model topological changes during an iterative reconstruction process. The level set technique is well known to be able to handle topological changes in a completely automatic fashion. Typical parameterized models would need a cumbersome reparameterization procedure in order to achieve this goal.

2.5. Level Set Representation of the Shapes

We define the *characteristic function* $\chi_D : \Omega \to \{0, 1\}$ for a given shape D as

$$\chi_D(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in D \\ 0, & \mathbf{x} \in \Omega \backslash D. \end{cases} \tag{20}$$

For simplicity, we will always assume here that $\partial D \subset D$, which means that the shape D is a closed set. Given a sufficiently smooth function $\phi : \Omega \to \mathbb{R}$, we call ϕ a *level set representation of the shape* D [1] if

$$\begin{cases} \phi(\mathbf{x}) \leq 0 & \text{for all} \quad \mathbf{x} \in D, \\ \phi(\mathbf{x}) > 0 & \text{for all} \quad \mathbf{x} \in \Omega \backslash D. \end{cases} \tag{21}$$

The level set representation of a given shape, D, is not unique, since it can be modified some distance away from the interfaces without changing the shape.

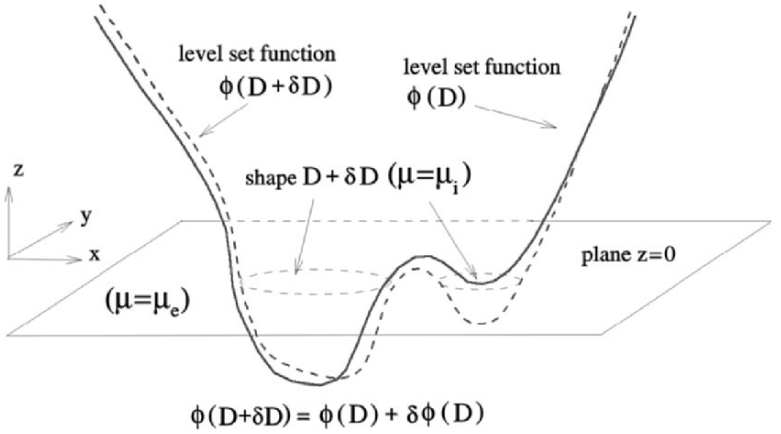


Figure 1. Deformation of shapes by the level set formulation.

On the other hand, every given sufficiently smooth (e.g., Lipschitz-continuous) function ϕ uniquely specifies a corresponding shape (which we denote $D[\phi]$) by the above definitions. Assuming that $|\nabla\phi|$ is well defined and nonzero along the shape boundary represented by a smooth level set function ϕ, we can characterize the boundary $\Gamma = \partial D$ as

$$\partial D \;=\; \{\mathbf{x} \in \Omega, \quad \phi(\mathbf{x}) = 0\}. \tag{22}$$

2.6. Shape Deformation and Change in the Cost

The link between pixel-based reconstruction and a level set-based reconstruction scheme was first established in [4]. We will demonstrate in the following the basic idea applied to x-ray CT.

A shape deformation can be modeled as follows. We assign to each point in the domain a *velocity field* $\mathbf{v}(\mathbf{x})$ and let the points $\mathbf{x} \in \Omega$ move a small artificial evolution time $[0, \tau]$ with constant velocity $\mathbf{v}(\mathbf{x})$. This gives rise to a displacement $\mathbf{y}(\mathbf{x})$ of each point if the domain modeled by

$$\mathbf{y}(\mathbf{x}) \;=\; \mathbf{v}(\mathbf{x})\tau. \tag{23}$$

We assume that the velocity field $\mathbf{v}(\mathbf{x})$ is regular enough such that the basic structure of D remains preserved during this evolution. Then, the points located on the boundary, $\Gamma = \partial D$, will move to the new locations $\mathbf{x}' = \mathbf{x} + \mathbf{y}(\mathbf{x})$, and boundary Γ will be deformed into a new boundary, $\Gamma' = \partial D'$. Assuming furthermore that the parameter distribution in Ω has the special form (18), it will change as well. In the following, we want to quantify this change in the parameter distribution

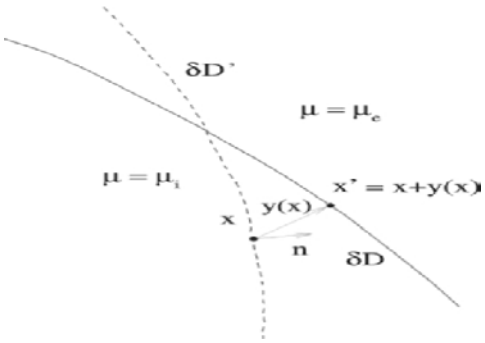


Figure 2. Deformation of shapes using calculation of small variations.

$\mu(\mathbf{x})$ due to an infinitesimal deformation as described above. As already said, we follow the approach given in [4] for a sufficiently regular boundary ∂D using formal calculation of variations.

Consider the inner product of $\delta\mu$ with a test function f:

$$\langle \delta\mu\,,\, f\rangle_\Omega \;=\; \int_\Omega \delta\mu(\mathbf{x}) f(x)\, d\mathbf{x} \;=\; \int_{\text{symdiff}(D,D')} \delta\mu(\mathbf{x}) f(\mathbf{x})\, d\mathbf{x}, \qquad (24)$$

where $\text{symdiff}(D, D') = (D \cup D')\backslash(D \cap D')$ is the symmetric difference of the sets D and D' (see Figure 2, inspired from [4]). Since the difference between D and D' is infinitesimal, we can reduce the area integral to a line integral. Let $\mathbf{n}(\mathbf{x})$ denote the outward normal to $\mathbf{x}$. Then, the integral in (24) becomes

$$\langle \delta\mu\,,\, f\rangle_{\partial D} \;=\; \int_{\delta D} (\mu_i(\mathbf{x}) - \mu_e(\mathbf{x}))\mathbf{y}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) f(\mathbf{x})\, ds(\mathbf{x}), \qquad (25)$$

where $ds(\mathbf{x})$ is the incremental arclength. We have used the fact that in the limit $\delta\mu(\mathbf{x}) = \mu_i(\mathbf{x}) - \mu_e(\mathbf{x})$ at boundary point $\mathbf{x} \in \partial D$ due to (18).

We arrive at the result

$$\delta\mu(\mathbf{x}) \;=\; (\mu_i(\mathbf{x}) - \mu_e(\mathbf{x}))\, \mathbf{y}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) \quad \text{at } \partial D, \qquad (26)$$

from which $\delta\mu(\mathbf{x})$ can be interpreted as a surface measure on ∂D. Alternatively, using the n-dimensional Dirac delta distribution $\delta_{\partial D}$ concentrated on the boundary ∂D of D, we can write (26) in the form

$$\delta\mu(\mathbf{x}) \;=\; (\mu_i - \mu_e)\, \mathbf{y}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})\, \delta_{\partial D}(\mathbf{x}), \qquad (27)$$

which is a distribution defined in the entire domain Ω but concentrated along ∂D.

Plugging now (23) into (27) we get for $t \in [0, \tau]$ the corresponding change in the parameters:

$$\delta\mu(\mathbf{x}; t) \;=\; (\mu_i - \mu_e)\, \mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) t\, \delta_{\partial D}(\mathbf{x}). \tag{28}$$

Plugging expression (28) into (14) and neglecting terms of higher than linear order, we arrive at

$$\mathcal{J}(\mu(t)) - \mathcal{J}(\mu(0)) \;=\; \Big\langle T^* \left(T\mu - g\right) \,,\, \delta\mu(\mathbf{x}; t) \Big\rangle_P \tag{29}$$

$$= \Big\langle T^* \left(T\mu - g\right) \,,\, (\mu_i - \mu_e)\, \mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) t\, \delta_{\partial D}(\mathbf{x}) \Big\rangle_P ,$$

or, in the limit $t \to 0$, evaluating the Dirac delta distribution, we get

$$\left.\frac{\partial \mathcal{J}(\mu)}{\partial t}\right|_{t=0} \;=\; \int_{\partial D} \Big[T^* \left(T\mu - g\right)\Big] (\mu_i - \mu_e) \mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) ds(\mathbf{x}). \tag{30}$$

If we succeed in finding a velocity field $\mathbf{v}(\mathbf{x})$ such that $\left.\frac{\partial \mathcal{J}(\mu)}{\partial t}\right|_{t=0} < 0$, this inequality will hold in a sufficiently small time interval $[0, \tau]$ (for continuity reasons), and the total cost during the artificial flow will be reduced. In the following section we will demonstrate how this approach yields a direct link to more classical shape evolution by the level set technique.

2.7. Shape Evolution by the Level Set Technique

In the previous section we have derived an expression for the change in the cost due to small displacements of the interfaces produced by a velocity field $\mathbf{v}(\mathbf{x})$. We observe that only the normal component of the velocity field

$$F(\mathbf{x}) \;=\; \mathbf{v}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) \tag{31}$$

at the boundary ∂D of the shape D is of relevance for the change in the cost (30). This is because tangential components do not contribute to shape deformations. In order to solve the inverse problem, we now have to *choose* a velocity field $\mathbf{v}(\mathbf{x})$ such that the cost is reduced during the corresponding shape evolution. One obvious choice for such a velocity is

$$F_{SD}(\mathbf{x}) \;=\; -T^* \left(T\mu - g\right) (\mu_i - \mu_e) \quad \text{at } \partial D, \tag{32}$$

which is often called the gradient (or steepest-descent) direction with respect to the cost. Plugging this into (30) yields a descent direction for the cost $\mathcal{J}$. What remains to be done now is to numerically implement the resulting flow equation in order to follow the steepest descent flow. The practical procedure for this is standard [1–3]. We formulate the basic *Hamilton-Jacobi-type equation*

$$\frac{\partial \phi}{\partial t} \;+\; F_{SD}(\mathbf{x}, t) \cdot |\nabla \phi| \;=\; 0 \tag{33}$$

for the representing level set function ϕ. Notice that the velocity function (32) needs to be recalculated at each step of the artificial shape evolution (33) for each point of the current shape boundary $\mathbf{x} \in \partial D$. This means that a Radon transform needs to be applied to the current attenuation distribution $\mu(\mathbf{x}; t)$ at time step t, and then the backprojection operator T^* needs to be applied to the difference in the data $T\mu(\mathbf{x}; t) - g$ in order to calculate (32). Moreover, appropriate extension velocities need to be constructed at each step for the numerical implementation of the shape evolution equation (33).

2.8. Combined Inversion for Shape and Texture

As mentioned earlier, the search for shape and additional texture information inside and outside of the shape is possible and, using the theory developed so far in this text, more or less straightforward. We will outline a possible approach for this joint inversion in the following. Since in a typical application of shape-based inversion techniques only few and noisy data are available, we will assume that the texture information is described by a small set of basis functions whose coefficients need to be determined, in addition to the shape, from the given data [21].

Let us assume that the smoothly varying attenuation distributions, $\mu_i(\mathbf{x})$ and $\mu_e(\mathbf{x})$, are represented by

$$\mu_i(\mathbf{x}) = \sum_{j=1}^{N_i} \alpha_j a_j(\mathbf{x}), \qquad \mu_e(\mathbf{x}) = \sum_{k=1}^{N_e} \beta_k b_k(\mathbf{x}), \tag{34}$$

with basis functions $a_j(\mathbf{x})$ and $b_k(\mathbf{x})$ and expansion coefficients α_j and β_k, respectively. The cost (data misfit) will now not only depend on the shape D (i.e., the describing level set function ϕ), but also on the set of expansion coefficients $\{\alpha_j\}$ and $\{\beta_k\}$. We write this relationship in the form

$$\mathcal{J}(\phi, \{\alpha_j\}, \{\beta_k\}) = \frac{1}{2} \big\| T\mu(\phi, \{\alpha_j\}, \{\beta_k\}) - g \big\|_Z^2. \tag{35}$$

We will now take a slightly different approach than before. We want to find a general time-evolution law for the unknowns of the problem that reduces gradually the cost (35). We make the ansatz

$$\frac{d\phi}{dt} = f(\mathbf{x}, t), \qquad \frac{d\alpha_j}{dt} = \hat{f}_j(t), \qquad \frac{d\beta_k}{dt} = \tilde{f}_k(t), \tag{36}$$

where $f(\mathbf{x}, t)$ does depend on the spatial position $\mathbf{x}$ but $\hat{f}_j(t)$ and $\tilde{f}_k(t)$ do not. With evolving time, the cost will evolve as well. Using the chain rule, we get

$$\frac{d\mathcal{J}}{dt} = \frac{d\mathcal{J}}{d\mu} \frac{\partial \mu}{\partial \phi} \frac{d\phi}{dt} + \sum_{j=1}^{N_i} \frac{d\mathcal{J}}{d\mu} \frac{\partial \mu}{\partial \alpha_j} \frac{d\alpha_j}{dt} + \sum_{k=1}^{N_e} \frac{d\mathcal{J}}{d\mu} \frac{\partial \mu}{\partial \beta_k} \frac{d\beta_k}{dt}. \tag{37}$$

We write the attenuation distribution in the form

$$\begin{aligned}\mu(\mathbf{x}) &= \mu_i(\mathbf{x})(1-H(\phi)) + \mu_e(\mathbf{x})H(\phi) \\ &= \sum_{j=1}^{N_i}\alpha_j a_j(\mathbf{x})(1-H(\phi)) + \sum_{k=1}^{N_e}\beta_k b_k(\mathbf{x})H(\phi),\end{aligned} \tag{38}$$

where $H(\phi)$ denotes the one-dimensional Heaviside function. Formal differentiation yields

$$\frac{d\mu}{d\phi} = (\mu_e-\mu_i)\delta(\phi) = \Big(\sum_{k=1}^{N_e}\beta_k b_k(\mathbf{x}) - \sum_{j=1}^{N_i}\alpha_j a_j(\mathbf{x})\Big)\delta(\phi), \tag{39}$$

where $\delta(\phi)$ is the one-dimensional Dirac delta distribution. Furthermore, equation (38) yields

$$\frac{d\mu}{d\alpha_j} = a_j(\mathbf{x})(1-H(\phi)), \qquad \frac{d\mu}{d\beta_k} = b_k(\mathbf{x})H(\phi). \tag{40}$$

We know from (14) that for a perturbation $\delta\mu$

$$\frac{d\mathcal{J}}{d\mu}\delta\mu = \Big\langle T^*\left(T\mu-g\right),\, \delta\mu\Big\rangle_P. \tag{41}$$

We can now combine the above expressions and obtain from (37)

$$\begin{aligned}\frac{d\mathcal{J}}{dt} &= \Big\langle T^*\left(T\mu-g\right), \Big(\frac{\partial\mu}{\partial\phi}\frac{d\phi}{dt} + \sum_{j=1}^{N_i}\frac{\partial\mu}{\partial\alpha_j}\frac{d\alpha_j}{dt} + \sum_{k=1}^{N_e}\frac{\partial\mu}{\partial\beta_k}\frac{d\beta_k}{dt}\Big)\Big\rangle_P \\ &= \Big\langle T^*\left(T\mu-g\right), \Big((\mu_e-\mu_i)\delta(\phi)f(\mathbf{x},t) + \\ &\quad \sum_{j=1}^{N_i}a_j(\mathbf{x})(1-H(\phi))\hat{f}_j(t) + \sum_{k=1}^{N_e}b_k(\mathbf{x})H(\phi)\tilde{f}_k(t)\Big)\Big\rangle_P.\end{aligned} \tag{42}$$

We use now the formula

$$\delta(\phi) = \frac{\delta_{\partial D}(\mathbf{x})}{|\nabla\phi(\mathbf{x})|}, \tag{43}$$

with $\delta_{\partial D}(\mathbf{x})$ being the Dirac delta distribution concentrated on the boundary of the shape ∂D. Rearranging terms, we can write (42) with (43) in the form

$$\frac{d\mathcal{J}}{dt} = \Big\langle T^*\left(T\mu-g\right),\, (\mu_e-\mu_i)\frac{\delta_{\partial D}(\mathbf{x})}{|\nabla\phi(\mathbf{x})|}f(\mathbf{x},t)\Big\rangle_P \tag{44}$$

$$+ \quad \sum_{j=1}^{N_i} \hat{f}_j(t) \Big\langle T^* \left(T\mu - g\right) \, , \, a_j(\mathbf{x})(1 - H(\phi)) \Big\rangle_P$$

$$+ \quad \sum_{k=1}^{N_e} \tilde{f}_k(t) \Big\langle T^* \left(T\mu - g\right) \, , \, b_k(\mathbf{x}) H(\phi) \Big\rangle_P . \tag{45}$$

We now define the steepest descent directions to be

$$f_{SD}(\mathbf{x}, t) = c_1 T^* \left(T\mu - g\right) (\mu_i - \mu_e) |\nabla \phi(\mathbf{x})| \quad \text{on } \partial D, \tag{46}$$

$$\hat{f}_{j_{SD}} = -c_2 \Big\langle T^* \left(T\mu - g\right) \, , \, a_j(\mathbf{x})(1 - H(\phi)) \Big\rangle_P \text{ for } j = 1, \ldots, N_i, \tag{47}$$

$$\tilde{f}_{k_{SD}} = -c_3 \Big\langle T^* \left(T\mu - g\right) \, , \, b_k(\mathbf{x}) H(\phi) \Big\rangle_P \quad \text{for } k = 1, \ldots, N_e, \tag{48}$$

for some positive constant weighting parameters c_1, c_2 and c_3. Plugging these expressions into (44) will give us a descent flow for the cost $\mathcal{J}$. Using the inner product (10), the evolution equations (36) then acquire the form

$$\frac{d\phi}{dt} + c_1 F_{SD}(\mathbf{x}, t) \cdot |\nabla \phi| = 0, \tag{49}$$

$$\frac{d\alpha_j}{dt} = -c_2 \int_D T^* \left(T\mu - g\right) \, a_j(\mathbf{x}) \, d\mathbf{x}, \tag{50}$$

$$\frac{d\beta_k}{dt} = -c_3 \int_{\Omega \setminus D} T^* \left(T\mu - g\right) \, b_k(\mathbf{x}) \, d\mathbf{x}, \tag{51}$$

where $F_{SD}(\mathbf{x}, t)$ is defined as in (32). Equation (49) is again the Hamilton-Jacobi-type equation already found in (33) for level set function ϕ. Equations (50) and (51) are evolution laws for the individual expansion parameters α_j,$j = 1, \ldots, N_i$, and β_k, $k = 1, \ldots, N_e$. In order to calculate the right-hand sides of the evolution system (49)–(51) in a given time step t, one Radon transform needs to be calculated for the current attenuation distribution $\mu(\mathbf{x}, t)$ in order to calculate $T\mu - g$, and then this result needs to be backprojected by applying T^* in order to calculate $T^* \left(T\mu - g\right)$. Once this expression (which is a function in the space of attenuations) has been calculated, all three right-hand sides of (49)–(51) can be computed from it simultaneously by just calculating weighted integrals of this expression over the individual regions D, ∂D, and $\Omega \setminus D$. We mention that a related approach has been proposed earlier in [21], where numerical results can also be found.

2.9. Geometric Regularization Strategies

As mentioned above, a great advantage of iterative reconstruction techniques is the ease and flexibility with which additional regularization criteria can be incorporated into the inversion. In this section we will present four popular regularization schemes that all aim at restricting geometric properties of the final

shapes. Each of them can be written in the form of a cost functional which is added to least-squares functional (2.3). Recalculating the gradient (or steepest descent) directions for these extended cost functionals then yields additional terms in the normal velocity component. These additional terms will be derived in the following.

2.9.1. Penalizing Total Length of Boundaries

We define the total length (or surface) of Γ as

$$\mathcal{J}_{len\Gamma}(D) \;=\; \int_{\Gamma} d\Gamma \;=\; \int_{\Omega} \delta_{\partial D}(\mathbf{x})\, d\mathbf{x}. \tag{52}$$

It is shown in [22, 23] that applying a flow by a smooth vector field $\mathbf{v}(\mathbf{x})$ yields an infinitesimal response in this cost (52) that is given by

$$d\mathcal{J}_{len\Gamma}(D, \mathbf{v}) \;=\; \int_{\Gamma} \kappa \,\langle \mathbf{v}, \mathbf{n} \rangle \, d\Gamma, \tag{53}$$

where κ denotes the mean curvature:

$$\kappa(\mathbf{x}) \;=\; \nabla \cdot \mathbf{n}(\mathbf{x}) \;=\; \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right), \tag{54}$$

and where $\mathbf{n}$ is the outward normal to the boundary Γ. The relationship (53) can also be derived directly using a level set formulation. First, using (43), we write (52) in the form

$$\mathcal{J}_{len\Gamma}(D(\phi)) \;=\; \int_{\Omega} \delta(\phi) |\nabla \phi(\mathbf{x})|\, d\mathbf{x}. \tag{55}$$

Perturbing now $\phi \rightarrow \phi + \psi$, formal calculation (see, e.g., [14]) yields that the cost functional is perturbed by

$$\left\langle \frac{\partial \mathcal{J}_{len\Gamma}}{\partial \phi} \,,\, \psi \right\rangle \;=\; \int_{\Omega} \delta(\phi) \psi(\mathbf{x}) \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}\, d\mathbf{x}. \tag{56}$$

Therefore, using (54), we can formally identify

$$\frac{\partial \mathcal{J}_{len\Gamma}}{\partial \phi} \;=\; \delta(\phi) \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \;=\; \delta(\phi) \kappa. \tag{57}$$

For both representations (53) and (57), minimizing the cost by a gradient method leads to curvature-driven flow equations, which is $\mathbf{v} = -\kappa \mathbf{n}$.

2.9.2. Penalizing Volume or Area of Shape

We define the total area (volume) of D as

$$\mathcal{J}_{volD}(D) \;=\; \int_D d\mathbf{x} \;=\; \int_\Omega \chi_D(\mathbf{x})\, d\mathbf{x}. \tag{58}$$

It is shown in [22, 23] that applying a flow by a smooth vector field $\mathbf{v}(\mathbf{x})$ yields an infinitesimal response in this cost (58) that is given by

$$d\mathcal{J}_{volD}(D, \mathbf{v}) \;=\; \int_D \mathrm{div}\mathbf{v}\, d\mathbf{x} \;=\; \int_\Gamma \langle \mathbf{v}\,,\, \mathbf{n}\rangle\, d\Gamma. \tag{59}$$

Again, if the shape D is represented by a continuously differentiable level set function ϕ, an alternative derivation can be given. Using again the one-dimensional Heaviside function $H(\phi)$, we can write (58) in the form

$$\mathcal{J}_{volD}(D) \;=\; \int_\Omega H(\phi)\, d\mathbf{x}. \tag{60}$$

Perturbing as before $\phi \rightarrow \phi + \psi$, we get

$$\left\langle \frac{\partial \mathcal{J}_{volD}}{\partial \phi}\,,\, \psi \right\rangle \;=\; \int_\Omega \delta(\phi)\psi(\mathbf{x})\, d\mathbf{x} \tag{61}$$

such that we can identify

$$\frac{\partial \mathcal{J}_{volD}}{\partial \phi} \;=\; \delta(\phi). \tag{62}$$

In both formulations, the steepest-descent flow is given by motion with constant speed in the negative direction of the normal $\mathbf{n}$ to the boundary Γ, which is $\mathbf{v} = -\mathbf{n}$.

2.9.3. Mumford-Shah for Binary Media

A popular regularization functional in image segmentation is the Mumford-Shah functional [24]. In addition to a fidelity term inside each region of the segmented image it contains a term that encourages to shorten total curve-length of the interfaces. We will assume the special form of the attenuation distribution

$$\mu(\mathbf{x}) \;=\; \mu_i(1 - H(\phi)) \;+\; \mu_e H(\phi), \tag{63}$$

where μ_i and μ_e are given constant values. Then, this latter term can be written as

$$\mathcal{J}_{MS} \;=\; \int_\Omega |\nabla H(\phi)|\, d\mathbf{x}. \tag{64}$$

Taking into account that $\nabla H(\phi) \;=\; H^{'}(\phi)\nabla\phi \;=\; \delta(\phi)|\nabla\phi|\mathbf{n}$, we see that $\mathcal{J}_{MS}$ $= \mathcal{J}_{len\Gamma}(D(\phi))$ as given in (55) such that we arrive again at the curvature-driven flow equation (57). For more details we refer to [25, 26].

2.9.4. Total Variation for Binary Media

Another popular cost functional in image segmentation is the total variation (TV) functional, which for our situation of (piecewise constant) binary media (63) reads as

$$\mathcal{J}_{TV} \;=\; \int_\Omega |\nabla \mu(\phi)|\, d\mathbf{x} \;=\; |\mu_e - \mu_i| \int_\Omega |\nabla H(\phi)|\, d\mathbf{x}. \tag{65}$$

It coincides with the Mumford-Shah functional $\mathcal{J}_{MS}$ up to the factor $|\mu_e - \mu_i|$. Roughly we can say that the TV functional (65) penalizes the product of the jump between different regions and the arc length of their interfaces, whereas the Mumford-Shah functional (64) penalizes only this arc length. We refer for more information to [27].

2.10. Further Linear Inverse Problems in Medical Imaging

The theory discussed so far is fairly general and not restricted to the special application of x-ray CT. It can be applied to a broad class of linear inverse problems. Prominent examples for linear inverse problems in medical imaging are Single Photon Emission Computerized Tomography (SPECT), Positron Emission Tomography (PET), Magnetic Resonance Imaging (MRI), or Electron Microscopy. Several of the mentioned examples have been addressed in the literature using a level set technique. Examples are the works [4, 6, 21, 28–30], without claim of exhaustivity of this small list of references. Certainly, each of these applications has its peculiarities. For more details on how to address these, we refer to the given references.

3. LEVEL SET TECHNIQUES FOR NONLINEAR INVERSE PROBLEMS

In this section we describe the more general approach of level set techniques for nonlinear inverse problems in medical imaging. As a good example for a nonlinear inverse problem in medical imaging we will first describe the mathematical model for Diffuse Optical Tomography (DOT), which in some sense can be considered a generalization of x-ray CT to situations where scattering of photons becomes dominant. It leads to a nonlinear inverse problem since the linearization operation (6) does not yield simplified expressions anymore due to the scattering. Instead, so-called 'linearized sensitivity functions' need to be considered now, which correspond to linearizations (often called Jacobians or, in the continuous setting, Fréchet derivatives,) of the nonlinear forward problem. Physically, these linearized sensitivity functions have a similar meaning as the line integrals in x-ray tomography, indicating the information flow from events inside the domain toward the detectors. In other words, they replace the line integrals in (6) by weighted integrals over the whole domain with the weights indicating importance of a given region to the measurements. We will describe them below. Furthermore, the

parameter-based (or pixel-based) inversion algorithms are iterative, instead of being numerical implementations of explicit inversion formulas as it was the case in the filtered backprojection technique. Analogous nonlinear inversion techniques can be found in the applications of ultrasound tomography, microwave medical imaging, or electrical impedance tomography as pointed to in Section 3.5. In the following section, we will describe the basic model of DOT.

3.1. Example: Diffuse Optical Tomography (DOT)

The propagation of photons in tissue is modeled by the time-dependent radiative transfer equation (or 'linear transport equation' [16])

$$\frac{1}{c}\frac{\partial u}{\partial t} + \theta \cdot \nabla u(\mathbf{x},\theta,t) + (a(\mathbf{x}) + b(\mathbf{x}))u(\mathbf{x},\theta,t)$$

$$- b(\mathbf{x}) \int_{S^{n-1}} \eta(\theta \cdot \theta')u(\mathbf{x},\theta',t)d\theta' = q(\mathbf{x},\theta,t) \tag{66}$$

in $\Omega \times S^{n-1} \times [0,T]$ with initial condition

$$u(\mathbf{x},\theta,0) = 0 \qquad \text{in} \quad \Omega \times S^{n-1} \tag{67}$$

and boundary condition

$$u(\mathbf{x},\theta,t) = 0 \qquad \text{on} \quad \Gamma_-. \tag{68}$$

Here,

$$\Gamma_\pm := \left\{(\mathbf{x},\theta,t) \in \partial\Omega \times S^{n-1} \times [0,T], \quad \pm\mathbf{n}(\mathbf{x}) \cdot \theta > 0\right\}.$$

Ω is a convex, compact domain in $\mathbb{R}^n$, $n = 2, 3$, with smooth boundary $\partial\Omega$. In our numerical experiments, we will only consider the case $n = 2$, but the algorithm extends in a straightforward way to $n = 3$. $\mathbf{n}(\mathbf{x})$ denotes the outward unit normal to $\partial\Omega$ at the point $\mathbf{x} \in \partial\Omega$, and $u(\mathbf{x},\theta,t)$ describes the density of particles (photons) which travel in Ω at time t through the point $\mathbf{x}$ in the direction θ. The velocity c of the particles is assumed to be constant, which is typically a good approximation in optical tomography, where photons are assumed not to lose energy in scattering events (elastic scattering).

$a(\mathbf{x})$ is the absorption cross-section (in short 'absorption'), $b(\mathbf{x})$ is the scattering cross-section, and $\mu(\mathbf{x}) := a(\mathbf{x}) + b(\mathbf{x})$ is the total cross-section or attenuation as it was already encountered in the application of x-ray CT (even though for a different wavelength of the propagating energy). These parameters are assumed to be real, strictly positive functions of the position $\mathbf{x}$. The quantity μ^{-1} is the mean free path of the photons. Typical values in DOT are $a \approx 0.1 - 1.0\,\mathrm{cm}^{-1}$,

$b \approx 100 - 200\,\mathrm{cm}^{-1}$, $\mu^{-1} \approx 0.005 - 0.01\,\mathrm{cm}$ [31, 32]. The scattering function $\eta(\theta \cdot \theta')$ describes the probability for a particle entering a scattering process with the direction of propagation θ' to leave this process with the direction θ. It is normalized to

$$\int_{S^{n-1}} \eta(\theta \cdot \theta') d\theta \;=\; 1, \tag{69}$$

which expresses particle conservation in pure scattering events. The dot-product in the argument indicates that η depends only on the cosine of the scattering angle $\cos\vartheta = \theta \cdot \theta'$, an assumption typically made in DOT. Another assumption typically made in DOT is that η is independent of the position $\mathbf{x}$, although the theory developed in the following can easily be generalized to the more general case of position-dependent scattering functions. In our numerical experiments, we will use a 2D-adapted version of the following Henyey-Greenstein scattering function:

$$\eta(\theta \cdot \theta') \;=\; \frac{1}{4\pi} \frac{1-\gamma^2}{(1+\gamma^2-2\gamma\cos\vartheta)^{3/2}}, \tag{70}$$

with $-1 < \gamma < 1$. (See, for example, [33, 34] for possible choices.) The parameter γ in (70) is the mean cosine of the scattering function. Values of γ close to one indicate that the scattering is primarily forward directed, whereas values close to zero indicate that scattering is almost isotropic. In our numerical experiments we will choose γ to be 0.9, which is a typical value for DOT.

The initial condition (67) indicates that there are no photons moving inside Ω at the starting time of our experiment. The boundary condition (68) indicates that during the experiment no photons enter the domain Ω from the outside. All photons inside Ω originate from the source q, which, however, can be situated at the boundary $\partial\Omega$.

We consider the problem (66)–(68) for p different sources $q_j,\ j = 1, \ldots, p$, positioned at $\partial\Omega$. Possible sources in applications are delta-like pulses transmitted at time $t = 0$ at the position $\mathbf{s}_j \in \partial\Omega$ into the direction θ_j, which can be described by the distributional expressions

$$\tilde{q}_j(\mathbf{x}, \theta, t) \;=\; \delta_{\mathbf{s}_j}(\mathbf{x})\delta(t)\delta(\theta - \theta_j), \quad j = 1, \ldots, p, \tag{71}$$

where $\delta_{\mathbf{s}_j}(\mathbf{x})$ is a Dirac delta distribution concentrated on a small part of the boundary $\partial\Omega$ indicating the source extension, and where $\mathbf{n}(\mathbf{s}_j) \cdot \theta_j < 0$ along this part of the boundary. We assume that a given source q_j gives rise to the physical fields $\tilde{u}_j(\mathbf{x}, \theta, t)$, which are solutions of (66)–(68). Our measurements consist of the outgoing flux across the boundary $\partial\Omega$, which has with (68) the form

$$g_j(\mathbf{x}, t) \;=\; \int_{\mathbf{n}(\mathbf{x})\cdot\theta>0} \mathbf{n}(\mathbf{x}) \cdot \theta \tilde{u}_j(\mathbf{x}, \theta, t) d\theta \quad \text{on} \quad \partial\Omega \times [0, T] \tag{72}$$

for $j = 1, \ldots, p$. We will assume in the following that we know the scattering functions η and b, and we want to reconstruct the coefficient $a(\mathbf{x})$ from the data.

The theory for the more general case of more than one unknown function then follows the same line of reasoning, see, e.g., [31, 33, 35].

Let us call the data corresponding to the parameter distribution a by $\mathcal{A}(a)$. In contrast to the Radon transform T considered in the previous section, the operator $\mathcal{A}$ is now a nonlinear operator, mapping parameter distributions $a(\mathbf{x})$ (here the absorption coefficient) to the corresponding data $\mathcal{A}(a)$. Let us furthermore call the physically measured data by g. As before, we can consider the difference between these two and define the (now as well nonlinear) residual operator

$$\mathcal{R}(a) \;=\; \mathcal{A}(a) - g. \tag{73}$$

The goal will be to minimize this mismatch between calculated and measured data in some appropriate sense, as is described in the following section.

3.2. A Gradient Technique for DOT

We define the least-square cost functional

$$\mathcal{J}(a) \;=\; \frac{1}{2}\big\|\mathcal{R}(a)\big\|_Z^2 \;=\; \frac{1}{2}\Big\langle\, \mathcal{R}(a)\,,\, \mathcal{R}(a)\,\Big\rangle_Z, \tag{74}$$

where $\langle\ ,\ \rangle_Z$ denotes the canonical inner product in data space Z. We assume that $\mathcal{R}(a)$ admits the expansion

$$\mathcal{R}(a+\delta a) \;=\; \mathcal{R}(a) \;+\; \mathcal{R}'(a)\delta a \;+\; O(\|\delta a\|_P^2), \tag{75}$$

letting $\|\ \|_P$ be the canonical norm in parameter space P, for a sufficiently small perturbation (variation) $\delta a \in P$. The linear operator $\mathcal{R}'(a)$ is often called the *Fréchet derivative* of $\mathcal{R}$. Plugging (75) into (74) yields the relationship

$$\mathcal{J}(a+\delta a) \;=\; \mathcal{J}(a) \;+\; \Big\langle \mathcal{R}'(a)^*\mathcal{R}(a)\,,\, \delta a\Big\rangle_P \;+\; O(\|\delta a\|_P^2). \tag{76}$$

The operator $\mathcal{R}'(a)^*$ is the formal adjoint operator of $\mathcal{R}'(a)$ with respect to spaces Z and P:

$$\Big\langle \mathcal{R}'(a)^* g\,,\, \hat{a}\Big\rangle_P \;=\; \Big\langle g\,,\, \mathcal{R}'(a)\hat{a}\Big\rangle_Z \quad \text{for all } \hat{a}\in P,\ g\in Z. \tag{77}$$

We call

$$\mathbf{grad}_{\mathcal{J}}(a) \;=\; \mathcal{R}'(a)^*\mathcal{R}(a) \tag{78}$$

the *gradient direction* of $\mathcal{J}$ in a.

The remaining part of the theory is now very similar to the development presented in the previous section for linear operators. We only mention here the result analogous to (30):

$$\left.\frac{\partial \mathcal{J}(a)}{\partial t}\right|_{t=0} \;=\; \int_{\partial D}\Big[\mathcal{R}'(a)^*\mathcal{R}(a)\Big](a_i - a_e)\mathbf{v}(\mathbf{x})\cdot\mathbf{n}(\mathbf{x})ds(\mathbf{x}), \tag{79}$$

which describes the change in the cost due to a small deformation of the shape D by a vector field $\mathbf{v}(\mathbf{x})$, and the corresponding gradient (or steepest-descent) normal velocity field F_{SD}, which is

$$F_{SD}(\mathbf{x}) \;=\; -\mathcal{R}'(a)^*\mathcal{R}(a)(a_i - a_e) \quad \text{at } \partial D. \tag{80}$$

Plugging this into (79) gives us a descent direction for the cost $\mathcal{J}$.

The remainder of the theory, concerning for example the simultaneous reconstruction of shape and texture components, or the additional incorporation of geometric regularization schemes, is now completely analogous to the linear case, such that we will not discuss it here in particular. The main difference to the linear theory is the need for computing the adjoint linearized residual operators $\mathcal{R}'(a)^*$ in each step of the iterative inversion. We will address this important topic in the following section.

3.3. Adjoint Formulation for Calculating Gradient Directions

We have seen in the previous section that the linearized residual operator $\mathcal{R}'(a)$ and its adjoint $\mathcal{R}'(a)^*$ are key ingredients for gradient-based inversion routines. Typically, it is much too expensive to calculate the adjoint linearized residual operator $\mathcal{R}'(a)^*$ explicitly in each step of the iteration. On the other hand, this is not really necessary, since only its action on the current residuals $\mathcal{R}'(a)^*\mathcal{R}(a)$ is needed in order to find the steepest-descent (or related gradient-based) search directions. For this, a so-called 'adjoint scheme' has become quite popular due to its efficiency in solving this task. It only needs the solution of one forward and one adjoint problem in order to determine the gradient direction. In the following, we will show how this can be achieved for our example of DOT by the transport equation. Many other imaging techniques follow the same scheme. We will restrict ourselves, as before, to the case of determining only the absorption parameter $a(\mathbf{x})$. The scattering parameter $b(\mathbf{x})$ is assumed to be known.

A possible way of deriving explicit forms for the linearized residual operator $\mathcal{R}'(a)$ is to perturb the parameter functions $a(\mathbf{x}) \to a(\mathbf{x}) + \delta a(\mathbf{x})$ and plug this into (66)–(68). Assuming that the corresponding solution $u(\mathbf{x})$ of (66)–(68) responds to this perturbation according to $u(\mathbf{x}) \to u(\mathbf{x}) + w(\mathbf{x})$, and neglecting all terms in (1) which are of higher than linear order in δa and w, we arrive at the following expression for the linearized residual operator $\mathcal{R}'(a)$:

$$\mathcal{R}'(a)(\delta a)(\mathbf{x}_r, t_r) \;=\; \int_{S_+^{n-1}} \mathbf{n}(\mathbf{x}_r) \cdot \theta w(\mathbf{x}_r, \theta, t_r)\, d\theta, \tag{81}$$

where w solves the linearized equation

$$\begin{aligned}\frac{\partial w}{\partial t} &+ \theta \cdot \nabla w(\mathbf{x},\theta,t) + (a(\mathbf{x})+b(\mathbf{x}))w(\mathbf{x},\theta,t) - b(\mathbf{x}) \\ &\int_{S^{n-1}} \eta(\theta\cdot\theta')w(\mathbf{x},\theta',t)d\theta' \\ &= -\delta a(\mathbf{x})u(\mathbf{x},\theta,t) \end{aligned} \tag{82}$$

in $\Omega \times S^{n-1} \times [0,T]$ with the homogeneous initial condition

$$w(\mathbf{x},\theta,0) = 0 \qquad \text{in} \quad \Omega \times S^{n-1},$$

and an absorbing boundary condition

$$w(\mathbf{x},\theta,t) = 0 \qquad \text{on} \quad \Gamma_{-}. \tag{83}$$

Notice that, for a given perturbation function $\delta a(\mathbf{x}_{sc})$ (where the argument $\mathbf{x}_{sc} \in \Omega$ denotes the scattering points), the value of $\mathcal{R}'(a)\delta a$ is a function in the variables $\mathbf{x}_r$ and t_r, where $\mathbf{x}_r$ is the receiver location and t_r the receiver time. This explains the somewhat complicated notation in (81). The physical interpretation of this result is that the perturbation δa creates a scattering source $Q_{\delta a}(\mathbf{x},\theta,t) = -\delta a(\mathbf{x})u(\mathbf{x},\theta,t)$ inside the domain Ω. This gives rise to a distribution $w(\mathbf{x},\theta,t)$ (which can be positive or negative) of virtual 'secondary particles' propagating in the unperturbed medium to the receivers, where they are detected as the (linearized) residuals in the data.

The adjoint linearized residual operator $\mathcal{R}'(a)^*$ is formally defined by the identity (77). It is a mapping from the data space Z into the parameter space P. Notice the analogy to the adjoint Radon transform in linear x-ray tomography, which is as well a backprojection operator from the data space into the parameter space. We will see below that there is indeed a close analogy between the backprojection operator in x-ray tomography and the adjoint linearized residual operator in DOT.

An explicit expression for the action of the adjoint linearized residual operator on an arbitrary vector ζ of the data space can be derived by using Green's formula for the linear transport equation in time domain. Next, we formulate this result, and refer for a derivation to [36, 33].

Let z denote the solution of the following adjoint linear transport equation:

$$\begin{aligned}-\frac{\partial z}{\partial t} &- \theta \cdot \nabla z(\mathbf{x},\theta,t) + (a(\mathbf{x})+b(\mathbf{x}))z(\mathbf{x},\theta,t) - b(\mathbf{x}) \\ &\int_{S^{n-1}} \eta(\theta\cdot\theta')z(\mathbf{x},\theta',t)d\theta' \\ &= 0 \quad \text{in } \Omega \times S^{n-1} \times [0,T], \end{aligned}$$

with the 'final value condition'

$$z(\mathbf{x},\theta,T) = 0 \qquad \text{in} \quad \Omega \times S^{n-1}, \tag{84}$$

and the *inhomogeneous* outgoing boundary condition

$$z(\mathbf{x},\theta,t) = \zeta(\mathbf{x},t) \quad \text{uniformly in } \theta \text{ on } \Gamma_+, \tag{85}$$

and let u be the solution of the forward problem (66)–(68). Then we have

$$\mathcal{R}'(a)^*\zeta(\mathbf{x}) \;=\; -\int_{[0,T]}\int_{S^{n-1}} u(\mathbf{x},\theta,t) z(\mathbf{x},\theta,t)\, d\theta dt\,. \tag{86}$$

Solving (84)–(85) is called 'backtransport.'

We mention that the adjoint linearized residual operator admits an interesting physical interpretation. The argument ζ of this operator is an element of the data space Z, i.e., it is a function of detector position $\mathbf{x}_r$ and detection time t_r. The values of ζ, which in our applications will be the residuals, are attached as time-dependent adjoint sources at these receiver positions in (85), and transported backward in direction and in time into the medium Ω by solving (84)–(85). Notice the sign change in front of the time derivative ∂_t and the space derivative $\theta \cdot \nabla$ in (84) compared to (66), which indicates a reversed time and direction. This is also the reason why we have used a final value condition (84) and an 'outgoing flux condition' (85) for uniquely specifying the solution of (84). Notice also that the residuals ζ are applied uniformly in all directions in (84), which accounts for the averaging θ-integration of the measurement process (72).

Loosely speaking, these virtual 'adjoint particles' are propagating backward into the domain, retracing as much as possible the paths of those particles that have been created by the parameter perturbation δa and which have caused the mismatch in the data. The gradient direction (which is essentially $\mathcal{R}'(a)^*\zeta$) is then calculated by combining these backpropagated densities with the actual densities u of our forward problem (66)–(68). For example, if no particles of the forward solution reach a given point $\mathbf{x}_{sc}$ during the experiment, i.e., $u(\mathbf{x}_{sc},\theta,t) = 0$ for all $\theta \in S^{n-1}$ and all $t \in [0,T]$, then the update $\mathcal{R}'(a)^*\zeta(\mathbf{x}_{sc})$ will be zero at this location $\mathbf{x}_{sc}$. This makes sense physically, since no secondary particles could have been generated at this point in the medium by a parameter change $\delta a(\mathbf{x}_{sc})$ and reach the detector via (82). This means also that the 'sensitivity' of our source-receiver pair to parameter changes at this location will be zero. This observation motivates the introduction of (linearized) 'sensitivity functions,' which quantify the sensitivity of a given source-receiver pair to parameter perturbations at each point $\mathbf{x}_{sc}$ in the medium. These linearized sensitivity functions are calculated by putting a source at time zero and source position $\mathbf{x}_s$ and solving the forward problem (66)–(68). At the same time, a fictitious adjoint source is placed at the receiver position $\mathbf{x}_r$ and receiver time t_r, and the adjoint transport equation (84)–(85) is solved for this ficticious adjoint source. The two results are then combined evaluating the expression (86).

It can be shown formally (see, e.g., [37]) that the so-obtained linearized sensitivity functions have the same physical meaning as the lines of propagation have in

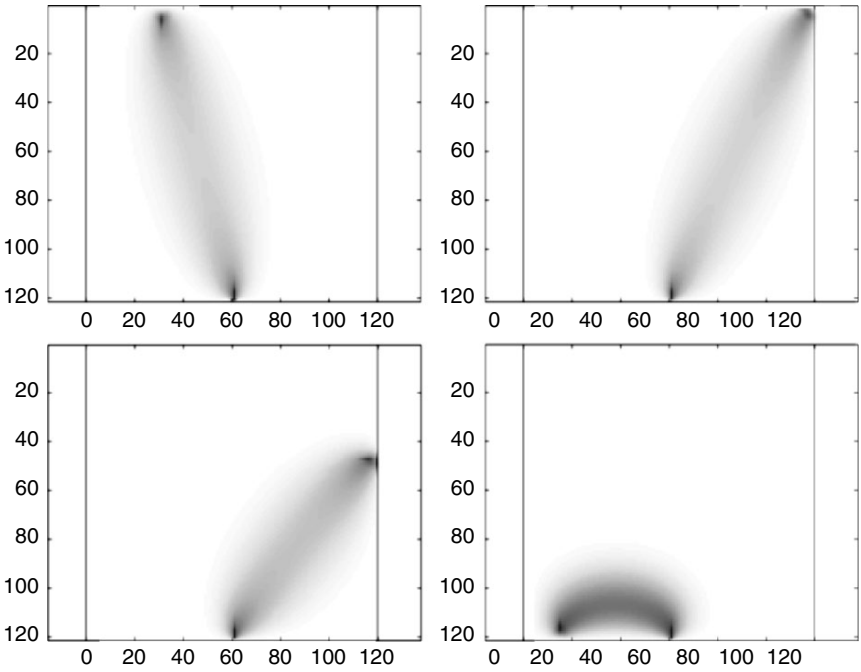


Figure 3. Numerically calculated sensitivity functions for four different source–receiver pairs and a fixed time step in diffuse optical tomography. Homogeneous background with $a = 0.1\,\mathrm{cm}^{-1}$, $b = 100\,\mathrm{cm}^{-1}$, and Henyey Greenstein scattering with $\gamma = 0.9$.

x-ray tomography. They indicate the 'importance' or 'sensitivity' of a given point in the domain to the specified source and receiver coordinates. Application of the adjoint linearized residual operator $\mathcal{R}'(a)^*$ to the current mismatch in the data $\mathcal{R}(a)$ amounts to 'backprojecting' these residuals along the linearized sensitivity functions. Instead of smearing them uniformly over lines, as it was the case in x-ray tomography, they are now distributed over the whole imaging domain with position-dependent weights given by these linearized sensitivity functions. We have displayed a few of these linearized sensitivity functions for one given time step in Figure 3. It can be seen that the lines of x-ray tomography are now replaced by 'clouds,' often called 'banana-shapes.' The examples shown in Figure 3 have been calculated for a homogeneous parameter background distribution. Their forms will change when changing this background distribution. This is the reason why the operator $\mathcal{R}'(a)^*$ needs to be recalculated in each step of the nonlinear iterative reconstruction algorithm.

3.4. Some Numerical Experiments for DOT

Figures 4–8 show the results of some numerical experiments for shape reconstruction in DOT. We model the human head for simplicity as a two-dimensional square domain of size $5 \times 5\,\mathrm{cm}^2$. It is composed of a background material with three different clear regions embedded. The clear regions have parameters $a = 0.01$ cm^{-1} and $b = 0.01$ cm^{-1}. One of these three embedded clear regions has a band-like structure parallel to the boundary, symbolically representing a layer of cerebrospinal fluid beneath the skull. The other two clear regions are small pockets of liquid located in the interior of the head. The background material (excluding these clear regions) is assumed to have a homogeneous absorption parameter value of $a = 0.1\,\mathrm{cm}^{-1}$ and a homogeneous scattering parameter of $b = 100\,\mathrm{cm}^{-1}$. These are typical values for clinical applications in DOT.

Along the boundary, we have indicated in Figures 4–7 the 16 source positions by dark bars having the extensions of these sources. At these positions, successively ultrashort laser pulses are emitted perpendicular to the boundary into the head. The measurements for each of these source positions consist of the time-resolved photon flux across the boundary (see (72)). The bottom right images of Figures 4–7 show the true objects that we want to reconstruct from the boundary data. In particular, in these images the two dark regions in the interior of the head symbolize the objects of interest. They might represent regions of blood accumulation (e.g., hematoma) or similar objects.

For our numerical experiments, we use a finite-difference discretization of the radiative transfer equations (66)–(68) and (84)–(85) as described in [33]. The whole domain is discretized into 50×50 pixels, and we monitor 100 discrete time steps of the propagating photons. 12 different directions are used for discretizing the angular variable. As scattering law, we use the Henyey-Greenstein function (70) with $\gamma = 0.9$. We run our finite-difference code on the above described model (which we call the 'true model') in order to simulate our data. Then we use these simulated data as input for our shape reconstruction method.

In our numerical experiments we compare the shape evolution for four different starting guesses. These initial guesses are displayed in the upper left images of Figures 4–7. The remaining images show the corresponding shape evolutions, as explained in the captions to these figures. As initial level set functions for these examples we use the so-called 'signed distance function' (see [2, 3]) representing the given initial shape. No reinitialization is applied during the evolution of the level set function. However, in each step it is rescaled such that its global minimum remains at a constant value. Figure 7 shows the behavior of the least-square data misfit during 500 steps of the shape evolution for these four examples. It is clearly visible that the speed of convergence, as well as its characteristic behavior, strongly depends on the initial guess. Nevertheless, all examples converge to the correct final shape without problems. Topological changes occur in all cases, and are modeled automatically by the level set formulation.

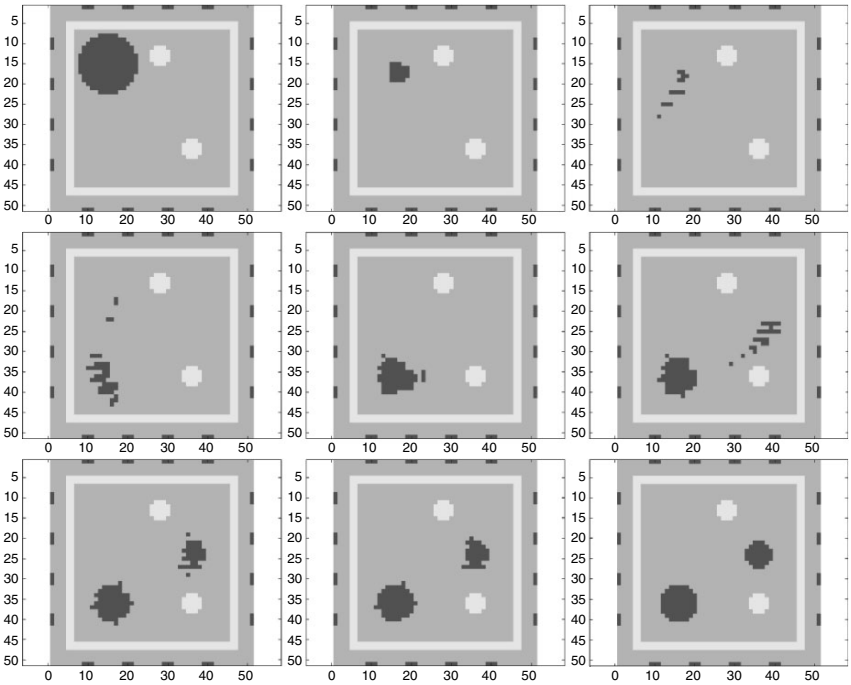


Figure 4. Shape evolution for the first example. Top row: first guess, after 2 and 14 iterations; center row: after 20, 80, and 130 iterations; bottom row: after 250 and 500 iterations; bottom right: true reference model. See also animated movie DOTmovie1 on the attached CD.

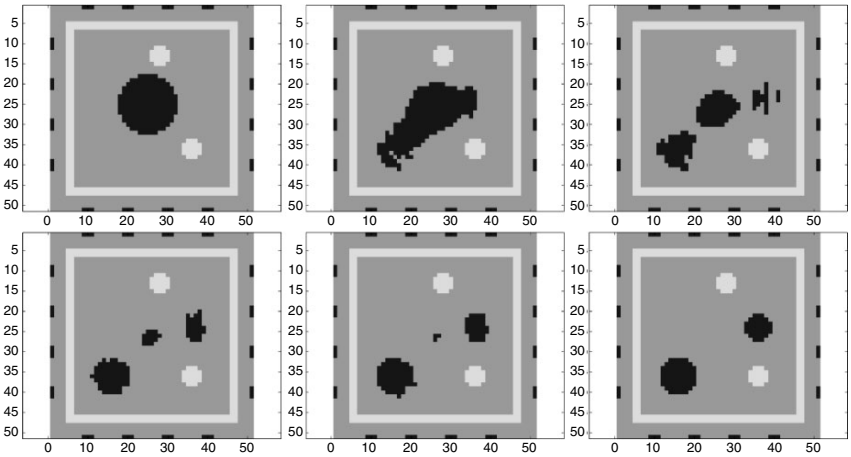


Figure 5. Shape evolution for the second example. Top row: first guess, after 10 and 40 iterations; bottom row: after 250 and 500 iterations; bottom right: true reference model. See also animated movie DOTmovie2 on the attached CD.

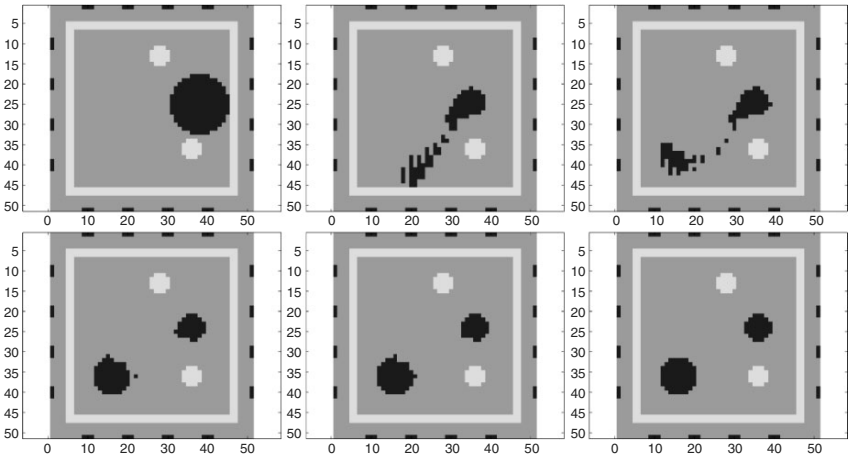


Figure 6. Shape evolution for the third example. Top row: first guess, after 30 and 35 iterations; bottom row: after 250 and 500 iterations; bottom right: true reference model. See also animated movie DOTmovie3 on the attached CD.

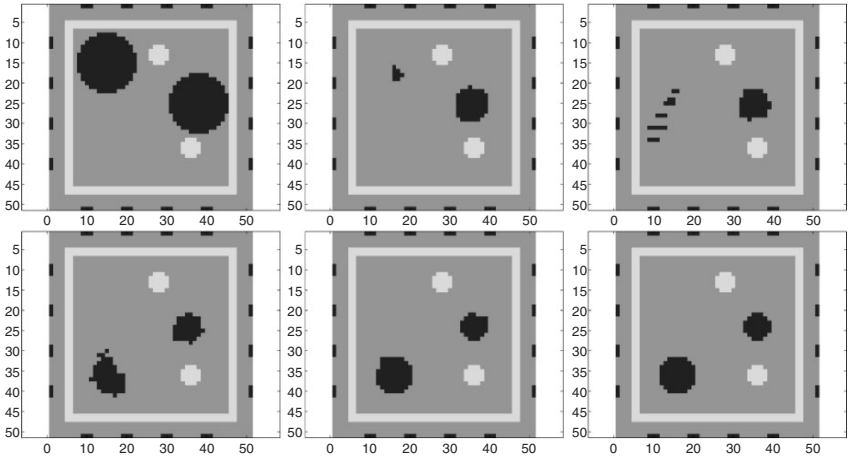


Figure 7. Shape evolution for the fourth example. Top row: first guess, after 4 and 10 iterations; bottom row: after 40 and 500 iterations; bottom right: true reference model. See also animated movie DOTmovie4 on the attached CD.

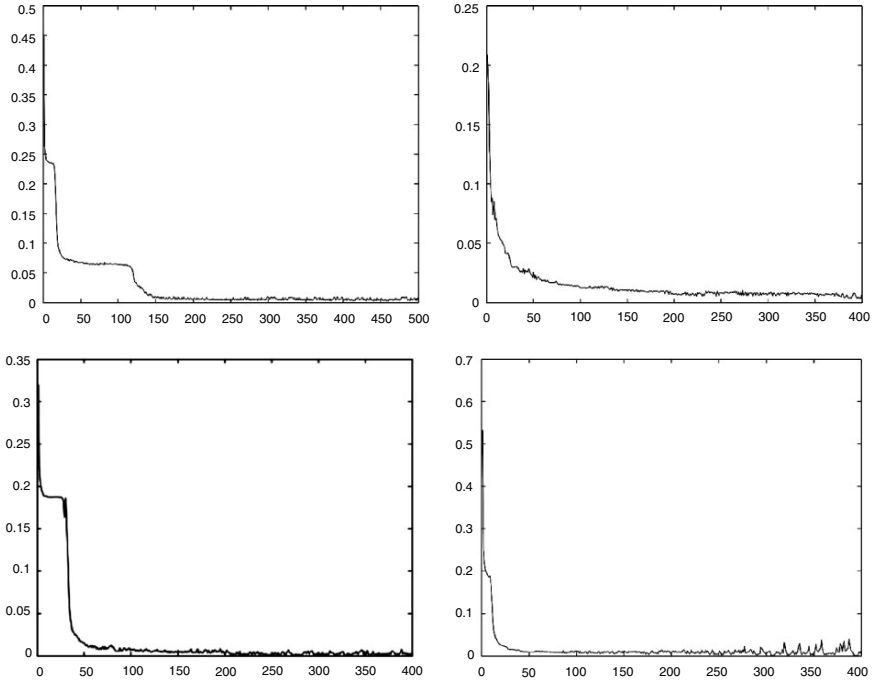


Figure 8. Evolution of least-squares data misfit. Top left: first example (Fig. 4); top right: second example (Fig. 5); bottom left: third example (Fig. 6); bottom right: fourth example (Fig. 7).

3.5. Other Examples for Nonlinear Inverse Problems

The theory presented in this section is again not restricted to special application of DOT, but is intended to be representative for a broader class of nonlinear inverse problems in medical imaging. Examples are microwave medical imaging [5, 7, 11, 38–43], electrical impedance tomography [9, 12, 14, 27, 44–48], and electrocardiography [49, 50]. Further examples for results discussed in diffuse optical tomography can be found in [35, 51–53]. Again, we cannot claim exhaustivity of this short list of references.

4. ACKNOWLEDGMENTS

O.D. gratefully acknowledges support of this work by the European Union through the Sixth Framework Programme as part of the project on 'Integrated technologies for in-vivo molecular imaging,' contract No. LSHG-CT-2003-503259.

5. REFERENCES

1. Osher S, Sethian JA. 1988. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J Comput Phys* **79**:12–49.
2. Osher S, Fedkiw R. 2003. *Level set methods and dynamic implicit surfaces.* New York: Springer.
3. Sethian JA. 1999. *Level set methods and fast marching methods*, 2nd ed. Cambridge: Cambridge UP.
4. Santosa F. 1996. A level set approach for inverse problems involving obstacles. *ESAIM Control Optim Calculus Variations* **1**:17–33.
5. Litman A, Lesselier D, Santosa D. 1998. Reconstruction of a two-dimensional binary obstacle by controlled evolution of a level set. *Inverse Probl* **14**:685–706.
6. Feng H, Karl WC, Castanon D. 2000. Tomographic reconstruction using curve evolution. *Proc IEEE Int Conf Computer Vision and Pattern Recognition* **1**:361–366.
7. Dorn O, Miller E, Rappaport C. 2000. A shape reconstruction method for electromagnetic tomography using adjoint fields and level sets. *Inverse Probl* **16**:1119–1156.
8. Burger M. 2001. A level set method for inverse problems. *Inverse Probl* **17**:1327–1355.
9. Ito K, Kunisch K, Li Z. 2001. Level set approach to an inverse interface problem. *Inverse Probl* **17**:1225–1242.
10. Ramananjaona C, Lambert M, Lesselier D, Zolésio J-P. 2001. Shape reconstruction of buried obstacles by controlled evolution of a level set: from a min–max formulation to numerical experimentation. *Inverse Probl* **17**: 1087–1111.
11. Ramananjaona C, Lambert M, Lesselier D. 2001. Shape inversion from TM and TE real data by controlled evolution of level sets. *Inverse Probl* **17**:1585–1595.
12. Burger M, Osher S. 2005. A survey on level set methods for inverse problems and optimal design. *Eur J Appl Math* **16**:263-301.
13. Dorn O, Lesselier D. 2006. Level set methods for inverse scattering, *Inverse Probl* 22:R67–R131.
14. Tai X-C, Chan TF. 2004. A survey on multiple level set methods with applications for identifying piecewise constant functions *Int J Num Anal Model* **1**:25–47.
15. Suri JS, Liu K, Singh S, Laxminarayan SN, Zeng X, Reden L. 2002. Shape recovery algorithms using level sets in 2D/3D medical imagery: a state-of-the-art review *IEEE Trans Inf Technol Biomed* **6**:8–28.
16. Case K, Zweifel P. 1967. *Linear transport theory*. New York: Addison Wesley.
17. Kak AC, Slaney M. 2001. *Principles of computerized tomographic imaging*. SIAM Classics in Applied Mathematics, Vol. 33. Philadelphia: SIAM.
18. Natterer F. 1986. *The mathematics of computerized tomography*. Stuttgart: Teubner.
19. Natterer F, Wübbeling F. 2001. *Mathematical methods in image reconstruction*. Monographs on Mathematical Modeling and Computation, Vol. 5. Philadelphia: SIAM.
20. Radon J. 1917. Über die bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten. *Ber Säch Akad der Wiss Leipzig, Math-Phys Kl* **69**:262–267.
21. Feng H, Karl WC, Castanon DA. 2003. A curve evolution approach to object-based tomographic reconstruction. *IEEE Trans Image Process* **12**:44–57.
22. Delfour MC, Zolésio J-P. 2001. *Shapes and geometries: analysis, differential calculus and optimization*. SIAM Advances in Design and Control. Philadelphia: SIAM.
23. Sokolowski J, Zolésio J-P. 1992. *Introduction to shape optimization: shape sensitivity analysis*. Springer series in Computational Mathematics, Vol. 16. Berlin: Springer.
24. Mumford D, Shah J. 1989. Optimal approximation by piecewise smooth functions and associated variational problems. *Commun Pure Appl Math* **42**:577–685.
25. Chan TF, Vese LA. 2001. Active contours without edges. *IEEE Trans Image Process* **10**:266–277.
26. Vese LA, Chan TF. 2002. A multiphase level set framework for image segmentation using the Mumford-Shah model. *Int J Comput Vision* **50** 271–293.

27. Chan TF, Tai X-C. 2003. Level set and total variation regularization for elliptic inverse problems with discontinuous coefficients. *J Comput Phys* **193**:40–66.
28. Lysaker M, Chan T, Tai X-C. 2004. *Level set method for positron emission tomography*. UCLA-CAM Preprint 04-30.
29. Whitaker RT, Elangovan V. 2002. A direct approach to estimating surfaces in tomographic data. *Med Imaging Anal* **6**, 235–249.
30. Ye JC, Bresler Y, Moulin P. 2002. A self-referencing level set method for image reconstruction from sparse Fourier samples. *Int J Comput Vision* **50**:253–270.
31. Arridge SR. 1999. Optical tomography in medical imaging. *Inverse Probl* **15**:R41–R93.
32. Okada E, Firbank M, Schweiger M, Arridge SR, Cope M, Delpy DT. 1997. Theoretical and experimental investigation of near-infrared light propagation in a model of the adult head. *Appl Opt* **36**(1):21–31.
33. Dorn O. 1998. A transport-backtransport method for optical tomography. *Inverse Probl* **14**:1107–1130.
34. Heino J, Arridge S, Sikora J, Somersalo E. 2003. Anisotropic effects in highly scattering media. *Phys Rev E*, **68**(3):031908-1–031908-8.
35. Schweiger M, Arridge SR, Dorn O, Zacharopoulos A, Kolehmainen V. 2006. Reconstructing absorption and diffusion shape profiles in optical tomography using a level set technique. *Opt Lett* **31**(4):471–473.
36. Dierkes T, Dorn O, Natterer F, Palamodov V, Sielschott H. 2002. Frechet derivatives for some bilinear inverse problems *SIAM J Appl Math* **62**:2092–2113.
37. Dorn O. 2000. Scattering and absorption transport sensitivity functions for optical tomography. *Opt Express* **7**:492–506.
38. Dorn O, Miller E, Rappaport C. 2001. Shape reconstruction in 2D from limited-view multifrequency electromagnetic data. *Radon transform and tomography*. AMS series on Contemporary Mathematics, Vol. 278, pp. 97–122.
39. Ferrayé R, Dauvignac J-Y, Pichot C. 2003. An inverse scattering method based on contour deformations by means of a level set method using frequency hopping technique. *IEEE Trans Antennas Propag* **51**:1100–1113.
40. Ferrayé R, Dauvignac J-Y, Pichot C. 2003. Reconstruction of complex and multiple shape object contours using a level set method *J Electromagn Waves Appl* **17**:153-181.
41. Irishina N, Moscoso M, Dorn O. 2006. Detection of small tumors in microwave medical imaging using level sets and MUSIC. *Proceedings of the progress in electromagnetics research symposium*, Cambridge, MA, March 26–29, 2006. To appear. http://ceta.mit.edu/PIER/.
42. Litman A. 2005. Reconstruction by level sets of n-ary scattering obstacles. *Inverse Probl* **21**:S131–S152.
43. Ramananjaona C, Lambert M, Lesselier D, Zolesio J-P. 2003. On novel developments of controlled evolution of level sets in the field of inverse shape problems. *Radio Sci* **38**:1–9.
44. Ascher UM, Van den Doel K. 2005. On level set regularization for highly ill-posed distributed parameter estimation problems. *J Comput Phys*. To appear. http://www.cs.ubc.ca/ kvdoel/publications/keesUri05.pdf
45. Chung ET, Chan TF, Tai XC. 2005. Electrical impedance tomography using level set representation and total variational regularization. *J Comput Phys* **205**:357–372.
46. Leitao A, Scherzer O. 2003. On the relation between constraint regularization, level sets and shape optimization. *Inverse Probl* **19**:L1–L11.
47. Soleimani M, Lionheart WRB, Dorn O. 2005. Level set reconstruction of conductivity and permittivity from boundary electrical measurements using experimental data. *Inverse Probl Sci Eng* **14**:193–210.
48. Soleimani M, Dorn O, Lionheart WRB. 2006. A narrowband level set method applied to EIT in brain for cryosurgery monitoring. *IEEE Trans Biomed Eng*. To appear.

49. Calderero F, Ghodrati A, Brooks DH, Tadmor G, MacLeod R. 2005. A method to reconstruct activation wavefronts without isotropy assumptions using a level set approach. In *Lecture Notes in Computer Science*, Vol. 3504: *Functional imaging and modeling of the heart: third international workshop*, FIMH 2005, Barcelona, Spain, June 2–4, 2005, pp. 195–204. Ed. AF Frangi, PI Radeva, A Santos, M Hernandez. Berlin: Springer.
50. Lysaker OM, Nielsen BF. 2006. Toward a level set framework for infarction modeling: an inverse problem. *Int J Num Anal Model* **3**:377–394.
51. Bal G, Ren K. 2005. *Reconstruction of singular surfaces by shape sensitivity analysis and level set method*. Preprint, Columbia University. http://www.columbia.edu/ gb2030/PAPERS/Sing LevelSet.pdf.
52. Dorn O. 2004. Shape reconstruction in scattering media with voids using a transport model and level sets. *Can Appl Math Q* **10**:239–275.
53. Dorn O. 2006. Shape reconstruction for an inverse radiative transfer problem arising in medical imaging. In *Numerical methods for multidimensional radiative transfer problems*. Springer series Computational Science and Engineering. Ed. G Kanschat, E Meinköhn, R Rannacher, R Wehrse. Springer: Berlin. To appear.
54. Ishimaru A. 1978. *Wave propagation and scattering in random media*. New York: Academic Press.
55. Natterer F, Wübbeling F. 1995. A propagation-backpropagation method for ultrasound tomography. *Inverse Probl* **11**:1225–1232.
56. Osher S, Santosa F. 2001. Level set methods for optimisation problems involving geometry and constraints I. Frequencies of a two-density inhomogeneous drum. *J Comput Phys* **171**:272–288.
57. Osher S, Paragios N. 2003. *Geometric level set methods in imaging, vision and graphics*. Berlin: Springer.
58. Sikora J, Zacharopoulos A, Douiri A, Schweiger M, Horesh L, Arridge S, Ripoll J. 2006. Diffuse photon propagation in multilayered geometries. *Phys Med Biol* **51**:497–516.
59. Zacharopoulos A, Arridge S, Dorn O, Kolehmainen V, Sikora J. 2006. 3D shape reconstruction in optical tomography using spherical harmonics and BEM. *Proceedings of the progress in electromagnetics research symposium*, Cambridge, MA, March 26–29, 2006. To appear.

4

SHAPE AND TEXTURE-BASED DEFORMABLE MODELS FOR FACIAL IMAGE ANALYSIS

Stan Z. Li and Zhen Lei

Institute of Automation, Chinese Academy of Sciences, Beijing, China

Ying Zheng and Zeng-Fu Wang

Department of Automation, Institute of Information Science and Technology, University of Science and Technology of China, Hefei, China

In this chapter we introduce concepts and algorithms of shape- and texture-based deformable models — more specifically Active Shape Models (ASMs), Active Appearance Models (AAMs), and Morphable Models — for facial image analysis. Such models, learned from training examples, allow admissible deformations under statistical constraints on the shape and/or texture of the pattern of interests. As such, the deformation is in accordance with the specific constraints on the pattern. Based on analysis of problems with the standard ASM and AAM, we further describe enhanced models and algorithms, namely Direct Appearance Models (DAMs) and a Texture-Constrained ASM (TC-ASM), for improved fitting of shapes and textures. A method is also described for evaluation of goodness of fit using an ASM. Experimental results are provided to compare different methods.

1. INTRODUCTION

Many image based systems require alignment between an object in the input image and a target object. The alignment quality can have a great impact on system

Address all correspondence to: Stan Z. Li, Center for Biometrics and Security Research, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, 95 Zhongguancun Donglu, Beijing 100080, China. Phone: +86-10-8262 6787; Fax: +86- 10-6265 9350. szli@cbsr.ia.ac.cn, szli@nlpr.ia.ac.cn. Web: http://www.cbsr.ia.ac.cn/, http://www.cbsr.ia.ac.cn/homapage/szli.

performance. For face analysis, in particular, both shapes and textures provide important clues useful for characterizing faces. The task of face alignment is to accurately locate facial features such as the eyes, nose, mouth, and outline, and to normalize facial shape and texture. Accurate extraction and alignment of these features offers advantages for many applications.

A sort of the most successful face alignment method is the deformable model, which can represent variations in either the shape or texture of the target objects. As two typical deformable model types, the active shape models (ASMs) [1] and active appearance models (AAMs) [2, 3] have been widely used as alignment algorithms in medical image analysis and face analysis [4] for the past decade. The standard ASM consists of two statistical models: (1) a global shape model, which is derived from the landmarks in the object contour, and (2) a local appearance model, which is derived from the profiles perpendicular to the object contour around each landmark. The ASM uses local models to find the candidate shape and the global model to constrain the searched shape. The AAM makes use of subspace analysis techniques, PCA in particular, to model both shape variation and texture variation, and the correlations between them. The integrated shape and texture is referred to as *appearance*. In searching for a solution, it assumes linear relationships between appearance variation and texture variation, and between position variation and texture variation; and learns the two linear regression models from training data. Minimization in high-dimensional space is reduced in the two models. This strategy is also developed in the active blob model[5].

ASMs and AAMs can be expanded in several ways. The concept, originally proposed for the standard frontal view, can be extended to multi-view faces, either by using piecewise linear modeling [6] or nonlinear modeling [7]. Cootes and Taylor show that imposing constraints such as fixing eye locations can improve AAM search results [8]. Blanz and Vetter extended morphable models and the AAM to model the relationship of 3D head geometry and facial appearance [9]. Li et al. [10] present a method for learning 3D face shape modeling from 2D images based on a shape-and-pose-free texture model. In Duta et al. [11], the shapes are automatically aligned using procrustean analysis, and clustered to obtain cluster prototypes and statistical information about intra-cluster shape variation. In Ginneken et al. [12], a K-nearest-neighbors classifier is used and a set of features selected for each landmark to build local models. Baker and colleagues [13] propose an efficient method called an "inverse compositional algorithm" for alignment. Ahlberg [14] extends the AAM to a parametric method called an Active Appearance algorithm to extract positions parameterized by 3D rotation, 2D translation, scale, and six Action Units (controlling the mouth and the eyebrows). In the direct appearance model (DAM) [15, 16], shape is modeled as a linear function of texture. Using such an assumption, Yan et al. [17] propose a texture-constrained ASM (TC-ASM), which has the advantage of an ASM in having good localization accuracy and that of an AAM in having insensitivity to initialization. To construct an effective evaluation function, a statistical learning approach was

proposed for face alignment by Huang et al. [18] using a nonlinear classification function learned from a training set of positive and negative training examples.

The following sections first describe the classical ASM and AAM. We will then briefly review the 3D Morphable Model as an important 3D deformable model. After that, two of the improved face alignment algorithms — DAM and TC-ASM — will be introduced based on the analysis of the problems of classical the ASM and AAM. Then an alignment quality evaluation mechanism is addressed before the experimental results and conclusion to end this chapter are presented.

For all the algorithms presented here, a training set of shape–texture pairs is assumed to be available and denoted as $\mathbf{\Omega} = \{(S_0, T_0)\}$, where a *shape* $S_0 = ((x_1, y_1), \ldots, (x_K, y_K)) \in \mathbb{R}^{2K}$ is a sequence of K points in the 2D image plane, and a *texture* T_0 is the patch of pixel intensities enclosed by S_0. Let $\overline{S}$ be the mean shape of all the training shapes, as illustrated in Figure 1. All the shapes are aligned or warping to the tangent space of the mean shape $\overline{S}$. After that texture T_0 is warped correspondingly to $T \in \mathbb{R}^L$, where L is the number of pixels in the mean shape $\overline{S}$. The warping may be done by pixel value interpolation, e.g., using a triangulation or thin plate spline method.

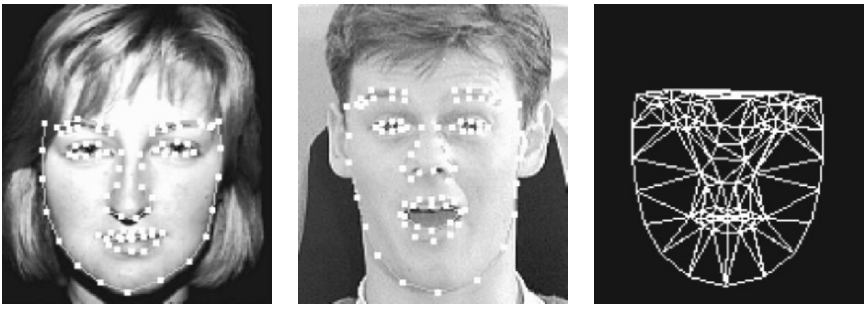


Figure 1. Two face instances labeled with 83 landmarks and the mesh of the mean shape. Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier. See attached CD for color version.

2. CLASSICAL DEFORMABLE MODELS

There are two classical deformable models for 2D face analysis — the Active Shape Model (ASM) and the Active Appearance Model (AAM). We first look through them; the model for 3D face analysis will be addressed later.

The ASM seeks to match a set of model points to an image by searching along profiles of each point under the constraint of a statistical shape model. The AAM

seeks to match both the position of the model points and a representation of the texture to an image by updating the model parameters using the difference between the current synthesized image and the target image.

There are three key differences between the two models [19]:

1. The ASM only uses models of the image texture in small regions about each landmark point, whereas the AAM uses a model of the appearance of the whole of the region (usually inside a convex hull around the points).

2. The ASM searches around the current position, typically along profiles normal to the boundary, whereas the AAM only samples the image enclosed by the current position.

3. The ASM essentially seeks to minimize the distance between model points and the corresponding points found in the image, whereas the AAM seeks to minimize the difference between the synthesized model image and the target image.

2.1. Active Shape Model

In the ASM a shape is represented as a vector s in the low-dimensional shape eigenspace $\mathbb{R}^k$, spanned by $k\,(<2K)$ principal modes (major eigenvectors) learned from the training shapes. A shape S could be linearly obtained from the shape eigenspace

$$S = \overline{S} + \mathbf{U}s, \tag{1}$$

where $\mathbf{U}$ is the matrix consisting of k principal modes of the covariance of $\{S_0\}$.

The local appearance models, which describe a local image feature around each landmark, are modeled as the first derivatives of the sampled profiles perpendicular to the landmark contour [4]. For the jth landmark ($j=1,\cdots,K$), we can derive the mean profile $\overline{g}_j$ and the covariance matrix $\mathbf{\Sigma}_j^g$ from the jth profile examples directly. At the current position $(x_j^{(n-1)}, y_j^{(n-1)})$ of the jth landmark, the local appearance models find the "best" candidate, (x_j^n, y_j^n), in neighborhood $N(x_j^{(n-1)}, y_j^{(n-1)})$ surrounding $(x_j^{(n-1)}, y_j^{(n-1)})$, by minimizing the energy:

$$(x_j^n, y_j^n) = \underset{(x,y)\in N(x_j^{(n-1)}, y_j^{(n-1)})}{\arg\min} \|g_j(x,y) - \overline{g}_j\|^2_{\mathbf{\Sigma}_j^g}, \tag{2}$$

where $g_j(x,y)$ is the profile of the jth landmark at (x,y) and $\|X\|^2_{\mathbf{A}} = X^T\mathbf{A}^{-1}X$ is the Mahalanobis distance measure with respect to a real symmetric matrix $\mathbf{A}$.

After relocating all the landmarks using the local appearance models, we obtain a new candidate shape S_{lm}^n. The solution in shape eigenspace is derived by maximizing the likelihood:

$$s^n = \arg\max_s p(S^n_{lm}|s) = \arg\min_s Eng(S^n_{lm}; s), \tag{3}$$

where[1]

$$Eng(S^n_{lm}; s) = \lambda \|S^n_{lm} - S^n_{lm}{}'\|^2 + \|s^n_{lm} - s\|^2_{\mathbf{\Lambda}}. \tag{4}$$

In above equation, $s^n_{lm} = U^T(S^n_{lm} - \overline{S})$ is the projection of S^n_{lm} to the shape eigenspace, $S^n_{lm}{}' = \overline{S} + U s^n_{lm}$ is the reconstructed shape, and $\mathbf{\Lambda}$ is the diagonal matrix of the largest eigenvalues of the training data $\{S_i\}$. The first term is the squared Euclidean distance from S^n_{lm} to the shape eigenspace, and the second is the squared Mahalanobis distance between s^n_{lm} and s; λ balances the two terms.

Using the local appearance models leads to fast convergence to the local image evidence. However, since they are modeled based on the local features, and the "best" candidate point is only evaluated in the local neighborhood, the solution of the ASM is often suboptimal, dependent on the initialization.

2.2. Active Appearance Model

In the AAM, a shape is also modeled by k ($<2K$) principal modes learned from the training shapes by PCA, just as Eq. (1) shows with the ASM.

After aligning each training shape S_0 to the mean shape and warping the corresponding texture T_0 to T, the warped textures are aligned to the tangent space of the mean texture $\overline{T}$ by using an iterative approach [2]. The PCA model for the warped texture is obtained as

$$T = \overline{T} + \mathbf{V}t, \tag{5}$$

where $\mathbf{V}$ is the matrix consisting of ℓ principal orthogonal modes of variation in $\{T\}$, and t is the vector of texture parameters. The projection from T to t is

$$t = \mathbf{V}^T(T - \overline{T}) = \mathbf{V}^T T. \tag{6}$$

By this, the L pixel values in the mean shape is represented as a point in the texture subspace $\mathbb{S}_t$ in $\mathbb{R}^\ell$.

The appearance of each example is a concatenated vector:

$$A = \begin{pmatrix} \mathbf{\Lambda} s \\ t \end{pmatrix}, \tag{7}$$

where $\mathbf{\Lambda}$ is a diagonal matrix of weights for the shape parameters allowing for the difference in units between the shape and texture variation, typically defined as $r\mathbf{I}$. Again, by applying PCA on the set $\{A\}$, one gets

$$A = \mathbf{W}a, \tag{8}$$

where $\mathbf{W}$ is the matrix consisting of principal orthogonal modes of the variation in $\{A\}$ for all training samples. The appearance subspace $\mathbb{S}_a$ is modeled by

$$a = \mathbf{W}^T A. \tag{9}$$

The search for an AAM solution is guided by the following difference between the texture T_{im} in the image patch and the texture T_a reconstructed from the current appearance parameters:

$$\delta T = T_{im} - T_a. \tag{10}$$

More specifically, the search for a face in an image is guided by minimizing the norm $\|\delta T\|$. The AAM assumes that the appearance displacement δa and the position (including coordinates (x, y), scale s, and rotation parameter θ) displacement δp are linearly correlated to δT:

$$\delta a = \mathbf{A}_a \delta T \tag{11}$$

$$\delta p = \mathbf{A}_p \delta T \tag{12}$$

The prediction matrices $\mathbf{A}_a, \mathbf{A}_p$ are to be learned from the training data by using linear regression. In order to estimate $\mathbf{A}_a$, a is displaced systematically to induce $(\delta a, \delta T)$ pairs for each training image. Due to the large consumption of memory required for the learning of $\mathbf{A}_a$ and $\mathbf{A}_p$, learning has to be done with a small, limited set of $\{\delta a, \delta T\}$.

2.3. 3D Morphable Model

While the ASM and AAM are for 2D image pattern analysis, in this section we temporarily deviate from analysis of a 2D face, and extend the dimension of face data to 3D by introducing morphable models [9, 20, 21]. With the presence of convenient 3D acquiring equipment and the development of the computer hardware, 3D face analysis has now become feasible and promising since it is invariant to the influence of pose, illumination, and expression. One of the most crucial problems for all 3D data-processing systems is the alignment between the input data and the standard. The 3D alignment may involve many rigid or non-rigid transformations. For 3D face analysis, in particular, alignment means reconstruction of a normalized 3D face model from either input 2D face images or unrestrained 3D data. The 3D Morphable Model (3DMM), as a typical 3D deformable model, inherits both the spirit of a multidimensional morphable model [22] and the AAM.

The 3DMM is a model of faces represented in 3D where shape information is separated from texture information. The shape and texture models are learned from a database of 3D faces, i.e., faces acquired by a 3D *Cyberware*^{TM} scanner. Building a 3DMM requires transforming the shape and texture spaces into

vector spaces for which any convex combination of exemplar shapes and textures describes a realistic human face. *Correspondence* is the basic requirement for constructing such a vector space. In [23], correspondences are established between all exemplar faces and a reference face by an optical flow algorithm. This scheme brings a consistent labeling of vertices and corresponding albedos across the whole set of exemplar faces. The shape of an exemplar face is then represented by a shape vector $S^{ex} = ((x_1, y_1, z_1) \ldots, (x_K, y_K, z_K)) \in \mathbb{R}^{3K}$ that contains the x, y, z coordinates of K vertices. The texture of the face is represented by a texture vector $T^{ex} = ((R_1, G_1, B_1) \ldots, (R_K, G_K, B_K)) \in \mathbb{R}^{3K}$ that contains the R, G, B texture values sampled at the same K vertices.

A new face can then be generated by convex combination of the K exemplar faces, with their shape and texture vectors, S and T, expressed as

$$S = \sum_{i=1}^{K} a_i S_i^{ex} \quad T = \sum_{i=1}^{K} b_i T_i^{ex} \quad \sum_{i=1}^{K} a_i = \sum_{i=1}^{K} b_i = 1. \tag{13}$$

Again, PCA is applied separately on the shape and texture space to reduce dimensionality. Now, instead of describing a new face as a convex combination of exemplars, as in Eq. (13), we use the similar shape and texture PCA model of Eq. (1), (5) as

$$S = \overline{S} + \mathbf{U}s \quad T = \overline{T} + \mathbf{V}t. \tag{14}$$

Note that U and V are the matrices consisting of orthogonal modes of variations in $\{S^{ex}\}$ and $\{T^{ex}\}$. The 3DMM shape and texture coefficient vectors s and t are low-dimensional coding of the identity of a face invariant to pose and illumination influence. Given an input 2D image under arbitrary pose and illumination conditions or unrestrained 3D face data, the 3DMM can recover the vectors of s and t by an *analysis by synthesis*, providing an alignment between input face and exemplar faces in the database.

3. MOTIVATIONS FOR IMPROVEMENTS

ASM uses the local appearance models to search along the profiles of candidate points. It leads to fast convergence to the local image evidence. However, since they are modeled based on local features, and the "best" candidate point is only evaluated in the local neighborhood, the solution of the ASM is often suboptimal, dependent on the initialization.

By analyzing the relationships between the shape, texture, and appearance subspaces in the AAM, we will show the defects of the model. Thereby, we suggest a property that an ideal appearance model should have, which motivates us to propose improvements to the classical model.

First, let us look into the relationship between shape and texture from an intuitive viewpoint. A texture (i.e., the patch of intensities) is enclosed by a shape

(before aligning to the mean shape); the same shape can enclose different textures (i.e., configurations of pixel values). However, the reverse is not true: different shapes cannot enclose the same texture. So the mapping from the texture space to the shape space is many-to-one. The shape parameters should be determined completely by texture parameters but not vice versa.

Let us now look further into the correlations or constraints between the linear subspaces $\mathbb{S}_s, \mathbb{S}_t$ and $\mathbb{S}_a$ in terms of their dimensionalities or ranks. Let us denote the rank of space $\mathbb{S}$ by dim($\mathbb{S}$). We have the following analysis:

1. When dim($\mathbb{S}_a$)=dim($\mathbb{S}_t$)+dim($\mathbb{S}_s$), the shape and texture parameters are independent of each other, and there exist no mutual constraints between the s and t parameters.

2. When dim($\mathbb{S}_t$)<dim($\mathbb{S}_a$)<dim($\mathbb{S}_t$)+ dim($\mathbb{S}_s$), not all the shape parameters are independent of the texture parameters. That is, one shape can correspond to more than one texture configuration in it, which conforms our intuition.

3. One can also derive the relationship dim($\mathbb{S}_t$)<dim($\mathbb{S}_a$) from Eqs. (7) and (8) and write
$$\mathbf{W}a = \begin{pmatrix} \mathbf{\Lambda}s \\ t \end{pmatrix} \tag{15}$$
when that s contains some components that are independent of t.

4. However, in the AAM it is often the case where dim($\mathbb{S}_a$)<dim($\mathbb{S}_t$) if the dimensionalities of $\mathbb{S}_a$ and $\mathbb{S}_t$ are chosen to retain, say, 98% of the total variations, which is reported by Cootes [2] and also observed by us. The consequence is that some admissible texture configurations cannot be seen in the appearance subspace because dim($\mathbb{S}_a$)<dim($\mathbb{S}_t$), and therefore cannot be reached by the AAM search. We consider this a flaw in the AAM's modeling of its appearance subspace.

From the above analysis we conclude that the ideal model should be dim($\mathbb{S}_a$) = dim($\mathbb{S}_t$), and hence that s is completely linearly determinable by t. In other words, the shape should be linearly dependent on the texture, so that dim($\mathbb{S}_t \cup \mathbb{S}_s$) = dim($\mathbb{S}_t$). The direct appearance model (DAM) is proposed mainly for this purpose.

Another motivation of the DAM is memory consumption: the regression of $\mathbf{A}_a$ with the AAM is very memory consuming. The AAM prediction needs to model linear the relationship between appearance and the texture difference according to Eq. (11). However, both δa and δT are high-dimensional vectors, and therefore the storage size of training data generated for learning Eq. (11) increases very rapidly as the dimensions increase. It is very difficult to train the AAM for $\mathbf{A}_a$ even with a moderate number of images. Learning in a low-dimensional space will relieve the burden.

4. DIRECT APPEARANCE MODELS

In this section we introduce an improved appearance model, called the Direct Appearance Model (DAM), for aligning and estimating face appearances.

The new appearance model is motivated by our findings of a flaw in AAM modeling and the difficulties in training the AAM presented in previous section. The DAM model overcomes these problems by its proper subspace modeling based on the fundament that the mapping from the texture subspace to the shape subspace is many-to-one, and therefore a shape can be determined entirely by the texture in it. From these relationships, the DAM model considers an appearance that is composed of both shape and texture, to be determinable by using just the corresponding texture. The DAM uses the texture information *directly* to predict the shape and to update the estimates of position and appearance (hence the name DAM) in contrast to the AAM's crucial idea of modeling the AAM appearance subspace from shape and texture combined. In this way, the DAM includes admissible textures previously unseen by the AAM and improves convergence and accuracy.

Another merit of the DAM is that it predicts the new face position and appearance based on principal components of texture difference vectors, instead of the raw vectors themselves as with the AAM. This cuts down the memory requirement to a large extent and further improves convergence and accuracy. The claimed advantages of the DAM are substantiated by comparative experimental results in Section 7.1.

4.1. DAM Modeling and Training

DAM consists of a shape model, a texture model, and a prediction model. It predicts the shape parameters directly from the texture parameters. The shape and texture models are built based on PCA in the same way as with the AAM. The prediction model includes two parts: prediction of position and prediction of texture.

Recall the conclusions we made earlier: (1) an ideal model should have $\dim(\mathbb{S}_a) = \dim(\mathbb{S}_t)$, and (2) shape should be computable uniquely from texture but not vice versa. We propose the following prediction model by assuming a linear relationship between shape and texture:

$$s = \mathbf{R}t + \varepsilon, \tag{16}$$

where $\varepsilon = s - \mathbf{R}t$ is noise and $\mathbf{R}$ is a $k \times l$ projection matrix. Denoting the expectation by $E(\cdot)$, if all the elements in variance matrix $E(\varepsilon\varepsilon^T)$ are small enough, the linear assumption made in Eq. (16) is approximately correct. This is true, as will be verified later by experiments. Define the objective cost function

$$C(\mathbf{R}) = E(\varepsilon^T\varepsilon) = \text{trace}[E(\varepsilon\varepsilon^T)]. \tag{17}$$

$\mathbf{R}$ is learned from training example pairs $\{(s, t)\}$ to minimize the cost function.

Consider variation $\delta C(\mathbf{R})$ caused by $\delta\mathbf{R}$:

$$\begin{aligned}
&\delta C(\mathbf{R}) \qquad (18)\\
&= \text{trace}\{E([s-(\mathbf{R}+\delta\mathbf{R})t][s-(\mathbf{R}+\delta\mathbf{R})t]^T)\}\\
&\quad -\text{trace}[E\{[s-\mathbf{R}t][s-\mathbf{R}t]^T\}]\\
&= \text{trace}\{E[\mathbf{R}tt^T\delta\mathbf{R}^T+\delta\mathbf{R}tt^T\mathbf{R}\\
&\quad -st^T\delta\mathbf{R}^T-\delta\mathbf{R}ts^T]\}\\
&= \text{trace}\{\mathbf{R}E(tt^T)\delta\mathbf{R}^T+\Delta\mathbf{R}E(tt^T)\mathbf{R}\\
&\quad -E(st^T)\Delta\mathbf{R}^T-\delta\mathbf{R}E(ts^T)\}.
\end{aligned}$$

Letting $\delta C(\mathbf{R})=0$, we get

$$\begin{aligned}
&\text{trace}\{\delta\mathbf{R}E(tt^T)\delta\mathbf{R}^T+\delta\mathbf{R}E(tt^T)\mathbf{R}\} \qquad (19)\\
&\quad = \text{trace}\{E(st^T)\Delta\mathbf{R}^T+\Delta\mathbf{R}E(ts^T)\}
\end{aligned}$$

for any $\|\delta\mathbf{R}\|\to 0$. Substituting $\delta\mathbf{R}$ by $\epsilon 1_{i,j}$ for any (i,j), where $\epsilon\to 0$ and $1_{i,j}$ is the matrix in which entry (i,j) is 1 and 0 elsewhere, we arrive at $\mathbf{R}E(tt^T)=E(st^T)$, and hence obtain an optimal solution:

$$\mathbf{R}=E(st^T)[E(tt^T)]^{-1}. \qquad (20)$$

The minimized cost is the trace of the following:

$$E(\varepsilon\varepsilon^T)=E(ss^T)-\mathbf{R}E(tt^T)\mathbf{R}^T. \qquad (21)$$

Instead of using δT directly as in the AAM search (cf. Eq. (12)), we use principal components of it, $\delta T'$, to predict the position displacement:

$$\delta p=\mathbf{R}_p\delta T', \qquad (22)$$

where $\mathbf{R}_p$ is the prediction matrix learned by using linear regression. To do this, we collect texture differences induced by small position displacements in each training image, and perform PCA on this data to get the projection matrix $\mathbf{H}^T$. A texture difference is projected onto this subspace as

$$\delta T'=\mathbf{H}^T\delta T, \qquad (23)$$

where $\delta T'$ is about 1/4 of δT in dimensionality, and this makes the prediction more stable. The DAM regression in Eq. (22) requires much less memory than the AAM regression in Eq. (11). This is because p is of much lower dimension than a and $\delta T'$ much lower than δT. This will be illustrated by numbers later.

Assume that a training set is given as $\mathbf{A} = \{(S_i, , T_i)\}$ where a shape $S_i = ((x_1^i, y_1^i), \ldots, (x_K^i, y_K^i)) \in \mathbb{R}^{2K}$ is a sequence of K points in the 2D image plane, and a texture T_i is the patch of image pixels enclosed by S_i. The DAM learning consists of two parts: (1) learning $\mathbf{R}$, and (2) learning $\mathbf{H}$ and $\mathbf{R}_p$. (1) $\mathbf{R}$ is learned from the shape–texture pairs $\{s, t\}$ obtained from the landmarked images. (2) To learn $\mathbf{H}$ and $\mathbf{R}_p$, artificial training data are generated by perturbing the position parameters p around the landmark points to obtain $\{\delta p, \delta T\}$; then learn $\mathbf{H}$ from $\{\delta T\}$ using PCA; $\delta T'$ is computed after that; and, finally, $\mathbf{R}_p$ is derived from $\{\delta p, \delta T'\}$.

The DAM regression in Eq. (22) requires much less memory than the AAM regression in Eq. (11); typically, a DAM needs only about 1/20th the memory required by an AAM. For the DAM, there are 200 training images, 4 parameters for the position $(x, y, \theta, scale)$, and 6 disturbances for each parameter to generate training data for the training $\mathbf{R}_p$. So the size of the training data for a DAM is $200 \times 4 \times 6 = 4,800$. For AAM there are 200 training images, 113 appearance parameters, and 4 disturbances for each parameter to generate training data for training $\mathbf{A}_a$. The size of the training data set for $\mathbf{A}_a$ is $200 \times 113 \times 4 = 90,400$. Therefore, the size of the training data set for an AAM's prediction matrices is $90,400 + 4,800 = 95,200$, which is 19.83 times that for a DAM. On a PC, for example, the memory capacity for AAM training with 200 images would allow DAM training with 3,966 images.

Note that there is a variant of a basic AAM [4], which uses texture difference to predict shape difference. The prediction of shape is done by $\delta s = \mathbf{B}\delta T$. However, this variant is not as good as the basic AAM [4].

4.2. DAM Search

The DAM prediction model leads to the following search procedure. The DAM search starts with the mean shape and mean texture, equivalent to the mean appearance with $a_0 = 0$, at a given initial position p_0. The texture difference δT is computed from the current shape patch at the current position, and its principal components are used to predict and update p and s using the DAM linear models described above. If $\|\delta T\|$ calculated using the new appearance at the position is smaller than the old one, the new appearance and position are accepted; otherwise, the position and appearance are updated by amounts $\kappa\delta a$ and $\kappa\delta p$ with varying κ values. The search algorithm is summarized below:

1. Initialize position parameters p_0, and set shape parameters $s_0 = 0$;

2. Get texture T_{im} from the current position, project it into the texture subspace $\mathbb{S}_t$ as t; reconstruct the texture T_{rec}, and compute texture difference $\delta T_0 = T_{im} - T_{rec}$ and the energy $E_0 = \|\delta T_0\|^2$;

3. Compute $\delta T' = \mathbf{H}^T \delta T$, and get the position displacement $\delta p = \mathbf{R}_p \delta T'$;

4. Set step size $\kappa = 1$;

5. Update $p = p_0 - \kappa\delta p$, $s = \mathbf{R}t$;

6. Compute difference texture δT using the new shape at the new position, and its energy $E_0 = \|\delta T_0\|^2$;

7. If $|E - E_0| < \epsilon$, the algorithm is converged; exit;

8. If $E < E_0$, then let $p_0 = p$, $s_0 = s$, $\delta T_0 = \delta T$, $E_0 = E$, goto 3;

9. Change κ to the next smaller number in $\{1.5, 0.5, 0.25, 0.125, \ldots, \}$, goto 5;

The above DAM search can be performed with a multi-resolution pyramid structure to improve the result.

4.3. Multi-View DAM

In multi-view face alignment, the whole range of views from frontal to side views are partitioned into several sub-ranges, and one DAM model is trained to represent the shape and texture for each sub-range. Which view DAM model to use may be decided by using some pose estimate for static images. In the case of face alignment from video, the previous view plus the two neighboring view DAM models may be attempted, and then the final result is chosen to be the one with the minimum texture residual error.

The full range of face poses are divided into 5 view sub-ranges: $[-90°, -55°]$, $[-55°, -15°]$, $[-15°, 15°]$, $[15°, 55°]$, and $[55°, 90°]$, with $0°$ being the frontal view. The landmarks for frontal, half-side, and full-side view faces are illustrated in Figure 2. The dimensions of shape and texture vectors before and after the PCA dimension reductions are shown in Table 1, where the dimensions after PCA are chosen to be such that 98% of the corresponding total energies are retained. The texture appearances due to respective variations in the first three principal components of texture are demonstrated in Figure 3.

The left- models and right-side models are reflections of each other, andn thus we only need train one side. So we train $[-15°, 15°]$, $[15°, 55°]$, and $[55°, 90°]$ for the 5 models. We can find the corresponding model for all the face with a view in $[-90°, 90°]$.

The multi-view DAM search has a similar process to that of the standard DAM. The difference lies in the beginning of the iteration where multi- view DAM has to determine which view the input image belongs to and select a proper DAM model. Note that p can be computed from δT in one step as $\delta p = \mathbf{R}_T \delta T$, where $\mathbf{R}_T = \mathbf{R}_p \mathbf{H}^T$, instead of two steps as in Eqs. (22) and (23). The search algorithm is summarized below:

Table 1. Dimensionalities of shape and texture variations for face data

View	#1	#2	#3	#4	#5
Fontal	87	69	3185	144	878
Half-Side	65	42	3155	144	1108
Full-Side	38	38	2589	109	266

#1 Number of landmark points. #2 Dimension of shape space $\mathbb{S}_s$. #3 Number of pixel points in the mean shape. #4 Dimension of texture space $\mathbb{S}_t$. #5 Dimension of texture variation space ($\delta T'$). Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309–314. Copyright ©2002, IEEE.

1. Initialize position parameters p_0, and determine the view by which to select the DAM model to use; set shape parameters $s_0 = 0$;
2. Get texture T_{im} from the current position; project it into texture subspace $\mathbb{S}_t$ as t; reconstruct texture T_a, and compute texture difference $\delta T_0 = T_{im} - T_a$ and the energy $E_0 = \|\delta T_0\|^2$;
3. get position displacement $\delta p = \mathbf{R}_T \delta T$;
4. Set step size $\kappa = 1$;
5. Update $p = p_0 - \kappa \delta p$, $s = \mathbf{R}t$;
6. Compute difference texture δT using the new shape at the new position, and its energy $E = \|\delta T\|^2$;
7. If $|E - E_0| < \epsilon$, the algorithm is converged; exit;
8. If $E < E_0$, then let $p_0 = p$, $s_0 = s$, $\delta T_0 = \delta T$, $E_0 = E$, goto 3;
9. Change κ to the next number in $\{1.5, 0.5, 0.25, 0.125, \ldots, \}$, goto 5.

In our implementation, the initialization and pose estimation are performed automatically by using a robust real-time multi-view face detector [24], as shown in Figure 4. A multi-resolution pyramid structure is used in the search to improve the result. Figure 5 demonstrates scenarios of how DAM converges.

When the face has undergone large variation due to stretch in either the x or y direction, model fitting can be improved by allowing different scales in the two directions. This is done by splitting the scale parameter in two: s_x and s_y. The improvement is demonstrated in Figure 6.

Figure 2. Frontal, half-side, and full-side view faces and the labeled landmark points. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309– 314. Copyright ©2002, IEEE.

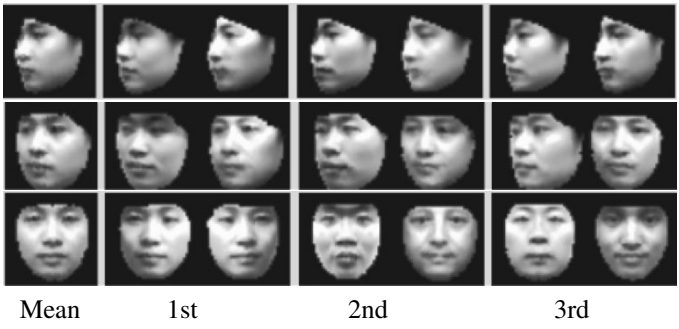


Figure 3. Texture and shape variations due to variations in the first three principal components of the texture (the shapes change in accordance with $s = \mathbf{R}t$) for full-side ($\pm 1\sigma$), half-side ($\pm 2\sigma$), and frontal ($\pm 3\sigma$) views. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309–314. Copyright ©2002, IEEE.

5. TEXTURE-CONSTRAINED ACTIVE SHAPE MODEL

A TC-ASM [17] imposes the linear relationship of the direct appearance model (DAM) to improve the ASM search. The motivation is as follows. The ASM has better accuracy in shape localization than the AAM when the initial shape is placed close enough to the true shape, whereas the latter model incorporates information about texture enclosed in the shape and hence yields lower texture reconstruction error. However, the ASM makes use of constraints near the shape only, without a global optimality criterion, and therefore the solution is sensitive to the initial shape position. In the AAM, the solution-finding process is based on the linear relationship between the variation of the position and the texture reconstruct error. The reconstruct error, δT, is influenced very much by the illumination. Since δT

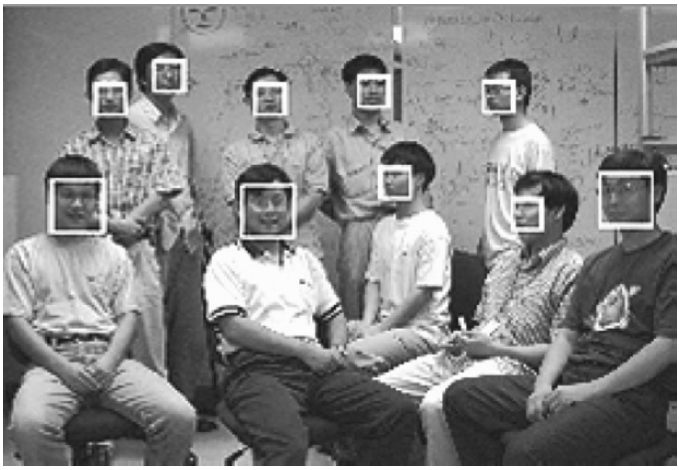


Figure 4. Initial alignment provided by a multi-view face detector. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309–314. Copyright ©2002, IEEE.

Figure 5. DAM aligned faces (from left to right) at the 0th, 5th, 10th, and 15th iterations, and the original images for (top–bottom) frontal, half-side and full-side view faces. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309–314. Copyright ©2002, IEEE.

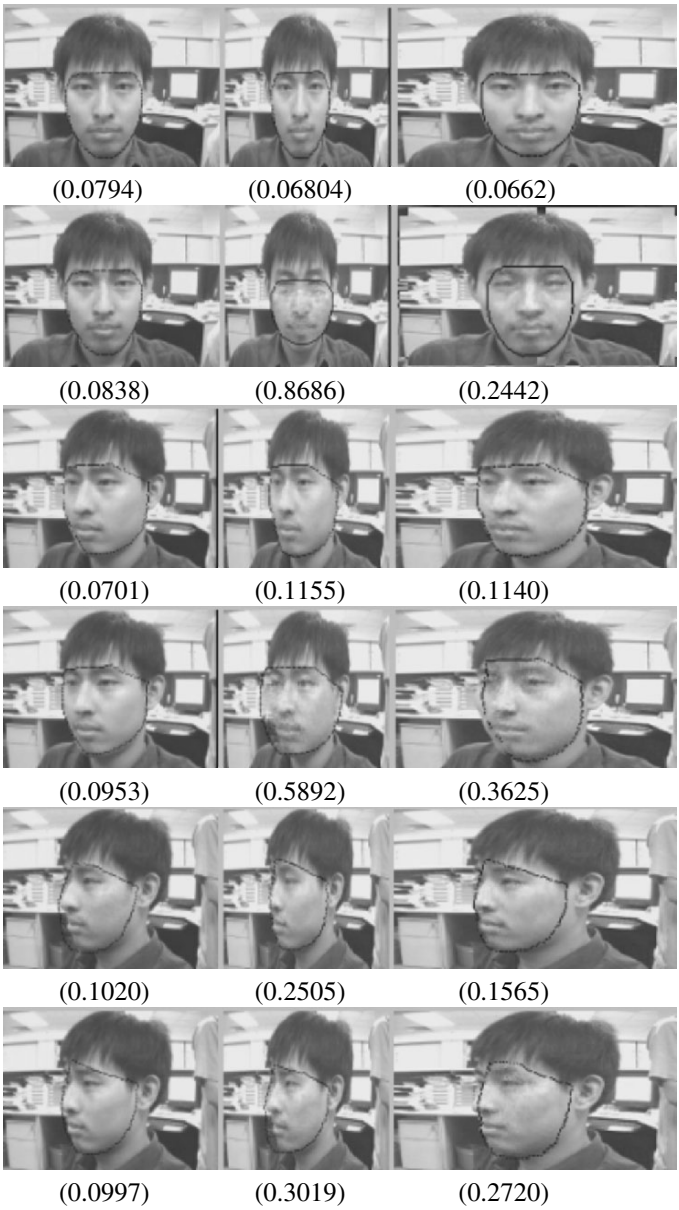


Figure 6. Results of non-isometric (top of each of the three blocks) and isometric (bottom) search for frontal (top block), half-side (middle block), and full-side (bottom block) view faces. From left to right of each row are normal, and stretched faces. The number below each result is the corresponding residual error. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309–314. Copyright ©2002, IEEE.

is orthogonal to $\mathbb{S}_t$ (projected back to $\mathbb{R}^L$) and $\dim(\mathbb{S}_t) \ll \dim(T)$, the dimension of space $\{\delta T\}$ is very high, and it is hard to train regression matrix $\mathbf{A}_a, \mathbf{A}_p$, and the prediction of the variance of position can be subject to significant errors. Also it is time and memory intensive. The TC-ASM is aimed at overcoming the above problems.

The TC-ASM consists of a shape model, a texture model, K local appearance models, and a *texture-constrained* shape model. The former three types are exactly the same as in the ASM and AAM. The texture-constrained shape model, or the mapping from texture to shape, is simply assumed linear and could be easily learned. In each step of the optimization, a better shape is found under a Bayesian framework. The details of the model will be introduced in the following.

5.1. Texture-Constrained Shape Model

In the shape model, there are some landmarks defined on the edges or contours. Since they have no explicit definition for their positions, there exists uncertainty of the shape given the texture, while there are correlations between the shape and the texture. The conditional distribution of shape parameters s given texture parameters t is simply assumed Gaussian, i.e.,

$$p(s|t) \sim N(s_t, \mathbf{\Sigma}_t), \tag{24}$$

where $\mathbf{\Sigma}_t$ stands for the covariance matrix of the distribution, and s_t is linearly determined by texture t. The linear mapping from t to s_t is:

$$s_t = \mathbf{R}t, \tag{25}$$

where $\mathbf{R}$ is a projection matrix that can be pre-computed from training pairs $\{(s_i, t_i)\}$ by singular-valued decomposition. For simplicity, $\mathbf{\Sigma}_t$ is assumed to be a known constant matrix. Figure 7 demonstrates the accuracy of the prediction in the test data via matrix $\mathbf{R}$. We may see that the predicted shape is close to the labeled shape even under varying illuminations. Thus, the constraints over the shape from the texture can be used as an evaluation criterion in the shape localization task. The prediction of matrix $\mathbf{R}$ is also affected by illumination variation, yet since Eq. (25) is formulated based on the eigenspace, the influence of the unfamiliar illumination can be alleviated when the texture is projected to the eigenspace.

Distribution Eq. (24) can also be represented as the prior distribution of s given shape s_t:

$$p(s|s_t) \propto \exp\{-Eng(s; s_t)\}, \tag{26}$$

where the energy function is:

$$Eng(s; s_t) = \|s - s_t\|^2_{\mathbf{\Sigma}_t}. \tag{27}$$

Figure 7. Comparison of the manually labeled shape (middle row) and the shape (bottom row) derived from the enclosed texture using the learned projection matrix: $s_t = \mathbf{R}t$. In the top row are the original images. All the images are test data. Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier.

5.2. TC-ASM in Bayesian Framework

The TC-ASM search begins with the mean shape, namely shape parameters $s^0 = 0$. The whole search process is outlined as below:

1. Set the iteration number $n = 1$;
2. Using the local appearance models in the ASM, we may obtain the candidate shape, S^n_{lm}, with shape parameters s^n_{lm} based on the shape, $S^{(n-1)}$, of the previous iteration;
3. The texture enclosed by S^n_{lm} is warped to the mean shape, denoted by t^n. The texture-constrained shape s^n_t is predicted from t^n by Eq. (25);
4. The *posterior* (MAP) estimation of S^n or s^n, given by S^n_{lm} and s^n_t, is derived based on the Bayesian framework;

5. If the stopping condition is satisfied, exit; otherwise, $n = n + 1$, goto step 2.

In the following we illustrate step 4 and the stopping condition in detail. To simplify the notation, we shall omit superscript n in the following deduction since the iteration number is constant. In step 4 the *posterior* (MAP) estimation of s, given by S_{lm} and s_t, is

$$p(s|S_{lm}, s_t) = \frac{p(S_{lm}|s, s_t)p(s, s_t)}{p(S_{lm}, s_t)}. \tag{28}$$

Assume that S_{lm} is conditionally independent from s_t, given s, i.e.,

$$p(S_{lm}|s, s_t) = p(S_{lm}|s). \tag{29}$$

Then

$$p(s|S_{lm}, s_t) \propto p(S_{lm}|s)p(s|s_t). \tag{30}$$

The corresponding energy function is

$$Eng(s; S_{lm}, s_t) = Eng(S_{lm}; s) + Eng(s; s_t). \tag{31}$$

From Eqs. (4) and (27), the best shape obtained in each step is

$$\begin{aligned} s &= \arg\min_s [Eng(s; S_{lm}) + Eng(s; s_t)] \\ &= \arg\min_s \|s_{lm} - s\|^2_{\mathbf{\Lambda}} + \|s - s_t\|^2_{\mathbf{\Sigma}_t} \\ &= \arg\min_s [s^T(\mathbf{\Lambda}^{-1} + \mathbf{\Sigma}_t^{-1})s - 2s^T(\mathbf{\Lambda}^{-1}s_{lm} + \mathbf{\Sigma}_t^{-1}s_t)] \\ &= (\mathbf{\Lambda}^{-1} + \mathbf{\Sigma}_t^{-1})^{-1}(\mathbf{\Lambda}^{-1}s_{lm} + \mathbf{\Sigma}_t^{-1}s_t). \end{aligned}$$

After restoring the superscript of iteration number, the best shape obtained in step n is

$$s^n = (\mathbf{\Lambda}^{-1} + \mathbf{\Sigma}_t^{-1})^{-1}(\mathbf{\Lambda}^{-1}s^n_{lm} + \mathbf{\Sigma}_t^{-1}s^n_t). \tag{32}$$

This indicates that the best shape derived in each step is an interpolation between the shape from the local appearance model and the texture-constrained shape. In this sense, the TC-ASM could be regarded as a tradeoff between the ASM and AAM methods.

The stopping condition of the optimization is: if the shape from the local appearance model and the texture-constrained shape are the same, i.e., the solution generated by ASM is verified by the AAM, the optimal solution must have been touched. In practice, however, these two shapes would hardly turn out to be the same. A threshold is introduced to evaluate the similarity, and sometimes the convergence criterion in the ASM is used (if the above criterion has not been satisfied for a long time). For higher efficiency and accuracy, a multi-resolution pyramid method is adopted in the optimization process.

6. EVALUATION FOR FACE ALIGNMENT

The emergence of many effective face alignment algorithms serves as a contrast to the lack of an effective method for evaluation of face alignment results. In the ASM, there has been no convergence criterion for the iteration. As such, the ASM search can give a bad result without giving the user a warning. In the AAM and DAM, the PCA reconstruction error is used as a distance measure for evaluation of alignment quality. However, the reconstruction error may not be a good discriminant for evaluation of the alignment quality because a non-face can look like a face when projected onto the PCA face subspace. In the TC-ASM, the algorithm claims to reach a convergence when the solution generated by the ASM is verified by the AAM, whereas both convergence criteria are not yet stable.

In this section we propose a statistical learning approach for constructing an evaluation function for face alignment. A *nonlinear* classification function is learned from a training set of positive and negative training examples to effectively distinguish between qualified and unqualified alignment results. The positive subset consists of qualified face alignment examples, and the negative subset consists of obviously unqualified and near-but-not-qualified examples.

We use the AdaBoost algorithm [25, 26] for learning. A set of candidate weak classifiers are created based on edge features extracted using Sobel- like operators. We choose to use edge features because crucial cues for alignment quality are around edges. Experimentally, we also found that the Sobel features produced significantly better results than other features, such as Haar wavelets. AdaBoost learning selects or learns a sequence of best features and the corresponding weak classifiers, and combines them into a strong classifier.

In the training stage, several strong classifiers are learned in stages using bootstrap training samples, and in the test they are cascaded to form a stronger classifier, following an idea in boosting-based face detection [27]. Such a divide-and-conquer strategy makes the training easier and the good–bad classification more effective. The evaluation function thus learned gives a quantitative confidence and the good–bad classification is achieved by comparing the confidence with a learned optimal threshold.

There are two important distinctions between evaluation functions thus learned and the linear evaluation function of reconstruction error used in the AAM. First, the evaluation is learned in such a way to distinguish between good and bad alignment. Second, the scoring is nonlinear, which provides a semantically more meaningful classification between good and bad alignment. Experimental results demonstrate that the classification function learned using the proposed approach provides semantically meaningful scoring for classification between qualified and unqualified face alignment.

6.1. Solution Quality Evaluation in ASM/AAM

There has been no convergence criterion for ASM search. In ASM search, the mean shape is placed near the center of the detected image and a coarse-to-fine search performed. Large movements are made in the first few iterations, getting the position roughly. As the search is progressing, more subtle adjustments are made. The result can yield a good match to the target image or it can fail (see Figure 8). Failure can happen even if the starting position is near the target. When the variations of expression and illumination are large, ASM search can diverge in order to match the local image pattern.

Figure 8. Four face instances of qualified (top) and unqualified (bottom) examples with their warped images. Reprinted with permission from XS Huang, SZ Li, YS Wang. 2004. Statistical learning of evaluation function for ASM/AAM image alignment. In *Proceedings: Biometric Authentication, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004* (ECCV Workshop BioAW), pp. 45– 56. Ed D Maltoni, AK Jain. New York: Springer. Copyright ©2004, Springer.

A similar problem exists in AAM search. There, the PCA reconstruction error is used as a distance measure for evaluation of alignment quality (and for guiding the search as well). However, the reconstruction error may not be a good discriminant for evaluation of alignment quality because a non-face can look like a face when projected onto the PCA face subspace. Cootes pointed out that, of 2700 testing examples, 519 failed to converge to a satisfactory result (the mean point position error is greater than 7.5 pixels per point) [4].

In the following we present a learning-based approach for the learning evaluation function for ASM/AAM based alignment.

6.2. AdaBoost-Based Learning

Our objective is to learn an evaluation function from a training set of qualified and unqualified alignment examples. From now on we use the terms "positive" and "negative" examples for classes of data. These examples are the face image after warping to a mean shape, as shown in Figure 8. Face alignment quality evaluation can be posed as a two-class classification problem: given an alignment result x (i.e., warped face), evaluation function $H(x) = +1$ if x is positive example, or -1 otherwise. We want to learn such an $H(x)$ that can provide a score in $[-1, +1]$ with a threshold around 0 for the binary classification.

For two-class problems, a set of N labeled training examples is given as $(x_1, y_1), \ldots, (x_N, y_N)$, where $y_i \in \{+1, -1\}$ is the class label associated with example $x_i \in \mathbb{R}^n$. A stronger classifier is a linear combination of M weak classifiers:

$$H_M(x) = \sum_{m=1}^{M} h_m(x). \tag{33}$$

In the real version of AdaBoost [25, 26], the weak classifiers can take a real value, $h_m(x) \in \mathbb{R}$, and have absorbed the coefficients needed in the discrete version ($h_m(x) \in -1, +1$ in the latter case). The class label for x is obtained as $H(x) = \text{sign}[H_M(x)]$, while magnitude $|H_M(x)|$ indicates the confidence. Every training example is associated with a weight. During the learning process, the weights are updated dynamically in such a way that more emphasis is placed on hard examples that are erroneously classified previously. It has been noted in recent studies [28, 29, 30] that the artificial operation of explicit re-weighting is unnecessary and can be incorporated into a functional optimization procedure of boosting.

An error occurs when $H(x) \neq y$, or $yH_M(x) < 0$. The "margin" of an example, (x, y), achieved by $h(x) \in \mathbb{R}$ on the training set examples is defined as $yh(x)$. This can be considered a measure of the confidence of h's prediction. The upper bound of classification error achieved by H_M can be derived as the following exponential loss function [31]:

$$J(H_M) = \sum_i \mathrm{e}^{-y_i H_M(x_i)} = \sum_i \mathrm{e}^{-y_i \sum_{m=1}^{M} h_m(x)}. \tag{34}$$

AdaBoost constructs $h_m(x)$ by stagewise minimization of Eq. (34). Given the current $H_{M-1}(x) = \sum_{m=1}^{M-1} h_m(x)$, the best $h_M(x)$ for the new strong classifier, $H_M(x) = H_{M-1}(x) + h_M(x)$ is the one that leads to the minimum cost:

$$h_M = \arg\min_{h^\dagger} J(H_{M-1}(x) + h^\dagger(x)). \tag{35}$$

0. (Input)
   (1) Training examples $\{(x_1, y_1), \ldots, (x_N, y_N)\}$,
   where $N = a + b$; of which a examples have $y_i = +1$
   and b examples have $y_i = -1$;
   (2) The maximum number $M_{\max}$ of weak classifiers to be combined;
1. (Initialization)
   $w_i^{(0)} = \frac{1}{2a}$ for those examples with $y_i = +1$ or
   $w_i^{(0)} = \frac{1}{2b}$ for those examples with $y_i = -1$.
   $M = 0$;
2. (Forward Inclusion)
   while $M < M_{\max}$
   (1) $M \leftarrow M + 1$;
   (2) Choose h_M according to Eq.36;
   (3) Update $w_i^{(M)} \leftarrow \exp[-y_i H_M(x_i)]$, and normalize to $\sum_i w_i^{(M)} = 1$;
3. (Output)
   $H(x) = \text{sign}[\sum_{m=1}^{M} h_m(x)]$.

Figure 9. AdaBoost algorithm. Reprinted with permission from XS Huang, SZ Li, YS Wang. 2004. Statistical learning of evaluation function for ASM/AAM image alignment. In *Proceedings: Biometric Authentication, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004* (ECCV Workshop BioAW), pp. 45–56. Ed D Maltoni, AK Jain. New York: Springer. Copyright ©2004, Springer.

The minimizer is [25, 26]

$$h_M(x) = \frac{1}{2} \log \frac{P(y = +1|x, w^{(M-1)})}{P(y = -1|x, w^{(M-1)})}, \tag{36}$$

where $w^{(M-1)}(x, y) = \exp(-yF_{M-1}(x))$ is the weight for the labeled example (x, y) and

$$P(y = +1|x, w^{(M-1)}) = \frac{E\left(w(x, y) \cdot 1_{[y=+1]}|x\right)}{E\left(w(x, y) \mid x\right)}, \tag{37}$$

where $E(\cdot)$ stands for the mathematical expectation and $1_{[C]}$ is 1 if C is true or 0 otherwise. $P(y = -1|x, w^{(M-1)})$ is defined similarly.

The AdaBoost algorithm based on the descriptions from [25, 26] is shown in Figure 9. There, the re-weight formula in step 2.3 is equivalent to the multiplicative rule in the original form of AdaBoost [32, 25]. In Section 6.3, we will present a statistical model for stagewise approximation of $P(y = +1|x, w^{(M-1)})$.

6.3. Construction of Candidate Weak Classifiers

The optimal weak classifier at stage M is derived as Eq. (36). Using $P(y|x,w)= p(x|y,w)P(y)$, it can be expressed as

$$h_M(x) \quad = \quad L_M(x) - T, \tag{38}$$

where

$$L_M(x) \quad = \quad \frac{1}{2}\log\frac{p(x|y=+1,w)}{p(x|y=-1,w)}, \tag{39}$$

$$T \quad = \quad \frac{1}{2}\log\frac{P(y=+1)}{P(y=-1)}. \tag{40}$$

The log likelihood ratio (LLR), $L_M(x)$, is learned from the training examples of the two classes. The threshold T is determined by the log ratio of prior probabilities. In practice, T can be adjusted to balance between the detection and false alarm rates (i.e., to choose a point on the ROC curve).

Learning optimal weak classifiers requires modeling the LLR of Eq. (39). Estimating the likelihood for high-dimensional data x is a non-trivial task. In this work, we make use of the stagewise characteristics of boosting, and derive likelihood $p(x|y,w^{(M-1)})$ based on an over-complete scalar feature set $\mathcal{Z} = \{z'_1,\ldots,z'_K\}$. More specifically, we approximate $p(x|y,w^{(M-1)})$ by $p(z_1, \ldots, z_{M-1}, z'|y, w^{(M-1)})$, where z_m $(m = 1,\ldots,M-1)$ are the features that have already been selected from $\mathcal{Z}$ by the previous stages, and z' is the feature to be selected. The following describes the candidate feature set $\mathcal{Z}$, and presents a method for constructing weak classifiers based on these features.

Because the shape is about boundaries between regions, it makes sense to use edge information (magnitude or orientation or both) extracted from a grayscale image. In this work, we use a simple Sobel filter for extracting the edge information. Two filters are used: K_w for horizontal edges and K_h for vertical edges, as follows:

$$K_w(w,h) = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \quad and \quad K_h(w,h) = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}. \tag{41}$$

The convolution of the image with the two filter masks gives two edge strength values:

$$G_w(w,h) = K_w * I(w,h), \tag{42}$$

$$G_h(w,h) = K_h * I(w,h), \tag{43}$$

The edge magnitude and direction are obtained as

$$S(w,h) = \sqrt{G_w^2(w,h) + G_h^2(w,h)}, \tag{44}$$

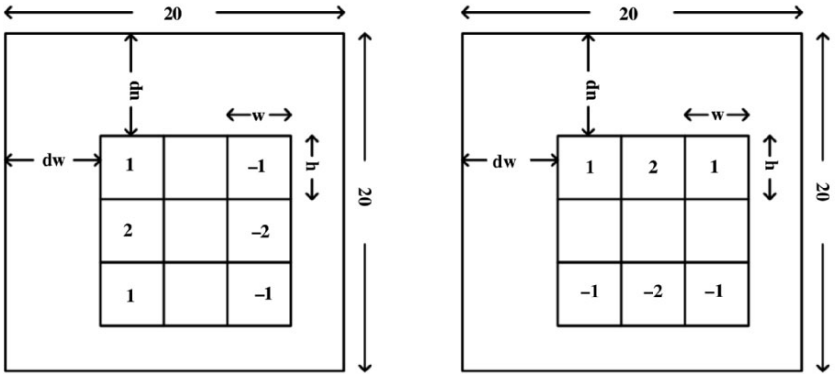


Figure 10. The two types of simple Sobel-like filters defined on sub-windows. The rectangles are of size $w \times h$ and are at distances of (dw, dh) apart. Each feature takes a value calculated by the weighted $(\pm 1, \pm 2)$ sum of the pixels in the rectangles. Reprinted with permission from XS Huang, SZ Li, YS Wang. 2004. Statistical learning of evaluation function for ASM/AAM image alignment. In *Proceedings: Biometric Authentication, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004* (ECCV Workshop BioAW), pp. 45–56. Ed D Maltoni, AK Jain. New York: Springer. Copyright ©2004, Springer.

$$\phi(w, h) = \arctan\left(\frac{G_h(w, h)}{G_w(w, h)}\right). \tag{45}$$

The edge information based on the Sobel operator is sensitive to noise. To solve this problem we use the sub-block of the image to convolve with the Sobel filter (see Figure 10), which is similar to Haar-like feature calculation.

6.4. Statistical Learning of Weak Classifiers

A scalar feature $z'_k : x \to \mathbf{R}$ is a transform from the n-dimensional (400D if a face example x is of size 20 $\times$ 20) data space to the real line. These block differences are an extension to the Sobel filters. For each face example of size 20 $\times$ 20, there are hundreds of thousands of different z'_k for admissible w, h, dw, dh values, so $\mathcal{Z}$ is an over-complete feature set for the intrinsically low-dimensional face pattern x. In this work, an optimal weak classifier (38) is associated with a single scalar feature; to find the best new weak classifier is to choose the best corresponding feature.

We can define the following component LLRs for target $L_M(x)$:

$$\tilde{L}_m(x) \quad = \quad \frac{1}{2} \log \frac{p(z_m | y = +1, w^{(m-1)})}{p(z_m | y = -1, w^{(m-1)})} \tag{46}$$

for the selected features, z_m's ($m = 1, \ldots, M-1$), and

$$L_k^{(M)}(x) = \frac{1}{2} \log \frac{p(z'_k(x)|y=+1, w^{(M-1)})}{p(z'_k(x)|y=-1, w^{(M-1)})} \tag{47}$$

for features to be selected, $z'_k \in \mathcal{Z}$. Then, after some mathematical derivation, we can approximate the target LLR function as

$$L_M(x) = \frac{1}{2} \log \frac{p(x|y=+1, w^{(M-1)})}{p(x|y=-1, w^{(M-1)})} \approx \sum_{m=1}^{M-1} \tilde{L}_m(x) + L_k^{(M)}(x). \tag{48}$$

Let

$$\Delta L_M(x) = L_M(x) - \sum_{m=1}^{M-1} \tilde{L}_m(x). \tag{49}$$

The best feature is the one whose corresponding $L_k^{(M)}(x)$ best fits $\Delta L_M(x)$. It can be found as the solution to the following minimization problem:

$$k^* = \arg\min_{k,\beta} \sum_{i=1}^{N} \left[\Delta L_M(x_i) - \beta L_k^{(M)}(x_i)\right]^2. \tag{50}$$

This can be done in two steps as follows. First, find k^* for which

$$(L_k^{(M)}(x_1), L_k^{(M)}(x_2), \ldots, L_k^{(M)}(x_N)) \tag{51}$$

is most parallel to

$$(\Delta L_M(x_1), \Delta L_M(x_2), \ldots, \Delta L_M(x_N)). \tag{52}$$

This amounts to finding k for which $L_k^{(M)}$ is most correlated with ΔL_M over the data distribution, and set $z_M = z'_{k^*}$. Then, we compute

$$\beta^* = \frac{\sum_{i=1}^{N} \Delta L_M(x_i) L_{k^*}(x_i)}{\sum_{i=1}^{N} [L_{k^*}(x_i)]^2}. \tag{53}$$

After that, we obtain

$$\tilde{L}_M(x) = \beta^* L_{k^*}(x). \tag{54}$$

The strong classifier is then given as

$$H_M(x) = \sum_{m=1}^{M} \left(\tilde{L}_m(x) - T\right) \tag{55}$$

$$= \sum_{m=1}^{M} \tilde{L}_m(x) - MT. \tag{56}$$

The evaluation function $H_M(x)$ thus learned gives a quantitative confidence and the good–bad classification is achieved by comparing the confidence with the threshold value of zero.

7. EXPERIMENTAL RESULTS

7.1. DAM

7.1.1. Computation of Subspaces

A total of 80 images of size 128×128 are collected. Each image contains a different face in an area of about 64×64 pixels. The images set are randomly partitioned into a training set of 40 images and a test set of the other 40. Each image is mirrored, and this doubles the total number of images in each set.

$K = 72$ face landmark points are labeled manually (see an example in Figure 11. The shape subspace is $k = 39$ dimensional, which retains 98% of the total shape variation. The mean shape contains a texture of $L = 3186$ pixels. The texture subspace is $\ell = 72$ dimensional, as the result of retaining 98% of total texture variation. These are common to both the AAM and DAM.

For the AAM, an appearance subspace is constructed to combine both shape and texture information. A concatenated shape and texture vector is 39 + 72 dimensional, where the weight parameter is calculated as $r = 7.5$ for $\mathbf{\Lambda} = r\mathbf{I}$ in Eq. (7). It is reduced to a 65-dimensional appearance subspace that retains 98% of total variation of the concatenated features.

For the DAM, the linearity assumption made for the model, $s = \mathbf{R}t + \varepsilon$, of Eq. (16) is well verified because all the elements in $E(\varepsilon\varepsilon^T)$ calculated over the training set are smaller than 10^{-5}.

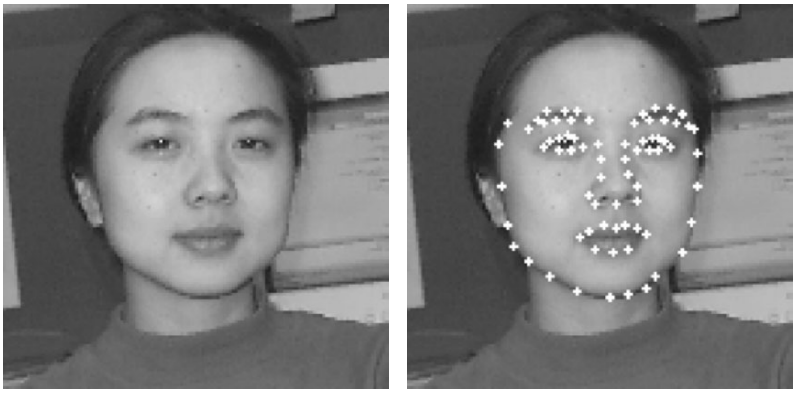


Figure 11. A face image and the landmark points. Reprinted with permission from XW Hou, SZ Li, HJ Zhang, QS Cheng. 2001. Direct appearance models. *Proc IEEE Conf Comput Vision Pattern Recogn* **1**:828–833. Copyright ©2001, IEEE.

The original texture difference δT, which is used in the AAM for predicating position displacement, is 3186 dimensional; it is reduced to 724-dimensional $\delta T'$, which is used in the DAM for prediction, to retain 98% of variation over the 1920 training examples.

The DAM requires much less memory during learning of prediction matrices $\mathbf{R}_p$ in Eq. (22) than AAM for learning $\mathbf{A}_a$ in Eq. (11). For the DAM, there are 80 training images, 4 parameters for the position $(x, y, \theta, scale)$, and 6 disturbances for each parameter to generate training data for training $\mathbf{R}_p$. So, the size of training data for the DAM is $80 \times 4 \times 6 = 1920$. For the AAM, there are 80 training images, 65 appearance parameters, and 4 disturbances for each parameter to generate training data for training $\mathbf{A}_a$. The size of the training data set for $\mathbf{A}_a$ is $80 \times 65 \times 4 = 20800$. Therefore, the size of the training data set for AAM's prediction matrices is $20800 + 1920 = 22720$, which is 11.83 times that for the DAM. On a PC, for example, the memory capacity for AAM training with 80 images would allow DAM training with 946 images.

7.1.2. Alignment and Appearance Estimation

Table 2 compares the DAM and AAM in terms of the quality of position and texture parameter estimates, and the convergence rates. The effect of using $\delta T'$ instead of δT is demonstrated through DAM$'$, which is DAM minus the PCA subspace modeling of δT. The initial position is a shift from the true position by $dx = 6, dy = 6$. The $\|\delta p\|$ is calculated for each image as the averaged distance between corresponding points in the two shapes, and therefore it is also a measure

Table 2. Comparisons of DAM, DAM' and AAM in terms of errors in estimated texture (appearance) parameters δT and position δp and convergence rates for the training images (first block of three rows) and test images (second block)

	$E(\|\delta T\|^2)$	std$(\|\delta T\|^2)$	$E(\|\delta p\|)$	std$(\|\delta p\|)$	cvg rate
DAM	0.156572	0.065024	0.986815	0.283375	100%
DAM'	0.155651	0.058994	0.963054	0.292493	100%
AAM	0.712095	0.642727	2.095902	1.221458	70%
DAM	1.114020	4.748753	2.942606	2.023033	85%
DAM'	1.180690	5.062784	3.034340	2.398411	80%
AAM	2.508195	5.841266	4.253023	5.118888	62%

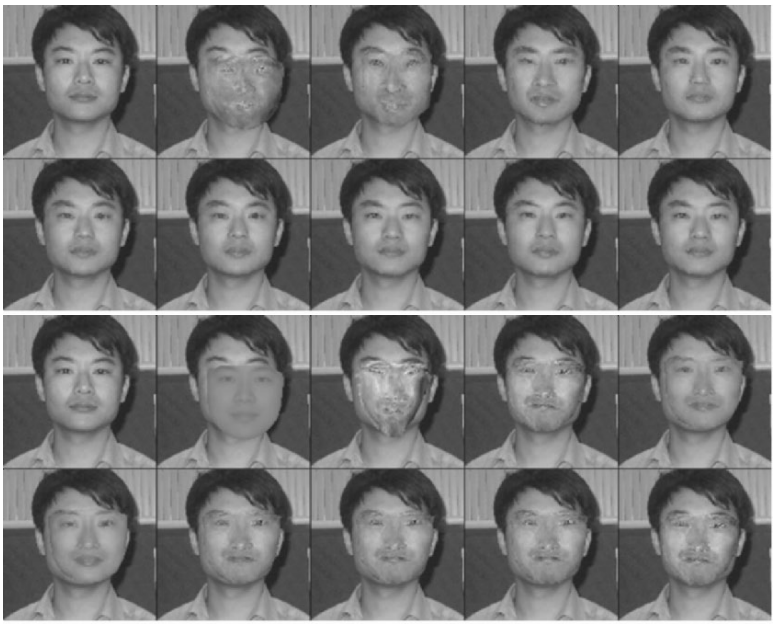


Figure 12. Scenarios of DAM (top) and AAM (bottom) alignment. Reprinted with permission from XW Hou, SZ Li, HJ Zhang, QS Cheng. 2001. Direct appearance models. *Proc IEEE Conf Comput Vision Pattern Recogn* **1**:828–833. Copyright ©2001, IEEE.

of difference in shape. The convergence is judged by satisfaction of two conditions: $\|\delta T\|^2 < 0.5$ and $\|\delta p\| < 3$.

Figure 12 illustrates average scenarios of DAM and AAM alignment. Figure 13 illustrates the dynamics of total error δT for 10 images randomly selected from the training set and 10 from the test set. We see that the DAM has faster convergence and smaller error than the AAM.

7.1.3. Multi-View DAM

The training set contains 200 frontal, 200 half-side, and 170 full-side view faces whose sizes are of about 64×64 pixels, while the test set contains 80 images for each view group. The landmark points are labeled manually (see Figure 2 and Table 1). They are used for the training and as ground-truth in the test stage.

To compare, we also implemented the AAM using the same data in the frontal view. The shape and texture parameter vectors are 69 + 144 dimensional, respectively, where the weight parameter for the concatenation of the two parts is calculated as $r = 8.84$ for $\mathbf{\Lambda} = r\mathbf{I}$ in Eq. (7). The concatenated vector space is reduced to a 113-dimensional appearance subspace that retains 98% of the total variation of the concatenated features.

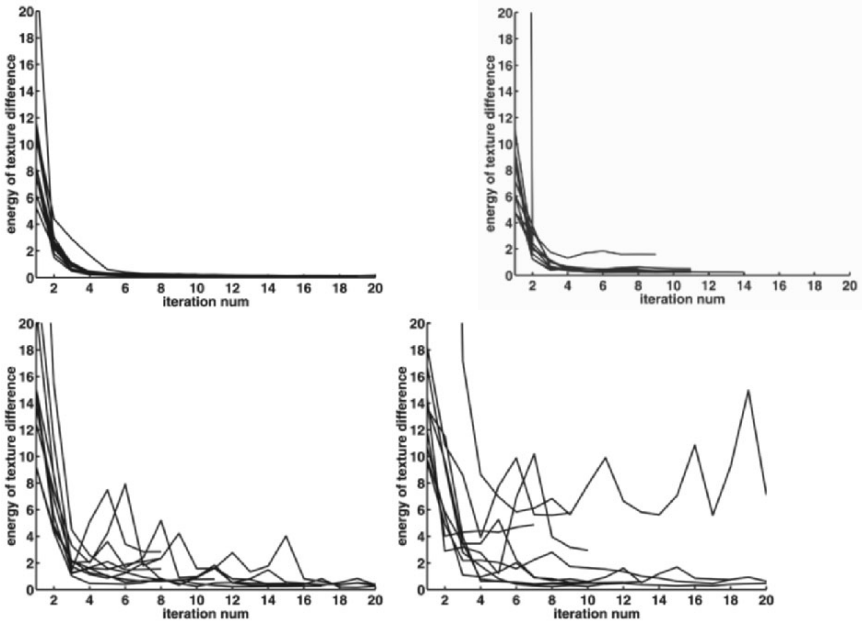


Figure 13. The evolution of total δT for the DAM (top) and AAM (bottom) as a function of iteration number for the training (left) and test (right) images. Reprinted with permission from XW Hou, SZ Li, HJ Zhang, QS Cheng. 2001. Direct appearance models. *Proc IEEE Conf Comput Vision Pattern Recogn* **1**:828–833. Copyright ©2001, IEEE.

Some results about DAM learning and search have been presented in Figure 2–6. Figure 14 compares the convergence rate and accuracy properties of the DAM and AAM (for the frontal view) in terms of the error in δT (cf. Eq. (10)) as the algorithms iterate. The statistics are calculated from 80 images randomly selected from the training set and 80 images from the test set. We can see that the DAM has faster a convergence rate and smaller error than the AAM. Figure 15 illustrates the error of DAM for non-frontal faces. Figure 16 compares the alignment accuracy of the DAM and AAM (for frontal faces) in terms of the percentage of images whose texture reconstruction error δT is smaller than 0.2, where the statistics are obtained using another test set including the 80 test images mentioned above and an additional 20 other test images. It shows again that the DAM is more accurate than the AAM.

The DAM search is fairly fast. It takes on average 39 ms per iteration for frontal and half-side view faces, and 24 ms for full-side view faces in an image of size 320×240 pixels. Every view model takes about 10 iterations to converge. If 3 view models are searched per face, as is done with image sequences from video, the algorithm takes about 1 second to find the best face alignment.

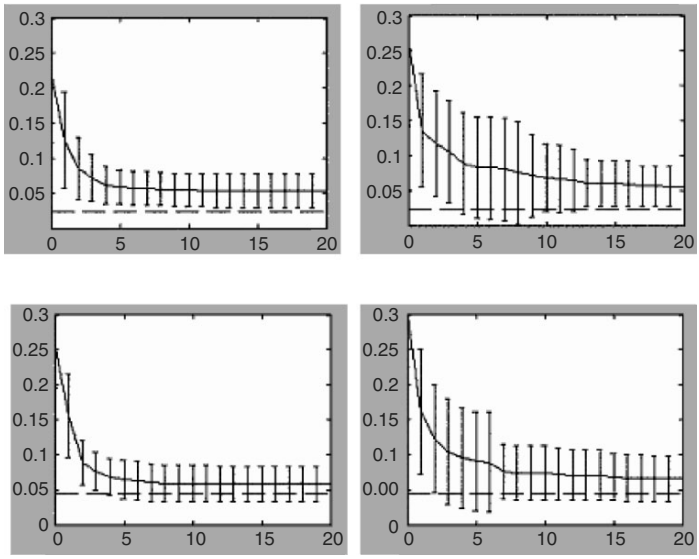


Figure 14. Mean error (the curve) and standard deviation (the bars) in reconstructed texture $\|\delta T\|$ as a function of iteration number for DAM (left) and AAM (right) methods with the training (top) and test (bottom) sets, for frontal face images. The horizontal dashed lines in the lower part of the figures indicate average $\|\delta T\|$ for the manually labeled alignment. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309– 314. Copyright ©2002, IEEE.

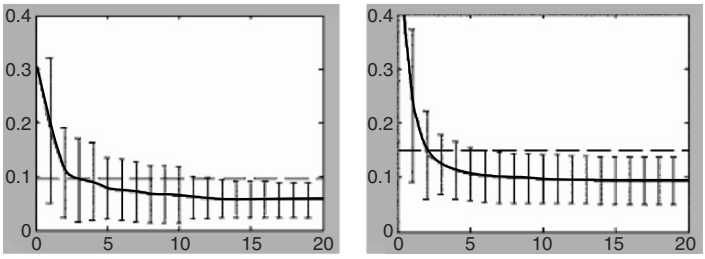


Figure 15. Mean error in $\|\delta T\|$ and standard deviation of DAM alignment for half- (left) and full- (right) side view face images from the test set. Note that the mean errors in the calculated solutions are smaller than obtained using the manually labeled alignment after a few iterations. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309–314. Copyright ©2002, IEEE.

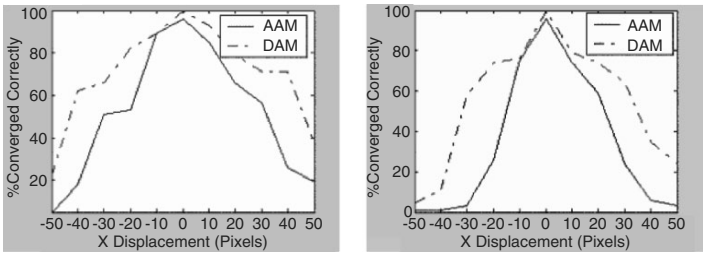


Figure 16. Alignment accuracy of the DAM (dashed) and AAM (solid) in terms of localization errors in the x (left) and y (right) directions. Reprinted with permission from SZ Li, SC Yan, HJ Zhang, QS Cheng. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn*, pp. 309–314. Copyright ©2002, IEEE.

7.2. TC-ASM

A data set containing 700 face images with different illumination conditions and expressions are selected from the AR database [33] in our experiments, each of which is 512×512, 256 gray images containing the frontal view face about 200×200. 83 landmark points are manually labeled on the face. We randomly select 600 for training and the other 100 for testing.

For comparison, the ASM and AAM are trained on the same data sets, in a three-level image pyramid (resolution is reduced 1/2 level by level) as with the TC-ASM. By means of PCA with 98% total variations retained, the dimension of the shape parameter in the ASM shape space is reduced to 88, and the texture parameter vector in the AAM texture space is reduced to 393. The concatenated vector of the shape and texture parameter vectors with the weighting parameter, $\gamma = 13.77$, is reduced to 277. Two types of experiments are presented: (1) comparison of the point-position accuracy, and (2) comparison of the texture reconstruction error. The experiments are all performed in the 3-level resolution image pyramid.

7.2.1. Point Position Accuracy

The average point-point distances between the searched shape and the manually labeled shape of the three models are compared in Figure 17. The vertical axis represents the percentage of the solutions for which the average point-to-point distances to the manually labeled ones are smaller than the corresponding horizonal axis value. The statistics are calculated from 100 test images with different initializations, with random displacements to the ground truth of 10, 20, 30, and 40 pixels. The results show that TC-ASM outperforms both the ASM and AAM in most cases since the TC-ASM curve lies above the ASM and AAM curves. It also suggests that the AAM outperforms the ASM when the initial displacement is small, while the ASM is more robust with an increasing initial displacement.

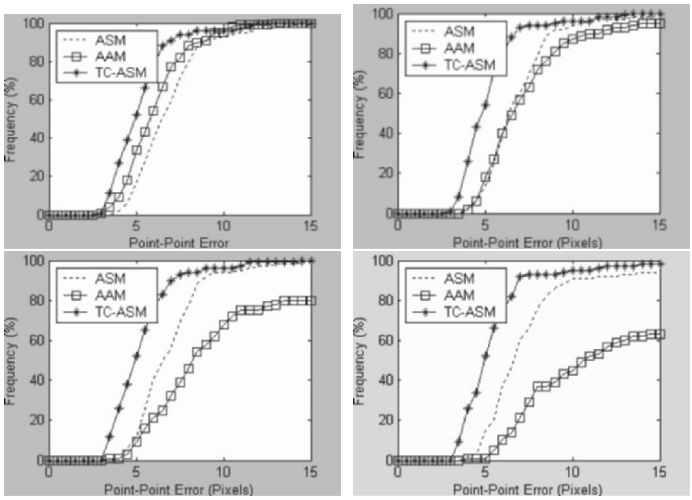


Figure 17. Accuracy of ASM, AAM, TC-ASM. From upper to lower, left to right, are the results obtained with the initial displacements of 10, 20, 30, and 40 pixels. Note that the value of the vertical coordinate is the percentage of examples that have the point-to-point distance smaller than the corresponding value of horizonal coordinate. Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier.

We compare the stability of the TC-ASM with the ASM in Figure 18. The value on the horizonal axis is the index number of the selected examples, whereas the value on the vertical axis is the average standard deviation of the results obtained from 10 different initializations that deviate from the ground-truth by approximately 20 pixels. The results are convincing that the TC-ASM is more stable to initialization. An example is given in Figure 19.

7.2.2. Texture Reconstruction Error

The texture reconstruction error comparison of the three models in Figure 20 illustrates that the TC-ASM improves the accuracy of texture matching. The texture accuracy of the TC-ASM is close to that of the AAM, while its position accuracy is better than that of the AAM (see Figure 17). Although the AAM has more cases with small texture reconstruction errors, the TC-ASM has more cases with a texture reconstruction error smaller than 0.2.

An example in which the AAM fails for a different illumination condition from the training data, yet the TC-ASM performs well, is presented in Figure 21. Figure 22 shows a scenario of AAM and TC-ASM alignment.

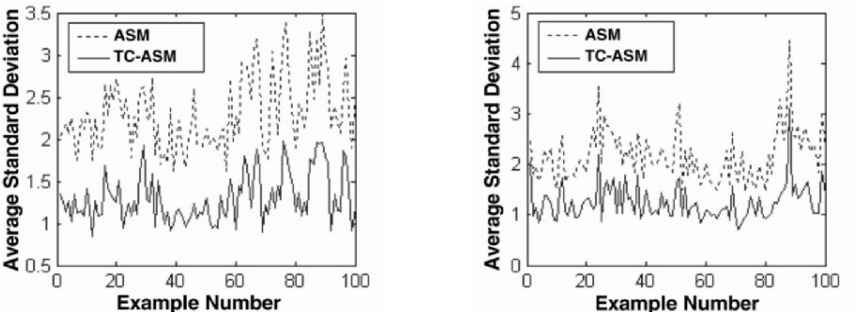


Figure 18. Standard deviation in the results of each example for ASM (dotted) and TC-ASM (solid) with the training set (left) and the test set (right). Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier.

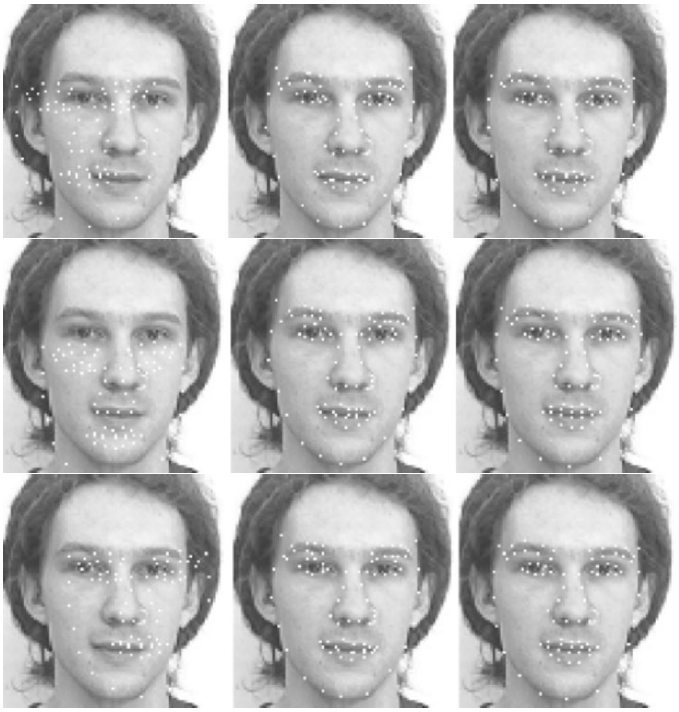


Figure 19. Stability of the ASM (middle column) and the TC-ASM (right column) in shape localization. The different initialization conditions are shown in the left column. Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier.

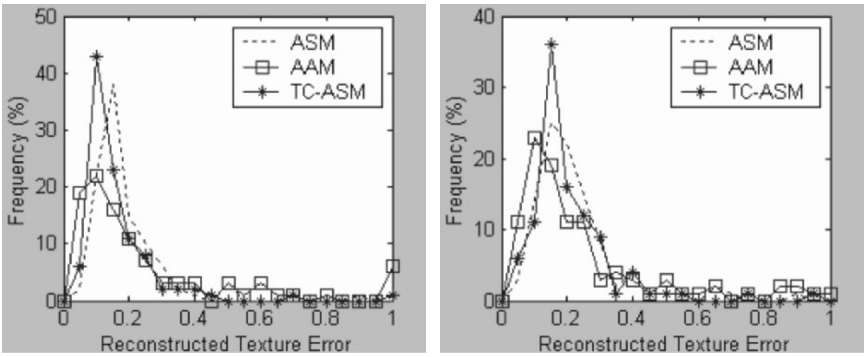


Figure 20. Distribution of the texture reconstruction error with the ASM (dotted), the AAM (square), and the TC-ASM (asterisk), with training data (left) and test data (right). Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier.

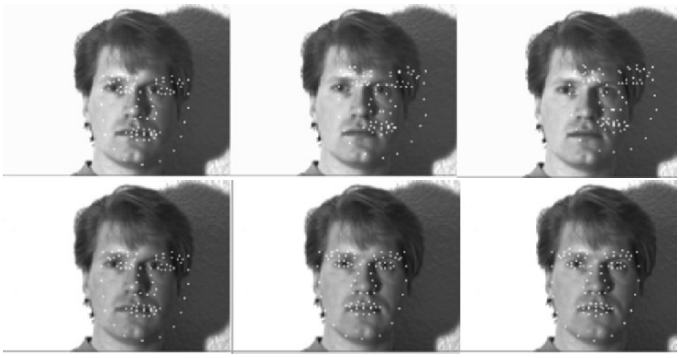


Figure 21. Sensitivities of the AAM (upper) and TC-ASM (lower) to an illumination condition not seen in the training data. From left to right are the results obtained at the 0th, 2th, and 10th iterations. Result in different levels of image pyramid is scaled back to the original scale. Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier.

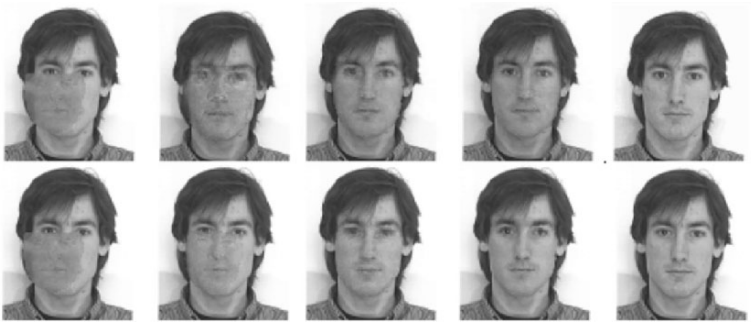


Figure 22. Scenarios of AAM (upper) and TC-ASM (lower) alignment with texture reconstruct errors 0.3405 and 0.1827, respectively. From left to right are the results obtained at the 0th, 5th, 10th, and 15th iterations, and the original image. Result in different levels of image pyramid is scaled back to the original scale. Reprinted with permission from SC Yan, C Liu, SZ Li, HJ Zhang, H Shum, QS Cheng. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75. Copyright ©2003, Elsevier.

From the experiment, the TC-ASM is more computationally expensive than ASM, but it is much faster than the AAM. In our experiment (600 training images, 83 landmarks, using a P-III 667 computer with 256Mb memory), it takes an average of 32 ms per iteration, which is twice that of the ASM (16 ms) but a fifth of the AAM (172 ms). The training time with the AAM is more than 2 hr, while for the TC-ASM it is only about 12 minutes.

7.3. Evaluation for Face Alignment

The positive and negative training and set data are generated as follows. All the shapes are aligned or are warping to the tangent space of the mean shape, $\overline{S}$. After that, the texture T_0 is warped correspondingly to $T \in \mathbb{R}^L$, where L is the number of pixels in the mean shape $\overline{S}$.

In our work, 2536 positive examples and 3000 negative examples are used to train a strong classifier. The 2536 positive examples are derived from 1268 original positive examples plus the mirror images. The negative examples are generated by random rotating, scaling, and shifting positive example shape points. A strong classifier is trained to reject 92% of the negative examples while correctly accepting 100% of the positive examples.

A cascade of classifiers is trained to train a computationally effective model, and makes training easier with a divide-and-conquer strategy. When training a new stage, negative examples are bootstrapped based on the classifier trained in the previous stages. The details of training a cascade of 5 stages is summarized

Table 3. Training results (WC: weak classifier)

stage	number of pos	number of neg	number of WC	False Alarm
1	2536	3000	22	0.076
2	2536	3000	237	0.069
3	2536	888	294	0.263
4	2536	235	263	0.409
5	2536	96	208	0.0

in Table 3. As the result of training, we achieved 100% correct acceptance and correct rejection rates on the training set.

We compare the learned evaluation function with the PCA texture reconstruction error-based evaluation method, using the same data sets (but PCA does not require negative examples in training). The dimensionality of the PCA subspace is chosen to retain 99% of the total variance of the data. The best threshold of reconstruction error is selected to minimize the classification error. Figure 23 shows the ROC curve for the reconstruction error-based alignment evaluation method for the training set. Note that this method cannot achieve 100% correct rates.

During the test, a total of 1528 aligned examples (800 qualified and 728 non-qualified images) are used. We evaluate each face images and give a score in terms of (a) the confidence value $H_M(x)$ for the learning-based method and (b) the confidence value $\mathrm{dist}_{PCA} - \mathrm{threshold}$ for the PCA-based method. The qualified and unqualified alignment decisions are judged by comparing the score with the normalized threshold of 0. Some examples of accepted (top part) and rejected (bottom part) face alignment results are shown in Figure 24. Figure 25 compares the two methods in terms of their ROC curves (first plot) and error curves (the second plot), where the axis label P (pos/neg) represents the false positive rate and so on.

Finally, we would like to mention that, experimentally, we also found that the Sobel features produced significantly better results than other features such as Haar wavelets. This is not elaborated here.

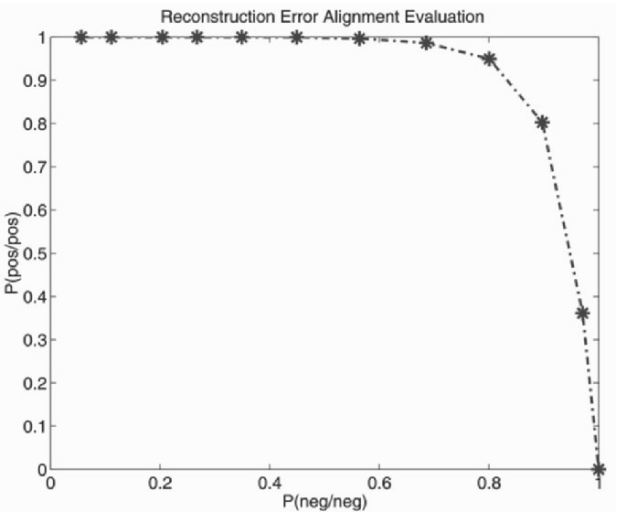


Figure 23. ROC curve for the reconstruction error-based alignment evaluation for the training set. Reprinted with permission from XS Huang, SZ Li, YS Wang. 2004. Statistical learning of evaluation function for ASM/AAM image alignment. In *Proceedings: Biometric Authentication, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004* (ECCV Workshop BioAW), pp. 45–56. Ed D Maltoni, AK Jain. New York: Springer. Copyright ©2004, Springer. See attached CD for color version.

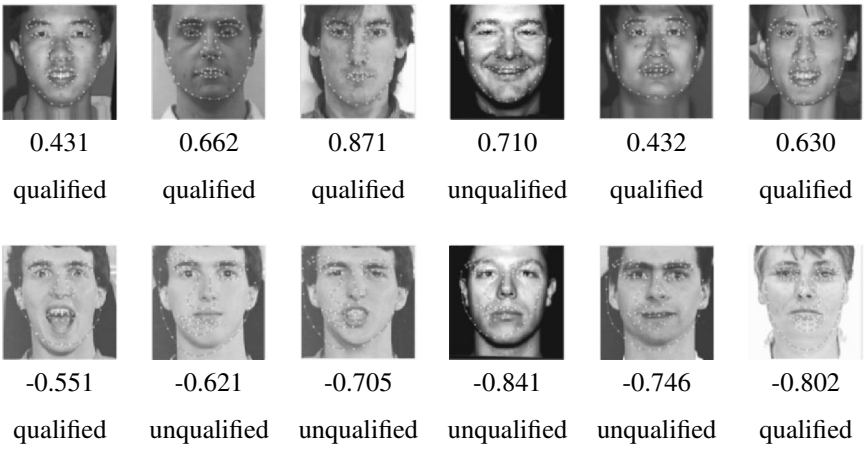


Figure 24. Alignment quality evaluation results: accepted images (top) and rejected images (bottom). Reprinted with permission from XS Huang, SZ Li, YS Wang. 2004. Statistical learning of evaluation function for ASM/AAM image alignment. In *Proceedings: Biometric Authentication*, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004 (ECCV Workshop BioAW), pp. 45–56. Ed D Maltoni, AK Jain. New York: Springer. Copyright ©2004, Springer. See attached CD for color version.

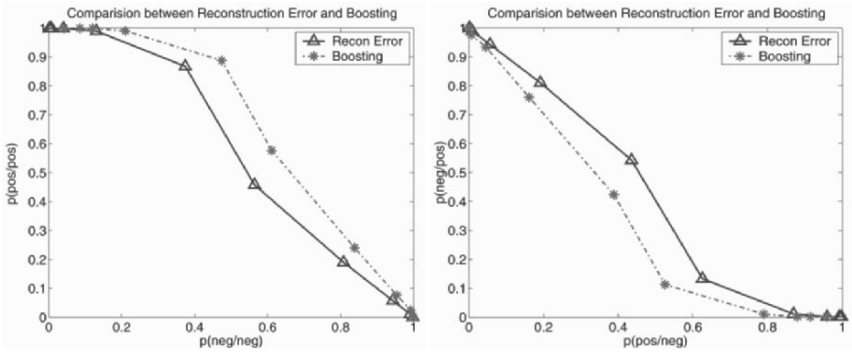


Figure 25. Comparison between reconstruction error method and boost method. Reprinted with permission from XS Huang, SZ Li, YS Wang. 2004. Statistical learning of evaluation function for ASM/AAM image alignment. In *Proceedings: Biometric Authentication, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004* (ECCV Workshop BioAW), pp. 45–56. Ed D Maltoni, AK Jain. New York: Springer. Copyright ©2004, Springer. See attached CD for color version.

8. CONCLUSION

In this chapter we reviewed important shape- and texture-based deformable models — such as the ASM, the AAM, and their variants — for image analysis. These image analysis tools not only provide alignment between the input and the target to best fit the constraints, but also provide aligned features for object pattern classification.

Although great advances have been made in the past decade, there remain challenges for future research. One area is the robustness of deformable models toward variance of pose, illumination, and expression. Existing models can only deal with a moderate amount of such variations, so performance deteriorates when extreme illumination conditions or exaggerated expressions are present. While a morphable model has demonstrated its effectiveness with 3D object analysis, efficient, real-time, and exact model searching algorithms are still lacking. Solving these problems will lead to better applications.

9. ACKNOWLEDGMENTS

This work was supported by the following funding: National Science Foundation of China Project #60518002, Chinese National 863 Program Projects #2004AA1Z2290 and #2004AA119050.

10. NOTES

1. It is a deviation of the mostly used energy function with a squared Euclidean distance between S_{lm}^n and shape $S \in \mathbb{R}^{2K}$ derived from parameter s. It is more reasonable to take into account the prior distribution in the shape space.

11. REFERENCES

1. Cootes TF, Taylor CJ, Cooper DH, Graham J. 1995. Active shape models: their training and application. *Comput Vision Image Understand* **61**:38–59.
2. Cootes TF, Edwards GJ, Taylor CJ. 1998. Active appearance models. In *Proceedings of the European conference on computer vision*, Vol. 2, pp. 484–498. Ed. H Burkhardt, B Neumann. New York: Springer.
3. Edwards GJ, Cootes TF, Taylor CJ. 1998. Face recognition using active appearance models. In *Proceedings of the European conference on computer vision*, Vol. 2, pp. 581–695. Ed. H Burkhardt, B Neumann. New York: Springer.
4. Cootes TF, Taylor CJ. 2001. *Statistical models of appearance for computer vision*. Technical Report, Wolfson Image Analysis Unit, Manchester University, www.isbe.man.ac.uk/simbim/refs.html.
5. Sclaroff S, Isidoro J. 1998. Active blobs. In *Proc IEEE Int Conf Comput Vision, Bombay, India*, pp. 1146–1153.
6. Cootes TF, Walker KN, Taylor CJ. 2000. View-based active appearance models. In *4th International conference on automatic face and gesture recognition, Grenoble, France*, pp. 227–232. http://citeseer.ist.psu.edu/cootes00viewbased.html.
7. Psarrou A, Romdhani S, Gong S. 1999. Learning a single active face shape model across views. In *Proceedings of the IEEE international workshop on recognition, analysis, and tracking of faces and gestures in real-time systems, Corfu, Greece, 26–27 September*, pp. 31–38. Washington, DC: IEEE Computer Society.
8. Cootes TF, Taylor CJ. 2001. Constrained active appearance models. *Proc IEEE Int Conf Comput Vision* **1**:748–754.
9. Blanz V, Vetter T. 1999. A morphable model for the synthesis of 3D faces. In *Proc. Siggraph'99*, pp. 187–194. New York: ACM Press.
10. Li Y, Gong S, Liddell and H. 2001. Constructing facial identity surfaces in a nonlinear discriminating space. *Proc IEEE Comput Soc Conf: Computer Vision and Pattern Recognition* **2**:258–263.
11. Duta N, Jain AK, Dubuisson-Jolly M. 2001. Automatic construction of 2D shape models. *IEEE Trans Pattern Analy Machine Intell* **23**(5):433–446.
12. van Ginneken B, Frangi AF, Staal JJ, ter Haar Romeny BM, Viergever MA. 2001. A nonlinear gray-level appearance model improves active shape model segmentation. In *IEEE workshop on mathematical models in biomedical image analysis*, pp. 205–212. Ed. L Staib, A Rangarajan. Washington, DC: IEEE Society Press.
13. Baker S, Matthews I. 2001. Equivalence and efficiency of image alignment algorithms. *Proc IEEE Conf Comput Vision Pattern Recognition* **1**:1090–1097.
14. Ahlberg J. 2001. Using the active appearance algorithm for face and facial feature tracking. In *IEEE ICCV workshop on recognition, analysis and tracking of faces and gestures in real-time systems, Vancouver, Canada, July 13, 2001*, pp. 68–72. Washington, DC: IEEE.
15. Hou XW, Li SZ, Zhang HJ, Cheng QS. 2001. Direct appearance models. *Proc IEEE Conf Comput Vision Pattern Recognition* **1**:828– 833.
16. Li SZ, Yan SC, Zhang HJ, Cheng QS. 2002. Multi-view face alignment using direct appearance models. *Proc 5th Int Conf Automatic Face Gesture Recogn, Washington, DC, 20–21 May 2002*, pp. 309–314. Washington, DC: IEEE.

17. Yan SC, Liu C, Li SZ, Zhang HJ, Shum H, Cheng QS. 2003. Face alignment using texture-constrained active shape models. *Image Vision Comput* **21**(1):69–75.
18. XS Huang, SZ Li, YS Wang. 2004. Statistical learning of evaluation function for ASM/AAM image alignment. In *Proceedings: Biometric Authentication, ECCV 2004 International Workshop, BioAW 2004, Prague, Czech Republic, May 15, 2004* (ECCV Workshop BioAW), pp. 45–56. Ed D Maltoni, AK Jain. New York: Springer.
19. Cootes TF, Edwards GJ, Taylor CJ. 1999. Comparing active shape models with active appearance models. In *10th British Machine Vison Conference*, Vol. 1, pp. 173–182. Ed. T Pridmore, D Elliman. Nottingham, UK: BMVA Press. http://citeseer.ist.psu.edu/article/cootes99comparing.html.
20. Basso C, Romdhani S, Blanz V, Vetter T. 2005. Morphable models of faces. In *Handbook of face recognition*, pp. 217–245. Ed S Li, A Jain. New York: Springer.
21. Blanz V, Vetter T. 2003. Face recognition based on fitting a 3D morphable model. *IEEE Trans Pattern Anal Machine Intell* **25**(9):1063- -1074.
22. Jones MJ, Poggio T. 1998. Multidimensional morphable models. *Proc IEEE Int Conf Comput Vision*, pp. 683–688. http://citeseer.ist.psu.edu/jones98multidimensional.html.
23. Bergen JR, Hingorani R. 1990. *Hierarchical motion-based frame rate conversion*. Technical Report, David Sarnoff Research Center, Princeton, NJ.
24. Li S, Zhang ZQ, Zhu L, Zhang HJ. 2002. Real-time multi-view face detection. In *Proc 5th Int Conf Automatic Face Gesture Recogn, Washington, DC, 20–21 May 2002*, pp. 149–154. Washington, DC: IEEE.
25. Schapire RE, Singer Y. 1999. Improved boosting algorithms using confidence-rated predictions. *Machine Learning* **37**(3):297–299.
26. Friedman J, Hastie T, Tibshirani R. 2000. Additive logistic regression: a statistical view of boosting. *Ann Stat* **28**(2):337–374.
27. Viola P, Jones M. 2001. Robust real-time object detection. *Int J Comput Vision*. To appear.
28. Friedman JH. 2001. Greedy function approximation: a gradient boosting machine. *Ann Stat*, **29**(5):1189–1232.
29. Mason L, Baxter J, Bartlett PL, Frean M. 1999. Functional gradient techniques for combining hypotheses. In *Advances in large margin classifiers*, pp. 221–247. Ed. AJ Smola, PL Bartlett, B Schölkopf, D Schuurmans. Cambridge: MIT Press.
30. Zemel RS, Pitassi T. 2001. A gradient-based boosting algorithm for regression problems. In *Advances in neural information processing systems*, Vol. 13. Ed. TK Leen, TG Dietterich, V Tresp. Cambridge: MIT Press. http://citeseer.ist.psu.edu/zemel01gradientbased.html.
31. Schapire RE, Freund Y, Bartlett P, Lee WS. 1998. Boosting the margin: a new explanation for the effectiveness of voting methods. *Ann Stat* **26**(5):1651–1686.
32. Freund Y, Schapire RE. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *J Comput Syst Sci* **55**(1):119–139.
33. Martinez AM, Benavente R. 1998. *The AR face database*. Computer Vision Center Technical Report No. 24. Barcelona, Spain.

5

DETECTION OF THE BREAST CONTOUR IN MAMMOGRAMS BY USING ACTIVE CONTOUR MODELS

Ricardo J. Ferrari

University of Calgary, Canada
University of Sáo Paulo, Brazil

Rangaraj M. Rangayyan

University of Calgary, Canada

J.E. Leo Desautels

Screen Test, Alberta Program for the Early Detection of Breast Cancer, Canada

Annie F. Frère

University of Mogi das Cruzes, Brazil
University of Sáo Paulo, Brazil

Rejane A. Borges

University of Mogi das Cruzes, Sáo Paulo, Brazil

Address all correspondence to: Ricardo J. Ferrari, 2 Forest Laneway, Suite 1901, Toronto, Ontario, M2N 5X9, Canada. Phones: (416) 987-7528 (home). Ferrari@mail.cs.ualberta.ca. Reproduced (with modifications) with permission from RJ Ferrari, RM Rangayyan, JEL Desautels, AF Frère. 2004. Identification of the breast boundary in mammograms using active contour models. *Med Biol Eng Comput* **42**(2):201–208.

We present a method for identification of the breast boundary in mammograms that is intended to be used in the preprocessing stage of a system for computer-aided diagnosis (CAD) of breast cancer and also in the reduction of image file size in Picture Archiving and Communication System (PACS) applications. The method starts by modifying the contrast of the original image. A binarization procedure is then applied to the image, and the chain-code algorithm is used to find an approximate breast contour. Finally, identification of the true breast boundary is performed by using the approximate contour as the input to an active contour model algorithm specially tailored for this purpose. After demarcating the breast boundary, all artifacts outside the breast region are eliminated. The method was applied to 84 medio-lateral oblique mammograms from the Mini-MIAS (Mammographic Image Analysis Society, London, UK) database. Evaluation of the breast boundary detected was performed based upon the percentage of false-positive (FP) and false-negative (FN) pixels determined by a quantitative comparison between the contours identified by a radiologist and by the proposed method. The average FP and FN rates are 0.41 and 0.58%, respectively. According to two radiologists who evaluated the results, the segmentation results were considered acceptable for CAD purposes.

1. INTRODUCTION

Identification of the breast boundary is important in order to demarcate the breast region. Inclusion of this preliminary procedure in computer-aided diagnosis (CAD) systems can avoid useless processing time and data storage. By identifying the boundary of the breast, it is possible to remove any artifact present off the breast, such as patient markings (often high-intensity regions) and noise, which can affect the performance of image analysis and pattern recognition techniques. In addition to the use as a preprocessing step in breast CAD systems, identification and extraction of the effective breast region is also important in picture archiving and communication systems (PACS) and telemammography systems [1].

The profile of the breast has been used as additional information in different tasks in mammography. Bick et al. [2] and Byng et al. [3], for example, used the skin–air boundary information to perform density correction of peripheral breast tissue on digital mammograms, which is affected by the compression procedure applied during imaging. Chandrasekhar and Attikiouzel [4] discussed the importance of the skin–air boundary profile as a constraint in searching for the nipple location, which is often used as a reference point for registering mammograms of the same subject. Other groups have used the breast boundary to perform registration between left and right mammograms in the process of detection of asymmetry [5, 6].

Most of the works presented in the literature to identify the boundary of the breast are based upon histogram analysis [1–3, 5–7], which may be critically dependent upon the threshold selection process and the noise present in the image. Such techniques, as discussed by Bick et al. [8], may not be sufficiently robust for a screening application. The major problem with techniques using global threshold is the high variability of the background region of mammogram images. Promising results were presented by Masek et al. by using a local threshold technique [9].

In the present work, an active contour model, especially designed to be locally adaptive, is used for identification of the breast boundary. An optimally computed threshold value is used only to find an initial boundary for the active contour model algorithm.

The paper is organized as follows. Section 2 provides a brief description of our initial technique (Method 1) based upon the traditional active contour model for detection of the breast boundary. An improved technique (Method 2) for identification of the breast boundary is described in Section 3. Section 3.2 presents the characteristics of the database used in this work. The protocol used for evaluation of the results is presented in Section 3.3, followed by the results and discussion in Section 4. Conclusions are presented in Section 5.

2. METHOD 1: IDENTIFICATION OF THE BREAST BOUNDARY USING A TRADITIONAL ACTIVE DEFORMABLE CONTOUR MODEL

In the initial stage of our investigation [10], we used the traditional active deformable contour model (or snake, [11]) for detection of the breast boundary. The method, summarized in Figure 1, is composed of six main stages [10]:

Stage 1: The image contrast is enhanced by using a simple logarithmic operation [12]. A contrast-correction step using a simple logarithmic operation as

$$G(x, y) = \log[1 + I(x, y)] \tag{1}$$

is applied to the original image $I(x, y)$; $G(x, y)$ is the transformed image. This dynamic range compression operation, although applied to the whole image, enhances significantly the contrast of the regions near the breast boundary in mammograms, which are characterized by low density and poor definition of details [2, 3]. The rationale behind the application of this procedure to the image is to determine an approximate breast contour as close as possible to the true breast boundary. The effect of this procedure can be seen by comparing the original and enhanced images in Figures 2(a) and 2(b).

Stage 2: A binarization procedure using the Lloyd-Max algorithm is applied to the image [14]. The Lloyd-Max least-squares algorithm is an iterative and fast technique (convergence was reached in an average of three or four cycles in the present work) for the design of a quantizer with low distortion. It uses the intensity distribution (histogram) of the image to optimize, in the sense of a mean-squared-error criterion, the quantization procedure applied to the image, checking each possible N-level quantizer ($N = 2$ for binarization purposes) to determine the quantizer that provides the lowest distortion. The distortion measure is given by

$$\varepsilon = \sum_{j=1}^{N} \sum_{x=a_j}^{b_j} (x - y)^2 f(x), \tag{2}$$

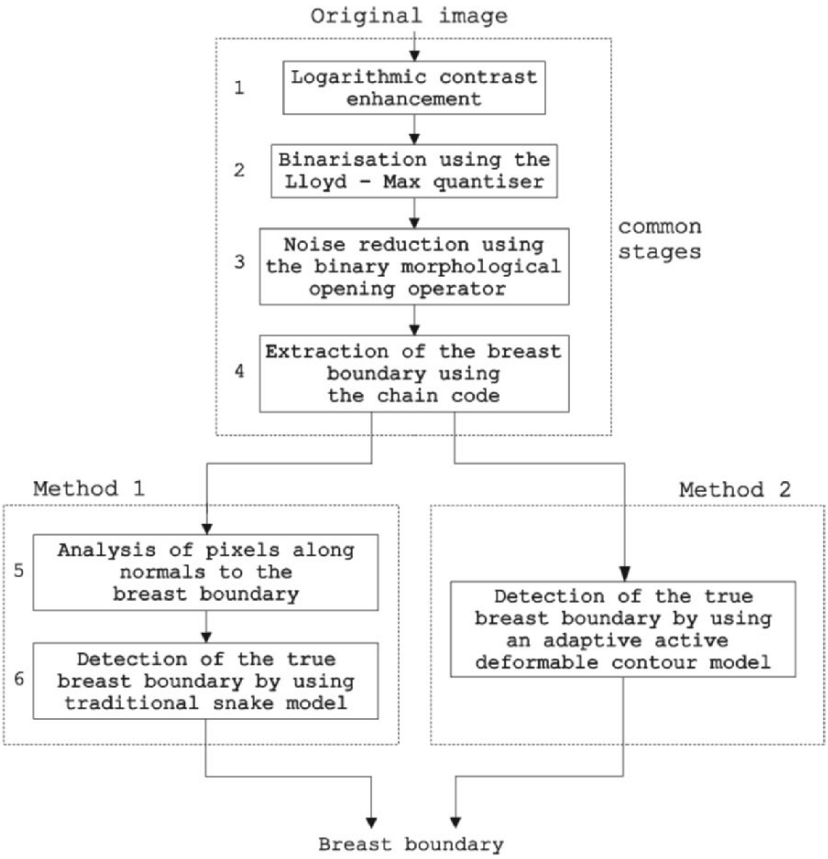
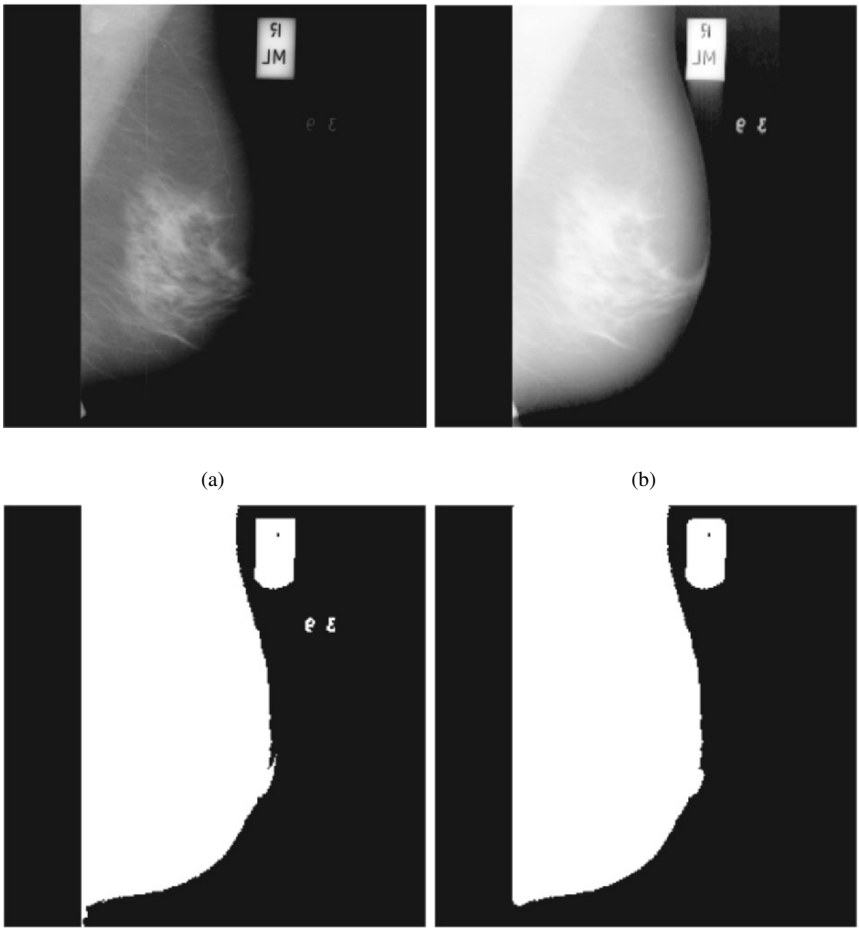


Figure 1. Flowchart of the procedures for identification of the skin–air boundary of the breast.

where $y = \frac{1}{2}(L_1 + L_2)$ is the threshold value for the $L_1 < L_2$ quantization levels, $f(x)$ is the probability density function (pdf) represented by the gray-level distribution or histogram of the image, and x stands for the gray level. The quantities a_j and b_j are, respectively, the minimum and maximum gray-level values for the quantization level L_j. The quantization levels are computed as the centroids of the quantization intervals as

$$L_j = \frac{\sum_{x=a_j}^{b_j} x f(x)}{\sum_{x=a_j}^{b_j} f(x)}, \; j = 1, 2. \tag{3}$$

The iterative procedure starts with L_1 and L_2 equal to the minimum and maximum gray-level values in the histogram, respectively, and stops when there is no change

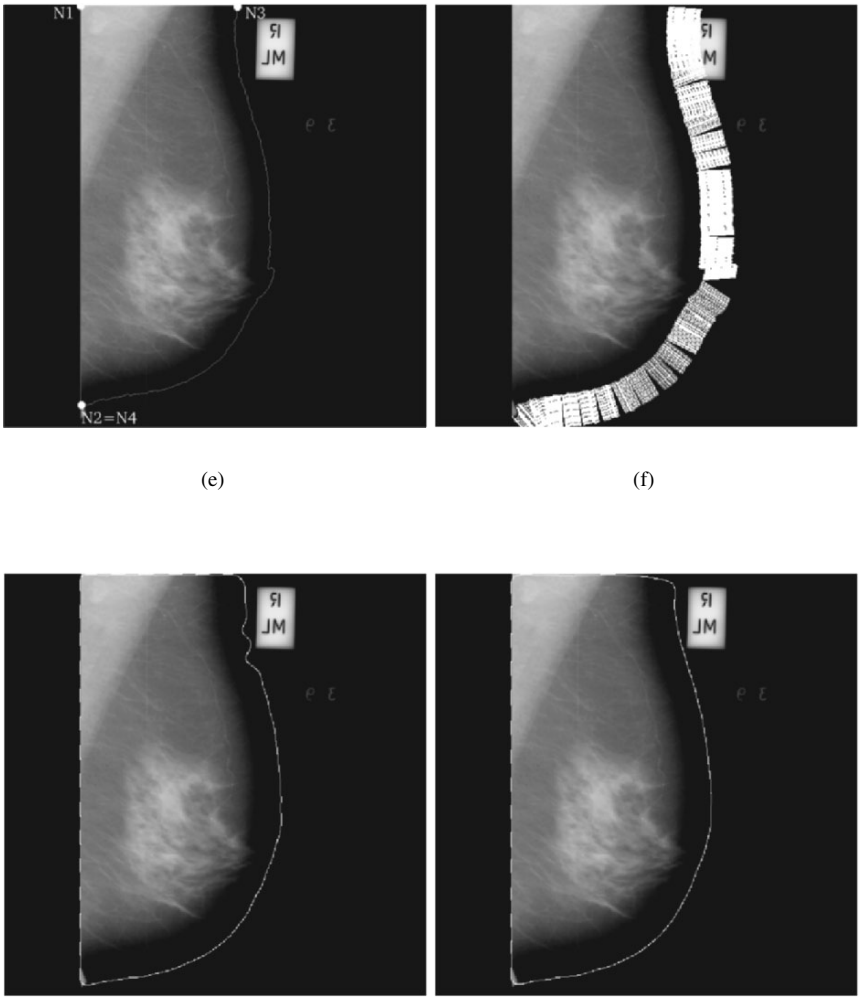


(a) (b)

(c) (d)

in the threshold value between two consecutive iterations. Algorithm 1 provides a description of the steps in the implementation of the binarization procedure.

Stage 3: Spurious details generated by the binarization step are removed by using a morphological opening operator [12] with a circular structuring element with a diameter of 7 pixels. Figures 2(c)–(d) show the result of the binarization procedure for the mammogram in Figure 2(a) before and after application of the morphological opening operator.



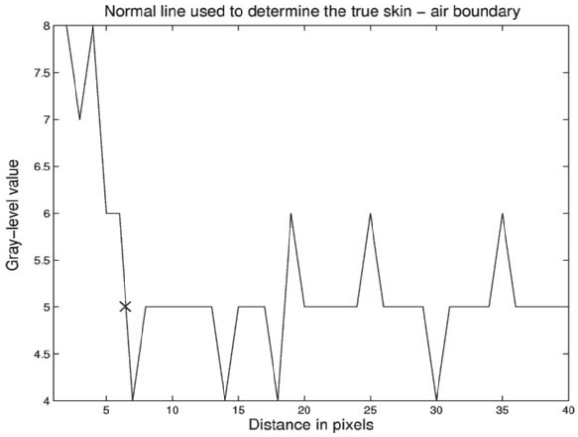
(e) (f)

(g) (h)

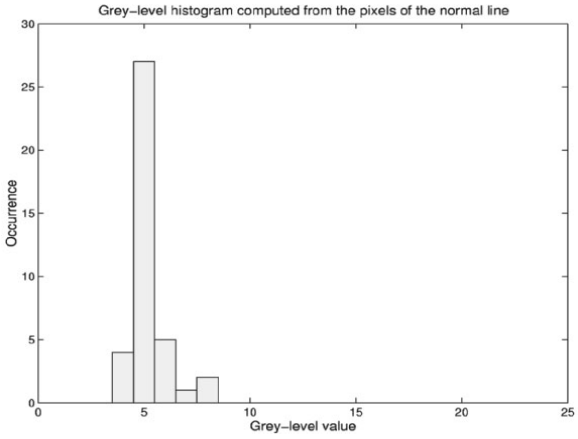
Figure 2. Results of each stage of Method 1 for identification of the breast boundary. (a) Original image mdb042 from the Mini-MIAS database [13]. (b) Image after the logarithmic operation. (c)–(d) Binary image before and after applying the binary morphological opening operator. (e) Control points N1 to N4 (automatically determined) used to limit the breast boundary. (f) Normal lines computed from each pixel in the skin–air boundary. (g) Boundary resulting after histogram-based analysis of the normal lines. (h) Final boundary.

Algorithm 1 : Algorithm for the Lloyd-Max method [14] used for binarization of mammograms.

```
//
// Compute probability density function (pdf) as the
// normalized image-intensity histogram and determine the
// Min and Max gray level values of the image
//
Pdf = image->computePdf ();
Min = image->getMinGrayLevel ();
Max = image->getMaxGrayLevel ();

//
// Initialize quantization levels with Min and Max values
//
L[0] = Min;
L[1] = Max;

//
// Variables used for the distortion measure (MSE)
//
NewDistortion = 0;

do {
    OldDistortion = NewDistortion;

    //
    // Compute the threshold value y
    //
    y = round ( (L[0] + L[1]) / 2 );

    NewDistortion = 0;
    for (x=L[0]; x < L[1]; x++) {
       Newdistortion += sqr(x - y) * Pdf[x - Min];
    }

    //
    // Two cycles loop for binarization
    //
    for (i=0; i < 2; i++) {
       //
       // Compute new quantization levels
       //
       sum1 = sum2 = 0;
       for (x=L[i]; x < y; x++) {
          sum1 += x * Pdf[x - Min];
          sum2 += Pdf[x - Min];
       }
       L[i] = round (sum1 / sum2);
    }
} while ( (NewDistortion - OldDistortion > EPSILON) );
```



(a)



(b)

Figure 3. (a) Profile of a sample normal line used to determine an approximate skin–air boundary. The symbol "×" indicates the skin–air intersection determined in Stage 5. (b) Histogram computed from (a).

Stage 4: After the binarization procedure, an approximate contour C_{appr} of the breast is extracted by using the chain-code method [12]. The starting point of C_{appr} is obtained by following the horizontal path that starts at the center of gravity of the binary image and goes toward the chest wall until the left-hand edge of the image is found. (Images in the MIAS database [13] have a blank region on the left-hand side of the mammograms.) This procedure avoids selecting an initial boundary from artifacts or patient labels that may be present in the image. A set of four control points [see Figure 2(e)] are automatically determined and used to limit the breast boundary. The points are defined as N1: the top-left corner pixel of the boundary loop; N2: the lowest pixel on the left edge of the boundary loop; N3: the farthest point on the boundary from N2 (in terms of the Euclidean distance through the breast); N4: the farthest point on the skin–air boundary loop from N1.

Stage 5: Normal lines of length 40 pixels (at a sampling resolution of 200 μm, length = 0.8 cm) are computed at each point of the approximate skin–air boundary in the original image [see Figure 2(f)]. The gray-level histogram of the pixels along each normal line is computed, and the skin–air intersection is defined as the first pixel, while traversing along the normal line from inside the breast toward the outside, that has the gray level equal to the mode of the histogram, as illustrated in Figure 3. This procedure was designed in order to provide a close estimate to the true skin–air boundary [see Figure 2(g)], and thereby reduce the chances of the active contour (used in the next stage) converging to a wrong contour.

Stage 6: The result of Stage 5 was often observed to be a noisy and rough contour due to the local nature of the corrections applied to the contour. In order to obtain a smoother result and to permit application of global constraints, the traditional parametric active contour or snake model [11] is applied to the result of Stage 5. This contour is moved through the spatial domain of the image in order to minimize the energy functional:

$$E = \int_0^1 \left\{ \frac{1}{2} \left[\alpha \left| v'(s) \right|^2 + \beta \left| v''(s) \right|^2 \right) + E_{\text{ext}}[v(s)] \right] \right\} ds, \tag{4}$$

where α and β are weighting parameters that control, respectively, the tension and rigidity of the snake; $v'(s)$ and $v''(s)$ values denote the first and second derivatives of $v(s)$ with respect to s, where $v(s)$ indicates the continuous representation of the contour. The external energy function $E_{\text{ext}}[v(s)]$ is derived from the image $I(x, y)$, and is defined in this work as

$$E_{\text{ext}}(x, y) = - \left\| \nabla I(x, y) \right\|^2, \tag{5}$$

where ∇ is the gradient operator. In the present work, the values $\alpha = 0.001$ and $\beta = 0.09$ were experimentally derived based upon the approximate boundary obtained in the previous stage, the quality of the external force derived from the original image, and the final contours obtained.

Figure 2(h) illustrates the result of application of the active contour model to the contour estimate shown in Figure 2(g). The global nature of the active contour model has removed the local irregularities present in the contour near the "R–ML" label in Figure 2(g).

Sixty-six images from the Mammographic Image Analysis Society (Mini-MIAS, [13]) database were used to assess the performance of the method. The results were subjectively analyzed by an expert radiologist (JELD). According to the opinion of the radiologist, the method detected accurately the breast boundary in 50 images, and reasonably well in 11 images. In five images the method failed completely (the result was considered not acceptable for CAD purposes) because of distortions and artifacts present near the breast boundary (see Figure 4). Limitations of the method exist mainly in Stages 5 and 6. Although Stage 5 helps in obtaining a good breast contour, it is time consuming and may impose a limitation in practical applications. The traditional snake model used in Stage 6 is not robust (the method is not locally adaptive) in the presence of noise and artifacts, and has a short range of edge capture.

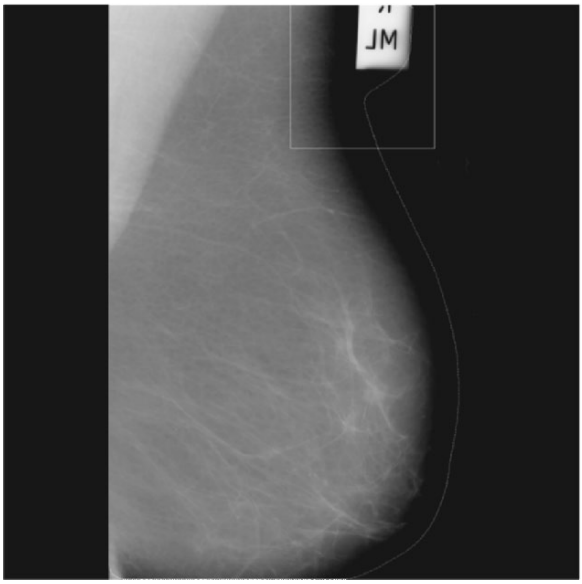
In the improved method (Method 2) described in the following section, we replace the traditional snake algorithm with an adaptive active deformable contour model (AADCM) specially designed for the present application. The algorithm includes a balloon force in an energy formulation that minimizes the influence of the initial contour on the convergence of the algorithm. In this energy formulation, the external energy is also designed to be locally adaptive. In formulating the AADCM, we removed Stage 5 of Method 1; see Figure 1. A pseudo-code of the snake algorithm used in the Method 2 is presented in Algorithm 2.

3. METHOD 2: IDENTIFICATION OF THE BREAST BOUNDARY USING AN ADAPTIVE ACTIVE DEFORMABLE CONTOUR MODEL (AADCM)

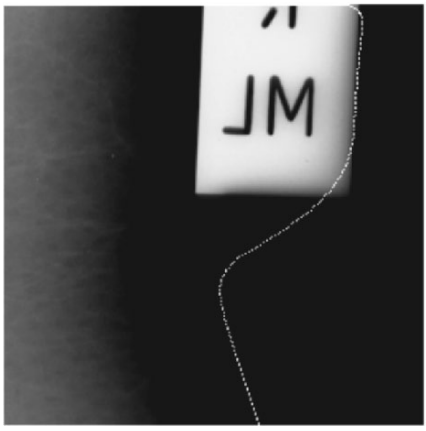
The improved method to identify the breast boundary is summarized in the flowchart in Figure 1. Stages 1–4 of the initial method described in the preceding section are used to find an approximate breast boundary. The sampled contour, $V = v_1, v_2, \cdots, v_N$, with an ordered collection of N points $v_i = (x_i, y_i)$, $i = 1, 2, \cdots, N$, is obtained from Stage 4 of Method 1 by sampling the approximate contour C_{appr} [see Figure 5(a)], and used as the initial contour in the AADCM; see Figure 5(b). Only 10% of the total number of points present in C_{appr}, obtained by equidistant downsampling of the approximate contour, are used in the sampled contour.

In our implementation of the AADCM, which combines several characteristics from other known active contour models [15–17], the contour is moved through the spatial domain of the image in order to minimize the following functional of energy:

$$E_{\text{total}} = \sum_{i=1}^{N} \left[\alpha E_{\text{internal}}(v_i) + \beta E_{\text{external}}(v_i)\right], \tag{6}$$



(a)



(b)

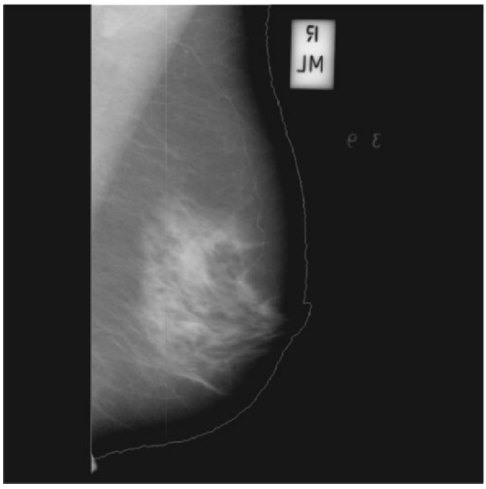
Figure 4. Result of the segmentation algorithm (Method 1) showing wrong convergence of the breast contour into a region of high gradient value. (a) The breast boundary detected automatically, superimposed on the original image (mdb006) from the Mini-MIAS database. (b) Details of the breast contour attracted to the image identification marker (corresponding to the boxed region in the original image).

Algorithm 2 : Pseudo-code of the active contour used in the Method 2.

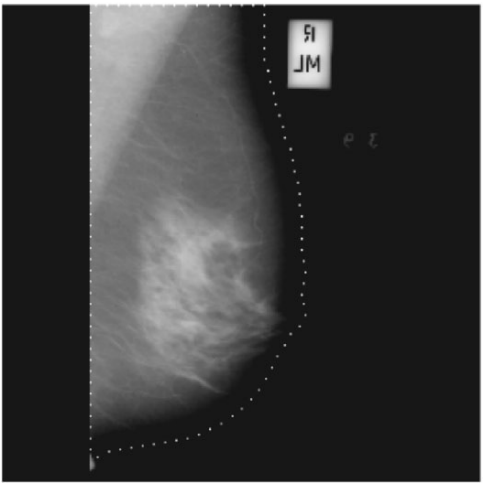
```
// Compute the average distance among the snake nodes
// used for the normalization of the continuity energy
averageDist = getAvgDistance();

// Compute the center of gravity used for the balloon energy
computeCG();

// Initial values for the energy
Energy_old = 0;
Energy_new = 0;

// Create a Snake object from the approximated contour
// represented by a linked list of type Point {x, y}
MySnake = Snake( ApproximatedContour );

// Loop for stages: at each stage the blur
// applied to the original image is reduced
do {
  // Counter variable for the number of adjusted snake nodes
  movedNodes = 0;

  // Count number of complete cycle through the snake nodes
  countIter = 0;

  // Initial values for the energy
  Energy_old = Energy_new;
  Energy_new = 0;

  // Create two auxiliar snake nodes for handle the
  // the last end node of the snake
  SnakeNode *nextNode = NULL, *prevNode = NULL;

  // Set position of the snakes to the initial position
  SnakeNode *start = MySnake.current;

  // For all snake nodes
  while (MySnake.current)
  {
    // Perform some special handling for prevNode and nextNode
    if (MySnake.current == start) {
      prevNode = (SnakeNode *) end;
      nextNode = (SnakeNode *) MySnake.current->next;

      if (++countIter > 1)
        break;

    } else {
      if (MySnake.current == end) {
        prevNode = (SnakeNode *) MySnake.current->prev;
        nextNode = (SnakeNode *) start;
      } else {
        prevNode = (SnakeNode *) MySnake.current->prev;
        nextNode = (SnakeNode *) MySnake.current->next;
      }
    }
```

```
// Get coordinates and energy of the coefficients for
// the current snake node
curX  = MySnake.current->data->GetXCoord();
curY  = MySnake.current->data->GetYCoord();
alpha = MySnake.current->data->GetAlpha ();
beta  = MySnake.current->data->GetBeta  ();
gamma = MySnake.current->data->GetGamma ();

// Compute the energy terms for all possible positions
// in the neighborhood
for (row = 0; row < NEIGHBOR_HEIGHT; row++)
  for (col = 0; col < NEIGHBOR_WIDTH; col++) {
    //
    // Compute indexes of the neighborhood
    //
    nIndex = row * NEIGHBOR_WIDTH + col;
    nX     = curX + (col - (NEIGHBOR_WIDTH  - 1) / 2);
    nY     = curY + (row - (NEIGHBOR_HEIGHT - 1) / 2);

    // Avoid neighborhood limits being out of range
    CheckAndCorrectLimits (nX, nY);

    // Get image gradient value in the position (nX, nY)
    I_grad[nIndex] = image->GetGradient (nX, nY);

    // Compute and store all energies in the position (nX, nY)
    // See equations 1.8, 1.9, and 1.10
    e_cont[nIndex] = Continuity (prevNode, nX, nY, nextNode);
    e_grad[nIndex] = Gradient   (prevNode, nX, nY, nextNode);
    e_ball[nIndex] = Balloon    (prevNode, nX, nY, nextNode);
  }

  // Normalize the energy to be in the range [0, 1]
  // See equations 1.11, 1.12, and 1.13
  normContinuity (e_cont, NEIGHBOR_WIDTH * NEIGHBOR_HEIGHT);
  normGradient   (e_grad, NEIGHBOR_WIDTH * NEIGHBOR HEIGHT);
  normBalloon    (e_ball, I_grad, NEIGHBOR_WIDTH * NEIGHBOR HEIGHT);

  // Start with an insanely high upper bound
  minEnergy = MAX_ENERGY;

  // Now find the minimum energy location in the neighborhood
  for (int row = 0; row < NEIGHBOR_HEIGHT; row++)
    for (int col = 0; col < NEIGHBOR_WIDTH; col++) {
      //
      // Compute index and position in the neighborhood
      //
      nIndex = row * NEIGHBOR_WIDTH + col;
      nX     = curX + (col - (NEIGHBOR_WIDTH  - 1) / 2);
      nY     = curY + (row - (NEIGHBOR_HEIGHT - 1) / 2);

      // Check neighborhood limits
      CheckAndCorrectLimits (nX, nY);
```

```
          // Compute energy value of a node in the (nX, nY) position
          energy = alpha * e_cont[nIndex] + beta  * e_grad[nIndex];
                 + gamma * e_ball[nIndex];

          // Save position and energy value of the
          // point with minimum energy in the neighborhood
          if (energy < minEnergy) {
            minEnergy = energy;
            EminX = nX;
            EminY = nY;
          }
        }

      // Move current snake node to a position of minimum energy,
      // if one is found in the neighborhood
      if ( (curX != EminX) || (curY != EminY) ) {
        MySnake.current->data->SetPoint (EminX, EminY);
        movedNodes++;
        Energy_new += minEnergy;
      }

      // Move to the next node of the snake
      MySnake.moveToNext();
    }

    // Update average distance among the snake nodes
    // and center of gravity of the snake
    averageDist = getAvgDistance ();
    computeCG ();

    //  Selectively relaxes the curvature term for particular
    //  snake node if the node satisfies the following conditions:
    //  -node must have higher curvature than its neighboring nodes
    //  -node must have a curvature above the threshold value 0.25
    AllowCorners ();

    // Add or remove nodes based on the average and
    // curvature values of snake nodes
    Resampling ();

    // Reduce blur applied to the image if number of moved nodes
    // is less than 10% the number of the total nodes in the snake
    if ( (movedNodes < 0.1 * MySnake.getTotalNodes()) ||
         (Energy_old < Energy_new) && (stageCount  <= no_stages) )
    {
      stageCount++;
      if (stageCount <= no_stages)
      {
        image->ReduceBlur ();
        Energy_new = MAX_ENERGY;
      }
    }

} while (stageCount <= no_stages);
```



(a)



(b)

Figure 5. (a) Approximate breast contour obtained from Stage 4 of Method 1, as described in Section 2, for image mdb042. (b) Sampled breast contour used as the input to the AADCM.

where α and β are weighting parameters that control the internal and external energies, E_{internal} and E_{external}, respectively, at each point v_i.

The internal energy is composed of two terms as

$$E_{\text{internal}}(v_i) = aE_{\text{continuity}}(v_i) + bE_{\text{balloon}}(v_i). \tag{7}$$

This energy component ensures a stable shape for the contour and tries to keep constant the distance between the points in the contour. The weighting parameters a and b were initially set to unity ($a = b = 1$) in this work, since the initial contours present smooth shapes and are close to the true boundary in most cases.

For each element (j, k) in a neighborhood of 7×7 pixels of v_i, element $e_{cjk}(v_i)$ of $E_{\text{continuity}}$ is computed as

$$e_{cjk}(v_i) = \frac{1}{l(v_i)} \left| p_{jk}(v_i) - \rho(v_{i-1} + v_{i+1}) \right|^2, \tag{8}$$

where $l(v_i) = \frac{1}{N} \sum_{i=1}^{N} |v_{i+1} - v_i|^2$ is the normalization factor that makes the continuity energy independent of the size, location, and orientation of V. $p_{jk}(v_i)$ is the point in the image at position (j, k) in the 7×7 neighborhood of v_i, and $\rho = \frac{1}{2\cos(\frac{2\pi}{N})}$ is a constant factor to keep the location of the minimum energy lying on the circle connecting v_{i-1} and v_{i+1}; the centroid of the contour forms the reference point or origin for this factor, as illustrated in Figure 6. It should be noted that, in the present work, the methods described are applied to closed contours.

The balloon force is used to force the expansion of the initial contour toward the breast boundary. In this work, the balloon force was made adaptive to the magnitude of the image gradient, causing the contour to expand faster in homogeneous regions and slower near the breast boundary. The balloon energy term, $e_{bjk}(v_i)$, is defined as

$$e_{bjk}(v_i) = n_i \cdot \{v_i - p_{jk}(v_i)\}, \tag{9}$$

where n_i is the outward unit normal vector of V at point v_i, and the symbol $\cdot$ indicates the dot product. n_i is computed by rotating vector $t_i = \frac{v_i - v_{i-1}}{\|v_i - v_{i-1}\|} + \frac{v_{i+1} - v_i}{\|v_{i+1} - v_i\|}$, which is the tangent vector at the point v_i, by 90°.

The external energy is based upon the magnitude and direction of the image gradient and is intended to attract the contour to the breast boundary. It is defined as

$$e_{ejk}(v_i) = -n_i \cdot \nabla I\{p_{jk}(v_i)\}, \tag{10}$$

where $\nabla I\{p_{jk}(v_i)\}$ is the image gradient vector at (j, k) in the 7×7 neighborhood of v_i. The direction of the image gradient is used to avoid attraction of the contour by edges that may be located near the true breast boundary, such as identification marks and small artifacts; see Figure 7(a) and (b). In this situation, the gradient direction at position (j, k) on an edge near the breast boundary and the direction of

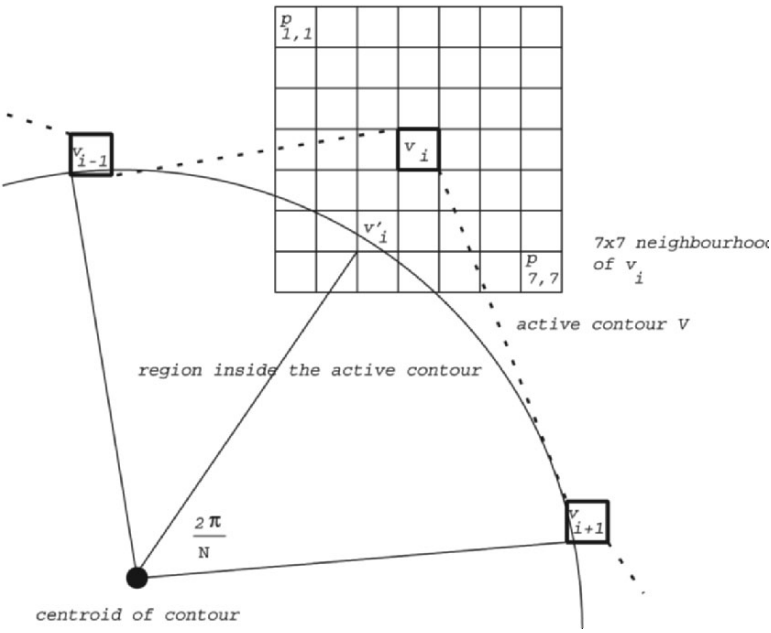


Figure 6. Characteristics of the continuity energy component in the adaptive active deformable contour model. The contour pixel, v_i, is moved to position v_i' by application of the continuity requirement. Adapted with permission from B Mackiewich, JEL Desautels, RA Borges, AF Frère. 2004. Intracranial boundary detection and radio frequency correction in magnetic resonance images. *Med Biol Eng Comput* **42**(2):201–208. Copyright ©2004, The Institute of Engineering and Technology.

the unit normal of the contour will have opposite signs, which makes the functional of energy present a large value at (j, k).

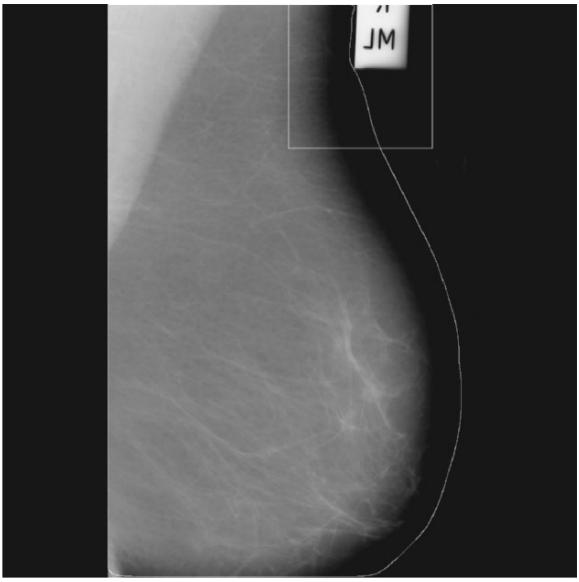
3.1. Minimization of the Energy Functionals

In order to allow comparison between the various energy components described above, each energy parameter is scaled to the range [0,1] according to the following equations:

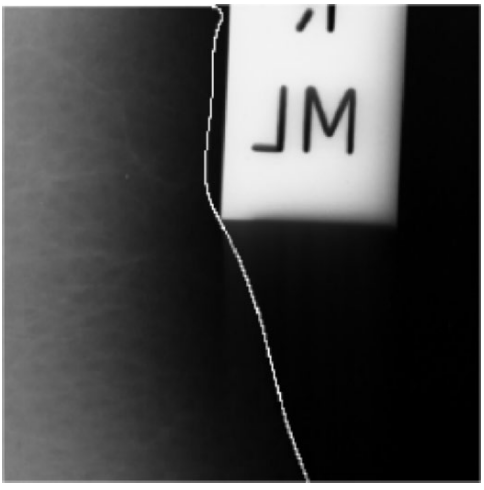
$$E_{\text{continuity}}(v_i) = \frac{e_{cjk}(v_i) - e_{c\,\min}(v_i)}{e_{c\,\max}(v_i) - e_{c\,\min}(v_i)}\,; \tag{11}$$

$$E_{\text{balloon}}(v_i) = \frac{e_{bjk}(v_i) - e_{b\,\min}(v_i)}{e_{b\,\max}(v_i) - e_{b\,\min}(v_i)} \cdot \left(1 - \frac{\|\nabla I(v_i)\|}{\|\nabla I\|_{\max}}\right); \tag{12}$$

$$E_{\text{external}}(v_i) = \frac{e_{ejk}(v_i) - e_{e\,\min}(v_i)}{\max[e_{e\,\max}(v_i) - e_{e\,\min}(v_i),\ \|\nabla I\|_{\max}]}. \tag{13}$$



(a)



(b)

Figure 7. Application of the gradient direction information for avoiding attraction of the boundary to objects near the true boundary. (a) Breast boundary detected automatically, superimposed on the original image (mdb006) from the Mini-MIAS database. (b) Details of the detected breast boundary close to the image identification marker (corresponding to the boxed region in the original image).

Here, $e_{\min}$ and $e_{\max}$ indicate the minimum and maximum of the corresponding energy component in the 7×7 neighborhood of v_i. $\|\nabla I\|_{\max}$ is the maximum gradient magnitude in the entire image. The minimization procedure is performed based on the energy components computed in a local region defined by the 7×7 neighborhoods of three adjacent contour pixels. At a given step, only one contour pixel v_i is modified.

In the present work, the Greedy algorithm proposed by Williams and Shah [17] was used to perform minimization of the functional of energy in Eq. (6). Although this algorithm has the drawback of not guaranteeing a global-minimum solution, it is faster than the other methods proposed in the literature — such as dynamic programming, variational calculus, and finite elements. It also allows insertion of hard constraints, such as curvature evaluation, as discussed below.

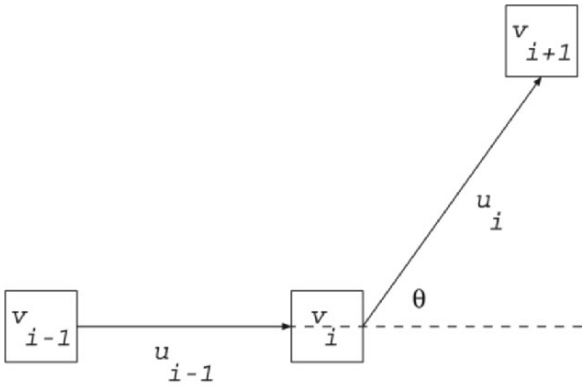
Convergence of the AADCM is achieved in two stages by smoothing the original image with two different Gaussian kernels defined with $\sigma_x = \sigma_y = 3$ pixels and $\sigma_x = \sigma_y = 1.5$ pixels. The first stage of smoothing introduces a large amount of blur in the image, and helps the contour to expand faster since low-contrast and small details will have been removed. In the second stage, the amount of blurring introduced is lesser, and the active contour is more sensitive to small variations; note that the second-stage filter is applied to the original image. At each stage, the iterative process is stopped when the total energy of the contour increases between consecutive iterations. This coarse-to-fine representation is intended to give more stability to the contour.

In order to allow the deformable contour to adjust to corner regions, such as the upper-right limit of the breast boundary, a constraint was inserted at the end of each iteration to relax the continuity term defined in Eq. (8). The curvature value $C(v_i)$ at each point v_i of the contour was computed as

$$C(v_i) = 2\sin(\theta/2) = \left\| \frac{u_i}{\|u_i\|} - \frac{u_{i-1}}{\|u_{i-1}\|} \right\|^2, \tag{14}$$

where $u_i = (v_{i+1} - v_i)$ is the vector joining two neighboring contour elements and θ is the external angle between two such vectors sharing a common contour element. Figure 8 illustrates the vectors and the external angle involved in the calculation of curvature as in Eq. (14). This curvature equation, as discussed by Williams and Shah [17], has three important advantages over other curvature measures: it requires only simple computation, gives coherent values, and depends solely on relative direction.

At each v_i, the weight values for the continuity term and the external energy are set, respectively, to zero ($a = 0$) and to twice the initial value ($\beta = 2\beta$) if $[C(v_i) > C(v_{i-1})]$ and $[C(v_i) > C(v_{i+1})]$ and $[C(v_i) > Thresh]$. The threshold value, $Thresh$, was set equal to 0.25 in the present work, which corresponds to an external angle of approximately 29°. According to Williams and Shah [17], this value of the threshold has been proven experimentally to be sufficiently large



(a)

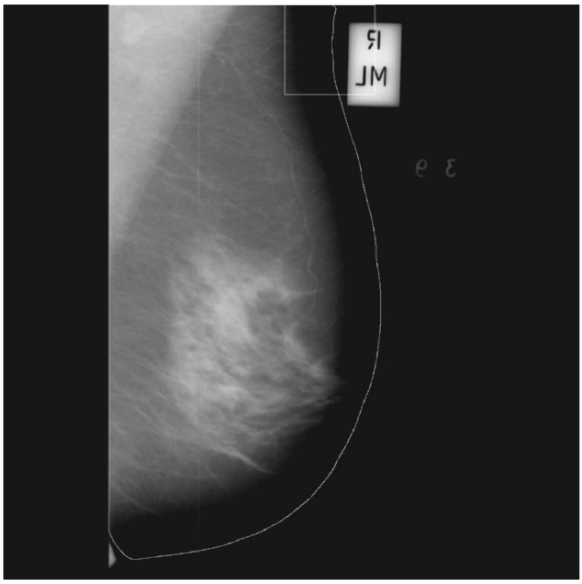
Figure 8. Illustration of the vectors and external angle used in Eq. (14) for calculation of curvature.

to differentiate between corners and curved lines. Figures 9(b) and (c) illustrate an example with and without the use of the curvature constraint to correct corner effects.

The weighting parameters, α and β, in Eq. (6) were initialized to 0.2 and 1.0, respectively, for each contour element. This set of weights was selected experimentally by using a group of 20 images randomly selected from the Mini-MIAS database [13], not including any image in the test set used in the present work to evaluate the results. A larger weight was given to the gradient energy to favor contour deformation toward the breast boundary rather than smoothing due to the internal force. Although these parameters were derived based upon experiments, they have proven to be robust when applied to other images from the Mini-MIAS database.

3.2. Database

Eighty-four images, randomly chosen from the Mini-MIAS database [13], were used in this work. All images are medio-lateral oblique (MLO) views with a 200-μm sampling interval and 8-bit gray-level quantization. For reduction of processing time, all images were downsampled with a fixed sampling distance so that the original images corresponding to a matrix size of 1024×1024 pixels were transformed to 256×256 pixels. All results obtained with the downsampled images were mapped to the original mammograms for subsequent analysis and display.



(a)

(b) (c)

Figure 9. Example of the constraint used in the active contour model to correct smoothing effects at corners. (a) Original image; the box indicates the region of concern; (b,c) show the details of the breast contour, with and without the use of the constraint for corner correction, respectively.

3.3. Protocol for Evaluation of the Results

The results obtained by the proposed method were evaluated in consultation with two radiologists experienced in mammography (JELD and RAB). The first radiologist (JELD) assessed the results visually by analyzing the segmented images on a monitor. The second radiologist (RAB) participated in both delineation of the breast contours and assessment of the results.

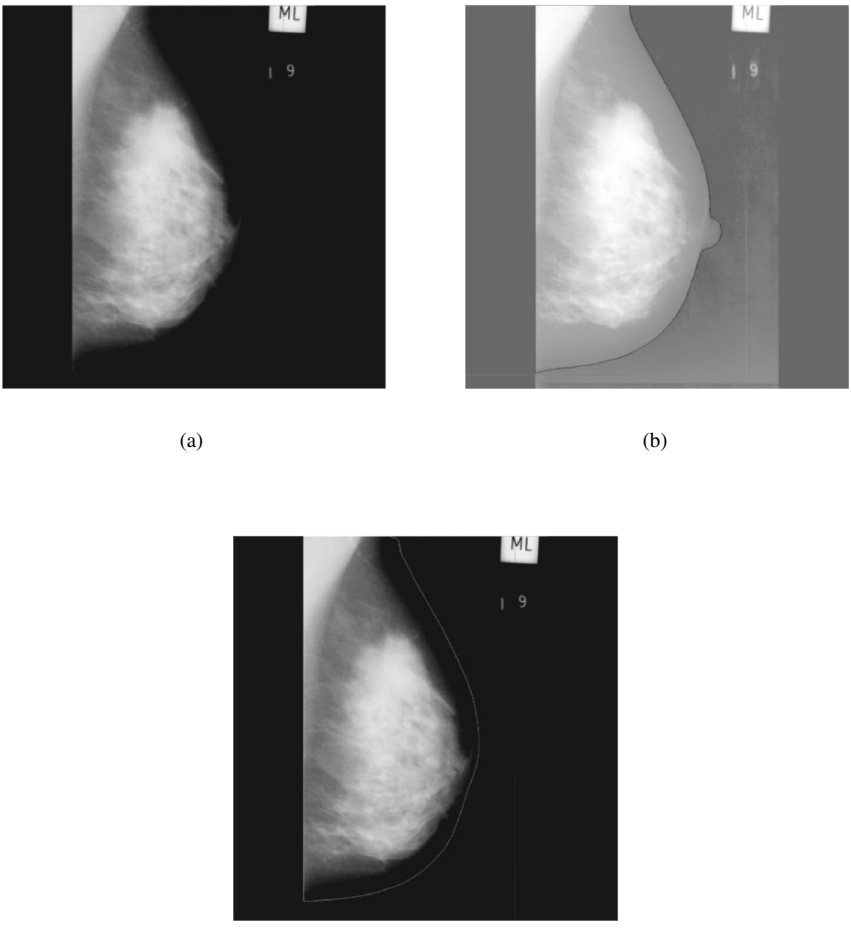
The test images were displayed on a computer monitor (19-in diagonal size, 0.27 mm dot pitch). By using the Gimp program [18], the contrast and brightness of each image were manually enhanced so that the breast contour could be easily visualized. Then, the breast boundary was manually drawn on the enhanced image by one of the authors (RJF) under the supervision of a radiologist (RAB), without referring to the results of detection by the proposed methods. The zoom option of the Gimp program was used to aid in drawing the contours. The results were printed on paper by using a laser printer with 600-dpi resolution. The breast boundaries of all images were visually checked by a radiologist (RAB) using the printed images (hardcopy) along with the displayed images (softcopy); the assessment was recorded for further analysis. (Data files of the manually drawn and automatically detected contours are available from the corresponding author upon request.)

The segmentation results related to the breast contours detected by image processing were objectively evaluated based upon the number of false-positive (FP) and false-negative (FN) pixels identified and normalized with reference to the corresponding areas demarcated by the manually drawn contours. The reference area for the breast boundary was defined as the area of the complete breast image delimited by the left-hand edge of the breast image and the hand-drawn breast boundary. An FP pixel was defined as a pixel outside the reference region that was included in the region segmented. An FN pixel was defined as a pixel in the reference region that was not present within the segmented region.

4. RESULTS AND DISCUSSION

The results obtained from the segmentation procedure were analyzed according to the protocol described in Section 5. A total of 84 images were analyzed; three examples of the results are illustrated in Figures 10, 11, and 12. It is worth noting from the example in Figure 11 that the method performs well even when the contour is interrupted by the image boundary; this is because the control points of the active contour model are kept fixed when they are on the boundary of the image. A few more examples of the segmented results are shown in Figure 13.

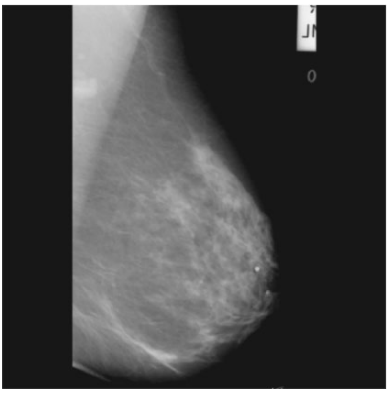
The FP and FN average percentages and the corresponding standard deviation values obtained for the 84 images are $0.41 \pm 0.25\%$ and $0.58 \pm 0.67\%$, respectively. Thirty-three images presented both FP and FN percentages less than 0.5%; 38 images presented FP and FN percentages between 0.5 and 1%; the FP and FN percentages were greater than 1% for 13 images.



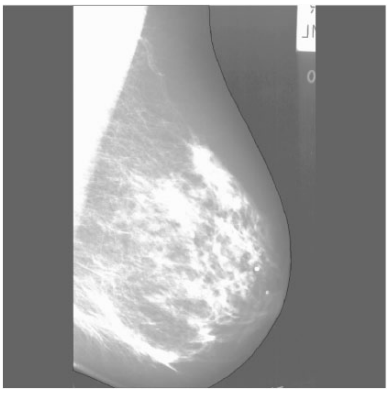
(a) (b)

(c)

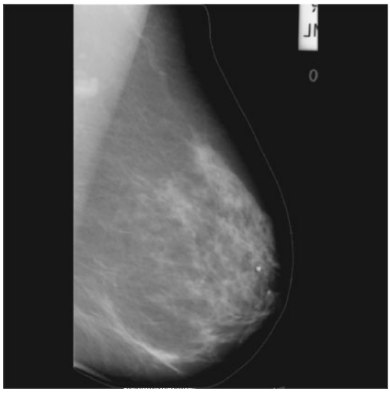
Figure 10. Results obtained for image mdb003. (a) Original image. (b) Hand-drawn boundary, superimposed on the histogram-equalized image. (c) Breast boundary detected automatically, superimposed on the original image.



(a)



(b)



(c)

Figure 11. Results obtained for image mdb008. (a) Original image. (b) Hand-drawn boundary, superimposed on the histogram-equalized image. (c) Breast boundary detected automatically, superimposed on the original image.

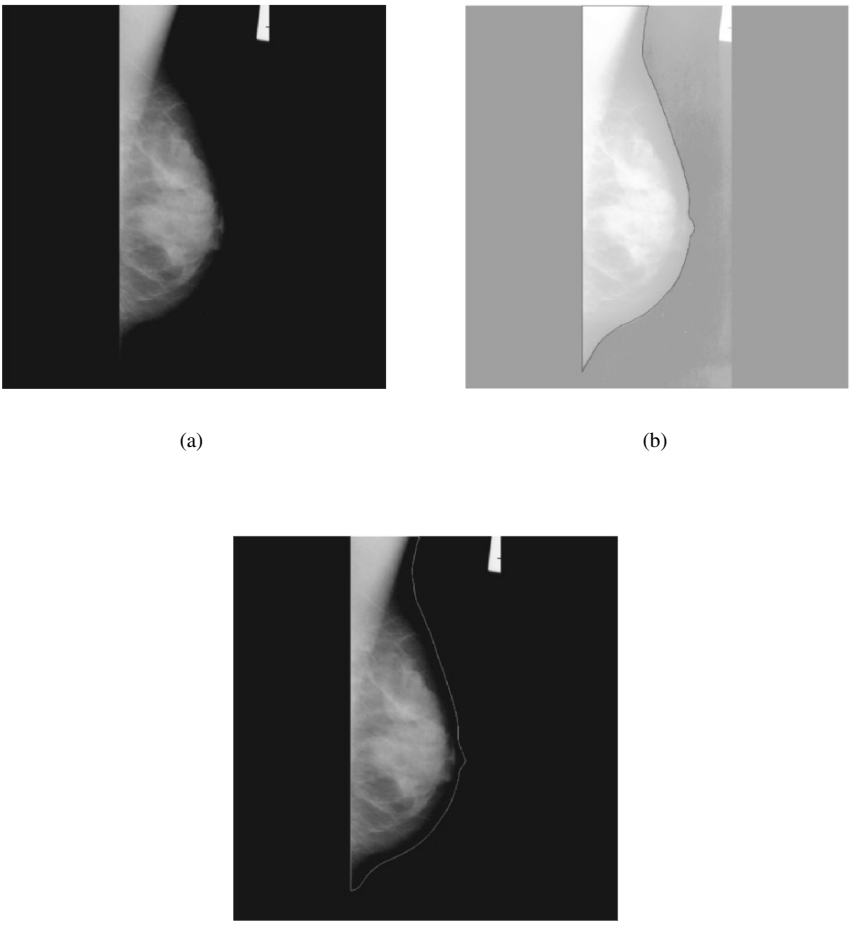
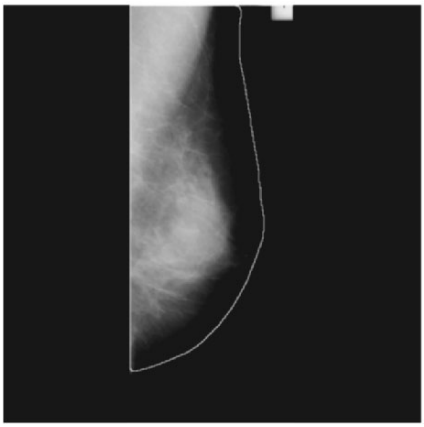
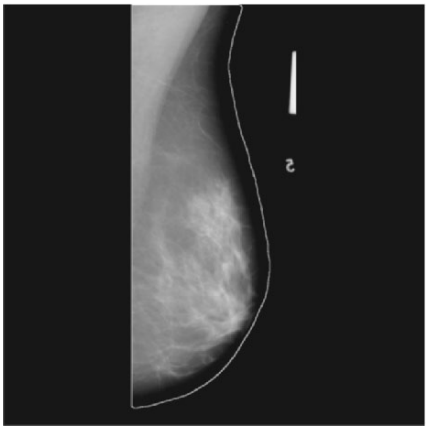


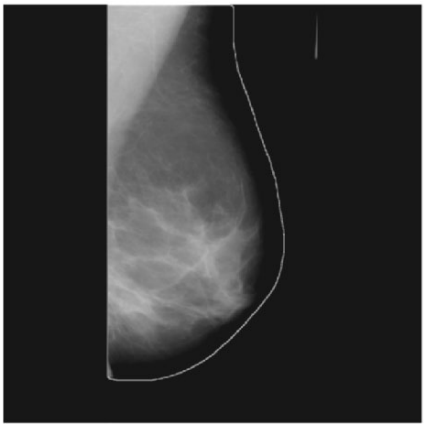
Figure 12. Results obtained for image mdb114. (a) Original image. (b) Hand-drawn boundary, superimposed on the histogram-equalized image. (c) Breast boundary detected automatically, superimposed on the original image.



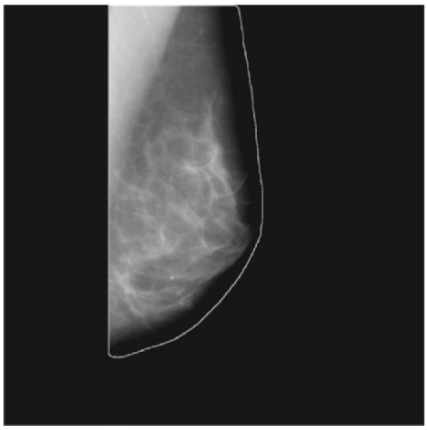
(a) mdb035



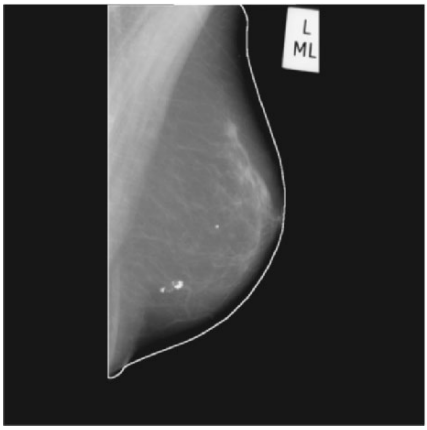
(b) mdb044



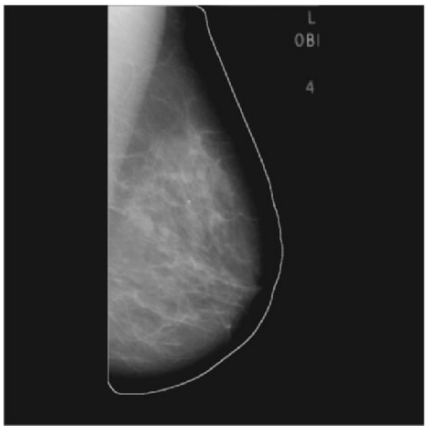
(c) mdb052



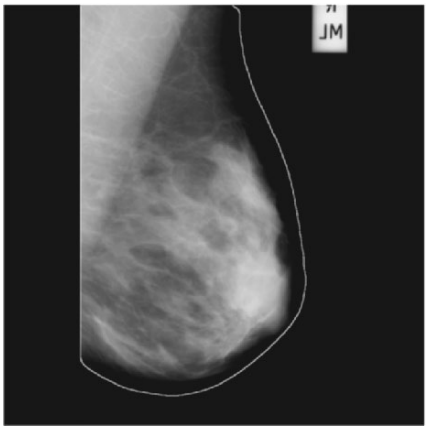
(d) mdb056



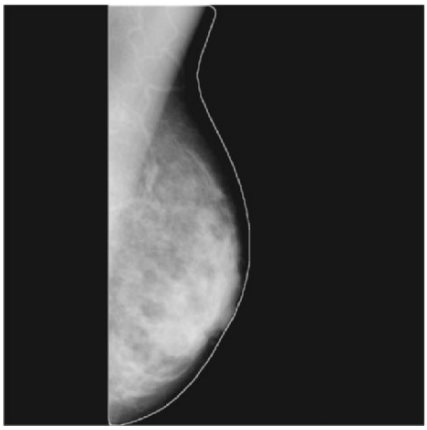
(e) mdb075



(f) mdb091



(g) mdb116



(h) mdb124

Figure 13. Results of segmented images from the Mini-MIAS database.

The most common cause of FN pixels was related to the non-detection of the nipple region [see the example in Figure 10(c)]. By removing Stage 5, used to approximate the initial contour to the true breast boundary (which is used only in Method 1 described in Section 2), we decreased the average time for processing an image from 2.0 to 0.18 min. However, the number of images where the nipple was not identified increased [see, e.g., Figures 10(b) and (c)].

The main cause of FP pixels was associated with smoothing of the limits of the breast boundary. Figure 14 presents a case where both the FP and FN percentages are greater than 1%. In the case of image mdb068, the automatically obtained initial contour was attracted to a high-density region inside the breast, instead of growing outward in the direction of the true breast boundary.

Due to the use of the gradient orientation in our active contour model, the method has shown good results in cases where small artifacts are present near the breast boundary (compare Figures 4 and 7). The parameters of our active contour model, although selected experimentally, have proven to be robust, as indicated by the results obtained.

The processing time to perform identification of the breast boundary in 256×256 images is about 0.1 min on average, using an 850-MHz computer with 512 Mb of memory. (Processing a 1024×1024 image took, on average, about 3 min.)

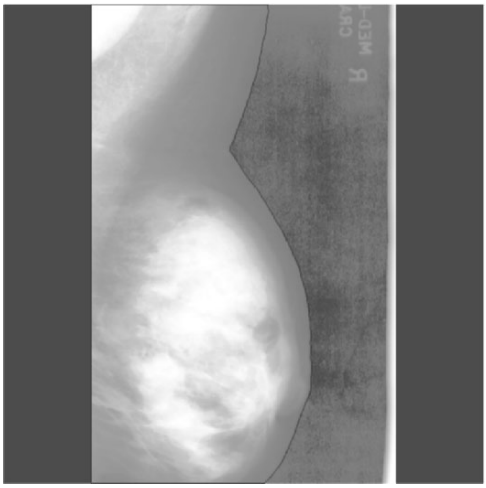
5. CONCLUSIONS

By using an energy formulation that includes a balloon force and an adaptive gradient force, the proposed adaptive active deformable contour model reduces the influence of the initial contour on the final results. Therefore, we could eliminate a time-consuming preprocessing stage used previously to determine an initial breast contour.

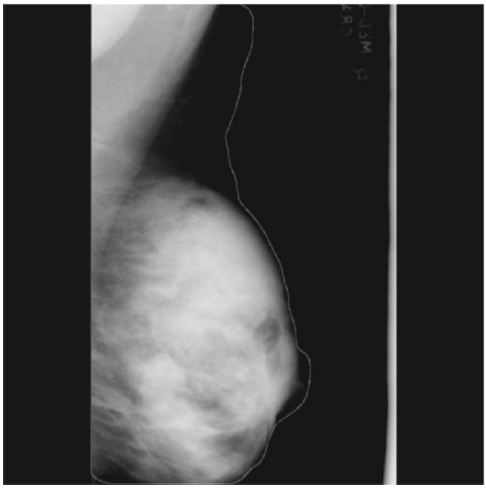
According to the opinion of the two radiologists involved in the study, the overall results of the proposed method to detect breast boundaries in mammograms, applied to 84 images from the Mini-MIAS database [13], are encouraging for application in the preprocessing stages of CAD systems. The method proposed in this work can also be used in other applications in mammography, such as image compression by using only the effective area of the breast, and image registration. The method has been applied to analysis of bilateral asymmetry [19].

6. ACKNOWLEDGEMENTS

This project was supported by grants from Coordenação de Aperfeiçoamento de Pessoal de Ensino Superior (CAPES) and Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Brazil; the Alberta Heritage Foundation for Medical Research (AHFMR), Alberta, Canada; and the Natural Sciences and Engineering Research Council (NSERC) of Canada.



(a)



(b)

Figure 14. Image mdb068, presenting problems in segmentation of the breast boundary. (a) The hand-drawn boundary, superimposed on the histogram-equalized image. (b) Automatically detected boundary superimposed on the original image. The active contour has been attracted to a high-density region in the breast. Such problems may be solved by equalizing the image contrast before application of the method.

7. REFERENCES

1. Lou SL, Lin HD, Lin KP, Hoogstrate D. 2000. Automatic breast region extraction from digital mammograms for PACS and telemammography applications. *Comput Med Imaging Graphics* **24**:205–220.
2. Bick U, Giger ML, Schmidt RA, Nishikawa RM, Doi K. 1996. Density correction of peripheral breast tissue on digital mammograms. *RadioGraphics* **16(6)**:1403–1411.
3. Byng JW, Critten JP, Yaffe MJ. 1997. Thickness-equalization processing for mammographic images. *Radiology* **203(2)**:564–568.
4. Chandrasekhar R, Attikiouzel Y. 1997. A simple method for automatically locating the nipple on mammograms. *IEEE Trans Medical Imaging* **16**(5):483–494.
5. Lau TK, Bischof WF. 1991. Automated detection of breast tumors using the asymmetry approach. *Comput Biomed Res* **24**:273–295.
6. Miller P, Astley S. Automated detection of mammographic asymmetry using anatomical features. 1993. *Int J Pattern Recognit Artif Intell* **7**(6):1461–1476.
7. Méndez AJ, Tahoces PG, Lado MJ, Souto M, Correa JL, Vidal JJ. 1996. Automatic detection of breast border and nipple in digital mammograms. *Comput Methods Programs Biomed* **49**:253–262.
8. Bick U, Giger ML, Schmidt RA, Nishikawa RM, Wolverton DE, Doi K. 1995. Automated segmentation of digitized mammograms. *Acad Radiol* **2**(1):1–9.
9. Masek M, Attikiouzel Y, deSilva CJS. 2000. Combining data from different algorithms to segment the skin–air interface in mammograms. In *Proceedings of the 22nd annual EMBS international conference*, Vol. 2, pp. 1195–1198. Washington, DC: IEEE.
10. Ferrari RJ, Rangayyan RM, Desautels JEL, Frère AF. 2000. Segmentation of mammograms: identification of the skin–air boundary, pectoral muscle, and fibro-glandular disc. In *Proceedings of the 5th international workshop on digital Mammography*, pp. 573–579. Ed MJ Yaffe. Madison, WI: Medical Physics Publishing.
11. Kass M, Witkin A, Terzopoulos D. 1988. Snakes: active contour models. *Int J Comput Vision* **1**(4):321–331.
12. Gonzalez RC, Woods RE. 1992. *Digital image processing*. Reading, MA:: Addison-Wesley.
13. Suckling J, Parker J, Dance DR, Astley S, Hutt I, Boggis CRM, Ricketts I, Stamatakis E, Cerneaz N, Kok SL, Taylor P, Betal D, Savage J. 1994. The mammographic image analysis society digital mammogram database. In *Proceedings of the 2nd international workshop on digital mammography*, pp. 375–378. Ed AG Gale, SM Astley, DR Dance, AY Cairns. *Excerpta Medica International Congress Series*, Vol. 1069. Amsterdam: Elsevier.
14. Lloyd S. 1982. Least squares quantization in PCM. *IEEE Trans Inf Theory* **28**:129–137.
15. Mackiewich B. 1995. *Intracranial boundary detection and radio frequency correction in magnetic resonance images.* Master's thesis, School of Computing Science, Simon Fraser University, Burnaby, BC, Canada.
16. Lobregt S, Viergever MA. 1995. A discrete dynamic contour model. *IEEE Trans Med Imaging* **14**(1):12–24.
17. Williams DJ, Shah M. 1992. A fast algorithm for active contours and curvature estimation. *Comput Vision Graphics Image Proces: Image Understand* **55**(1):14–26.
18. Mattis P, Kimball S. 2005. *GIMP–GNU image manipulation program*. http://www.gimp.org.
19. Ferrari RJ, Rangayyan RM, Desaultels JEL, Frère AF. 2001. Analysis of asymmetry in mammograms via directional filtering with Gabor wavelets. *IEEE Trans Med Imaging* **20**(9):953–964.
20. Mackiewich B, Desautels JEL, Borges RA, Frère AF. 2004. Intracranial boundary detection and radio frequency correction in magnetic resonance images. *Med Biol Eng Comput* **42**(2):201–208.

6

STATISTICAL DEFORMABLE MODELS FOR CARDIAC SEGMENTATION AND FUNCTIONAL ANALYSIS IN GATED-SPECT STUDIES

C. Tobon-Gomez, S. Ordas, and A.F. Frangi

Computational Imaging Laboratory, Universitat Pompeu Fabra, Barcelona, Spain

S. Aguade and J. Castell

Vall d' Hebron University Hospital, Barcelona, Spain

This chapter describes the use of statistical deformable models for cardiac segmentation and functional analysis in Gated Single Positron Emission Computer Tomography (SPECT) perfusion studies. By means of a statistical deformable model, automatic delineations of the endo- and epicardial boundaries of the left ventricle (LV) are obtained, in all temporal phases and image slices of the dynamic study. A priori spatio-temporal shape knowledge is captured from a training set of high-resolution manual delineations made on cine Magnetic Resonance (MR) studies. From the fitted shape, a truly 3D representation of the left ventricle, a series of functional parameters can be assessed, including LV volume–time curves, ejection fraction, and surface maps of myocardial perfusion, wall motion, thickness, and thickening. We present encouraging results of its application on a patient database that includes rest/rest studies with common cardiac pathologies, suggesting that statistical deformable models may serve as a robust and accurate technique for routine use.

Address all correspondence to: Catalina Tobon-Gomez, Computational Imaging Laboratory, Universitat Pompeu Fabra, Pg Circumvallacio 8, Barcelona 08003, Spain. Phone: +34 93-542-1451, Fax: +34 93-542-2517. catalina.tobon@upf.edu.

1. INTRODUCTION

Over the course of the last two decades, myocardial perfusion with Single Photon Emission Computed Tomography (SPECT) has emerged as an established and well-validated method for assessing myocardial ischemia, viability, and function. Such a technique has a widespread use in clinical cardiology practice, with a diagnostic accuracy and prognostic importance repeatedly confirmed in numerous studies, allowing for evaluation of patients with known or suspected coronary artery disease (CAD). These studies now represent an extensive database unequaled in the field of cardiac imaging. Gated-SPECT imaging integrates traditional perfusion information along with global left ventricular function. Therefore, identification of the presence, extent, and severity of irreversible and reversible perfusion defects, plus ventricular function, can be achieved within a single study. This enables, for instance, to effectively stratify patients into subgroups at low and high risk for acute cardiac events and subsequent death. Entering the new century, the field has matured to the point at which SPECT data are accepted as indispensable in the management of the decision-making process for many patients. Moreover, SPECT is also being incorporated as a cornerstone in the design of multicenter clinical trials [1–8]. This circumstance will undoubtedly have a major impact in the treatment of patients with acute coronary syndromes and chronic CAD, as well as in the evaluation of patients with heart failure and those with diabetes.

1.1. Chapter Outline

This chapter is organized in six sections. Section 1 introduces gated SPECT as imaging technique and most currently employed segmentation methods. Section 2 explains an automatic segmentation methodology based on shape and gray-level statistics priors. Section 3 describes the experiments conducted during this work. Section 4 presents the results obtained by previously referred methodology, followed by a discussion in Section 5. The final section exposes some conclusions derived from our research in the theme.

1.2. Imaging Description

In gated-SPECT imaging, acquisition is synchronized with the patient's electrocardiogram. RR interval is divided into equally spaced temporal phases. A full-volume data set is obtained for each temporal phase. Cardiac cycles around 10–20% of the mean RR interval are selected for acquisition. Even though this modality may acquire up to 32 frames per cardiac cycle, clinical routine studies usually include only 8 frames. Figure 1 shows an explanatory diagram of a gated-SPECT acquisition procedure. Data corresponding to the first temporal frame are collected at a certain position. After rotation of the detector, a new set of data is stored. This continues repeatedly over a 180-degree trajectory. Subsequently, a

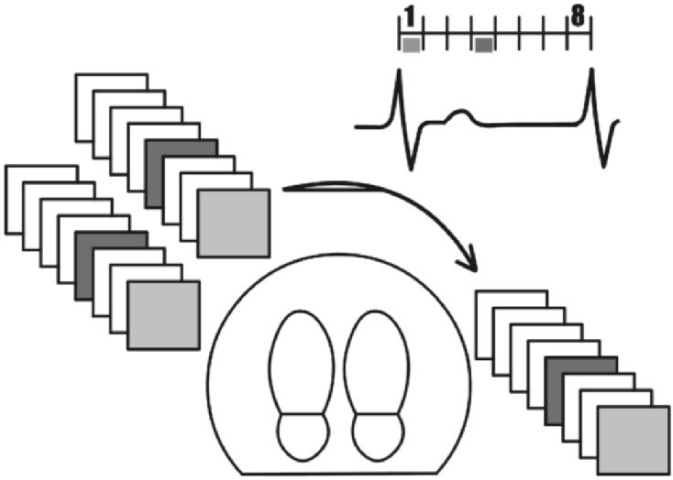


Figure 1. Gated-SPECT description. RR interval is divided into equally spaced temporal phases. A full volume dataset is obtained for each temporal phase.

reconstruction algorithm generates tomographic datasets of each temporal frame, which enables both perfusion and functional analysis. Figure 2 shows examples of produced images in a gated-SPECT rest study.

1.3. Systolic and Diastolic Function

Along with its traditional application on myocardial perfusion assessment, gated-SPECT imaging is widely used for evaluating systolic ventricular function (SFx) [9, 10]. SFx analysis comprises the most typically employed variables in clinical practice: End Diastolic Volume (EDV), End Systolic Volume (ESV), Ejection Fraction (EF) and the Volume–Time Curve (VTC), with EF defined as:

$$\mathrm{EF} = \frac{\mathrm{EDV} - \mathrm{ESV}}{\mathrm{EDV}} \times 100. \tag{1}$$

According to recent studies, assessment of diastolic function (DFx) is also achievable from gated-SPECT datasets [11]. Abnormalities in DFx have been recently emphasized [12–17] as much earlier predictors of LV dysfunction. Nonetheless, there is still unclarity regarding which DFx parameters are more stable and useful. Currently in gated-SPECT imaging, the two most exploited measurements are Peak Filling Rate (PFR) and Time to Peak Filling Rate (TTPF). For the purpose of calculating them, the filling rate vs. time curve is computed from the VTC first derivative. Subsequently, PFR is estimated as the maximum value of this

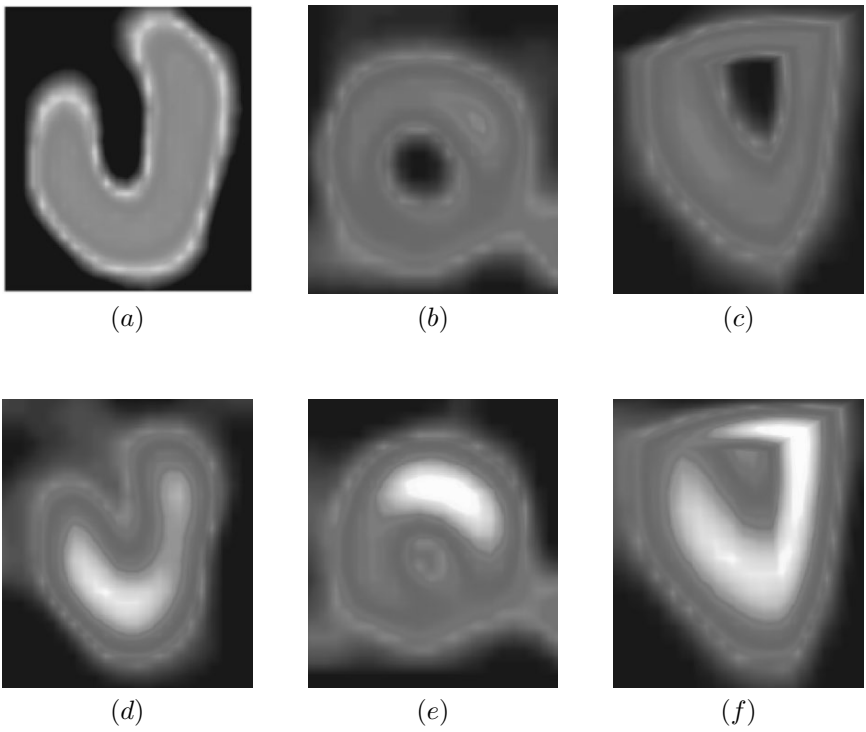


Figure 2. Example of gated-SPECT study. (a) Long axis, (b) short axis and (c) 3D views at End-diastole. (d) Long axis, (e) short axis and (f) 3D views at End-systole. See attached CD for color version.

curve, divided by the EDV in order to normalize its value (EDV/s). TTPF (in ms) corresponds to the time elapsed between ESV and PFR [11].

1.4. Challenges in Gated-SPECT Image Analysis

Despite the advantages detailed in Section 1, inherent limitations of SPECT imaging yield a challenging segmentation problem. The specific aim in nuclear medicine, and in particular ^{201}Tl and ^{99m}Tc cardiac SPECT imagery, is to study the physiology and not the structure of the imaged organ. Such functional information can be misleading, especially in the presence of hypoperfused regions in the data. However, it is also desirable, and indeed possible, to infer structural information from these images. The principal complications for automated segmentation in cardiac SPECT include:

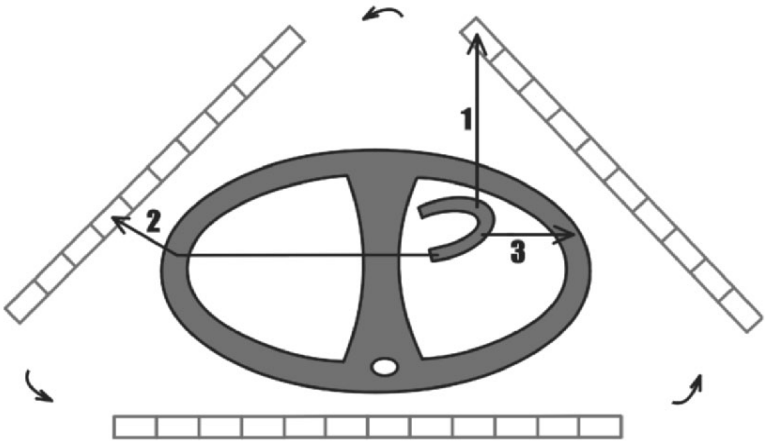


Figure 3. Three kinds of photons are present during image construction: (1) fully contributing photons, (2) scattered Photons, and (3) completely absorbed photons.

1. *Compton Scattering*: Part of the energy of the emitted photon may be absorbed by a surrounding structure. This results on a trajectory deviation that causes loss of resolution and contrast. Figure 3 illustrates this phenomenon.

2. *Photon Absorption*: Some photons are completely absorbed by a body structure. This causes inaccurate measurement of radioactivity on a given myocardial region that may result in artifactual perfusion defects. Refer to Figure 3 for a graphic depiction.

3. *Signal Drops*: In pathologic perfusion SPECT studies, situations with substantial sparse or missing information due to perfusion defects are quite common. View Figure 4 for examples.

4. *Image Resolution*: Current resolutions of SPECT imaging systems range from 4 to 7 mm, resulting in small-sized 3D data sets. Thus, small errors ($<$1 voxel) in segmentation can lead to considerable deviations in LV function parameter estimation.

5. *Patient Motion*: Patient motion creates artificial perfusion defects. Displacements of 2 or more pixels may result on clinically significant perfusion defects [18].

6. *Heart Motion*: The constant "beating" of the heart has an additional blurring effect on the already low-resolution data.

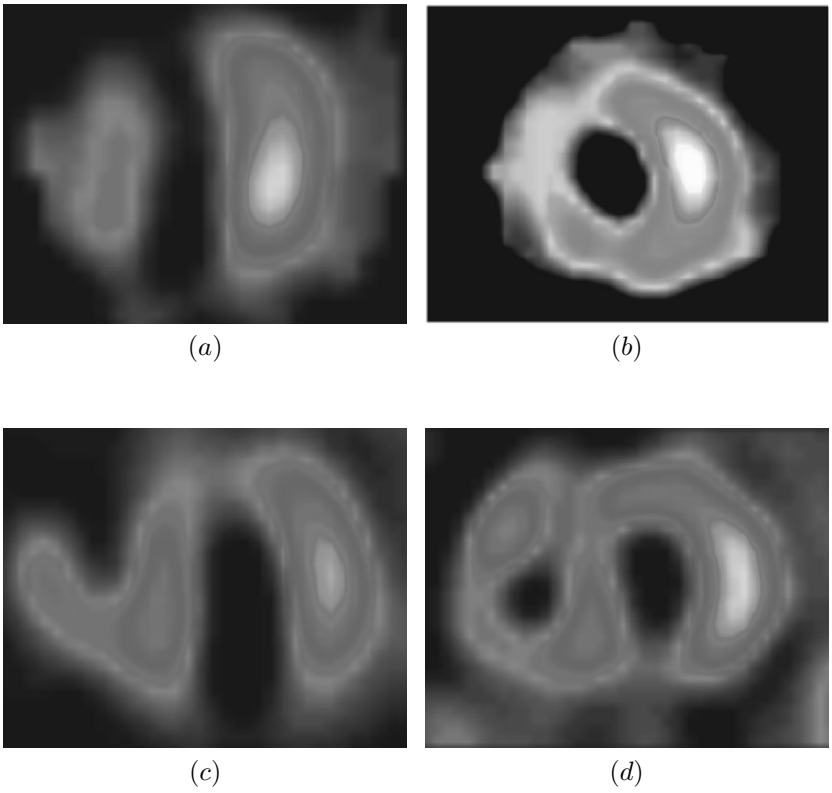


Figure 4. Some of the challenges inherent to gated-SPECT imaging: perfusion defects. (a)–(c) Long axis and (b)-(d) short axis views. See attached CD for color version.

7. *Absence of Anatomical Landmarks*: Valve planes (mitral and aortic), papillary muscles, and apex are anatomical landmarks commonly used by segmentation methods. They are not visible in SPECT imaging due to low tracer uptake. This fact greatly complicates automated basal plane definition. Also, apex determination may be confusing on images with perfusion defects (see Figure 3).

8. *Partial Volume Effect*: Partial volume effect establishes that the average counts measured in a structure are not only proportional to the amount of activity contained in that structure, but also to the size of the structure itself [19]. This principle causes the myocardium to appear thinner at the end of diastole (ED), when the activity count is lower, and thicker at end of systole (ES), due to a higher activity count.

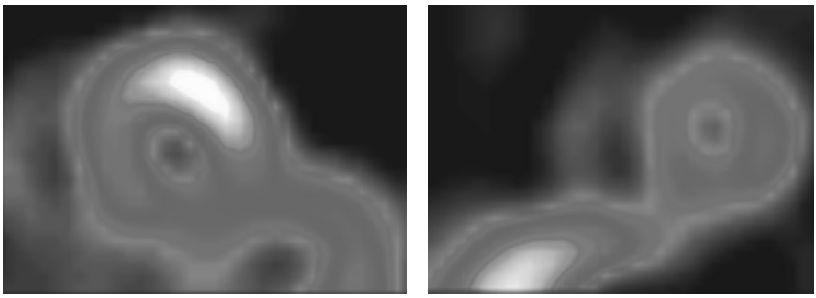


Figure 5. Some of the challenges inherent to gated-SPECT imaging: extra cardiac uptake. (a) Bowel uptake that does not affect activity counts. (b) Liver uptake that affects normalization of activity counts. See attached CD for color version.

9. *Extracardiac Uptake*: Liver and bowel tend to uptake a portion of the radiolabeled molecules administered to the patient. This aspect creates an inappropriate normalization of perfusion activity. It is also a regular source of complication for segmentation algorithms, as they tend to get attracted toward those stronger edges and areas (see Figure 5).

10. *Axial Slices*: Most algorithms for automatic segmentation operate on short-axis reformatted images. The LV long-axis usually has to be determined manually, which may be prone to inaccuracies and inter-operator variability [18].

Combining all these challenges, the data sets can be viewed as low-resolution, noisy, temporally integrated, and sparse. Therein lies the need for a robust segmentation methodology, since an error of only one voxel along the chamber surface may generate a huge difference in volume calculation and derived parameters.

1.5. Current Techniques for LV Segmentation

1.5.1. Description

The two most widespread approaches for automatic identification and segmentation of the LV in gated-SPECT studies in clinical practice are briefly commented upon below. These techniques start by identifying the LV region and LA in the transaxial images, either manually or using a threshold-based method [9, 20].

1. *Quantitative Gated-SPECT algorithm (QGS) [21]*: It begins by applying the dilation and eroding operations in order to isolate the myocardium from the other organs. The center of mass of this mask (which should be

located inside the LV blood pool; otherwise further refinement is required) constitutes the origin of a sampling coordinate system composed of radial count profiles. An ellipsoid is iteratively fit to a maximal count myocardial surface, which is considered to be located at the center geometry of the end-diastolic surface. The count distribution averaged over all intervals is mapped onto this surface. Finally, an asymmetric Gaussian profile is adapted to every profile and the locations of plus and minus one standard deviation define the inner and outer myocardial surfaces. The algorithm was implemented into the commercially available analysis package QGS, developed at Cedars-Sinai Medical Center (Los Angeles, CA, USA).

2. *Emory Cardiac Toolbox (ECTB) [10]*: This technique employs a geometric modeling approach in which maximal count circumferential profiles are obtained as the center of the myocardium. The algorithm assumes that the myocardial thickness at ED is 1 cm, and it calculates myocardial thickening along the cardiac cycle by using Fourier analysis. It assumes that the change in count is proportional to the change in thickness, owing to the partial-volume effect (Section 1.4). The center of the myocardium and its thickening (percentage) along the cardiac cycle are determined by determining the center and absolute thickness of the left ventricle myocardium at ED. The commercially available ECTB was developed at Emory University (Atlanta, GA, USA).

1.5.2. Complications

In order to cope with perfusion defects, the aforementioned algorithms use a large amount of parameters, rules, and criteria that are empirically determined. In addition, as mentioned before, extracardiac activity complicates segmentation, especially when the external uptake focus is close to the myocardium. We believe that the main drawback of these approaches is that their geometrical models are not appropriate for those circumstances. A priori information on the organ and confident information from other parts of the image (epicardium), should help determine a valid LV shape.

1.5.3. Comparison Studies

Several studies have been performed aiming to compare SFx analysis results from the previously described algorithms either within each other [1–3], with respect to MRI (regarded as the gold standard) [1, 4–8], or to a ground truth (phantoms) [22]. Most validation studies of QGS vs. MRI found a significant underestimation of EDV and EF by QGS. Regarding ECTB vs. MRI, significant underestimation of ESV estimated by ECTB has been manifested. As of ECTB vs. QGS, ECTB yielded lower values for ESV, while QGS revealed lower values for EDV and EF. The phantom study carried out by [22] confirmed these facts.

1.6. Statistical Deformable Models

In the last few years, quantification of images using model-based approaches has gained very much attention in the medical image analysis community [23]. The idea behind these approaches is to use a generic template model of the structure of interest, which is subsequently deformed to accommodate for the information provided by the image data.

Deformable surface-based segmentation of 3D and 4D ($3\mathrm{D} + t$) medical images make use of a priori knowledge about the shape of the target organ. Deformable surfaces produce a geometric representation of the segmented structures. They are composed of a geometric representation and an evolution law governing the surface deformations. A deformable surface model is sensitive to noise and data outliers. To avoid its free deformation, some regularization constraints have to be introduced into the deformation process. Prior information on the data may be incorporated by limiting the possible variations of the model to increase the robustness of the segmentation process. Many other complementary ways of relying on prior information have been proposed, such as statistical variability of shape [24] and intensity of anatomical structures in images [25]. This topic has been extensively studied, especially for 3D object reconstruction and image segmentation [26, 27].

Statistical model-based approaches allow learning shape statistics from a given population of training samples and, therefore, are able to construct a compact and specific anatomical model. Statistical models of shape (ASM) [24] and appearance (AAM) [28] variability are two model-driven segmentation schemes initially described by Cootes et al. and further employed by several groups in a variety of medical applications. 3D extensions of these methods have appeared [29–32], with an encouraging performance in delineating both epi- and endocardial borders of the left ventricle in cardiac Magnetic Resonance (MRI), Ultrasound (US), Computer Tomography (CT), and Single Positron Emission Computer Tomography (SPECT).

Deformable surfaces may either have a continuous representation or be defined as a discrete mesh with a finite set of vertices. Discrete meshes are defined by a finite set of vertices and a connectivity function between these vertices. Among the possible geometric representations of deformable surfaces, we have chosen a simple triangulated surface model. An obvious method to segment 4D datasets is to consider independently every 3D volume of the time sequence. The result of the segmentation at instant t may be used as an initialization for segmentation of instant $t + 1$ [33]. Even though more information on the sequence might be used by considering the whole 4D dataset [34], our approach considers each 3D volume independently for the 4D analysis.

2. THREE-DIMENSIONAL ACTIVE SHAPE MODELS (3D-ASM)

2.1. Overview

An Active Shape Model (ASM) [24] is an example of a statistical deformation model, and is composed of three parts: (1) a Point Distribution Model (PDM) that models the shape variability of an object, (2) an appearance model that describes gray-level patterns around the object's boundaries, and (3) a matching algorithm that drives the PDM deformation toward the target in an iterative manner. In the following, we briefly describe each constitutive part of the algorithm, but thorough descriptions are provided elsewhere [24, 28, 31].

2.2. Shape Model

Let $\{\mathbf{x}_i; i = 1 \cdots n\}$ denote n shapes. Each shape consists of m 3D landmarks, $\{\mathbf{p}_j = (p_{1j}, p_{2j}, p_{3j}); j = 1 \cdots m\}$, which represent the nodes of a surface triangulation. Each vector $\mathbf{x}_i$ is of dimension $3m$ and consists of landmarks $(p_{11}, p_{21}, p_{31}, \cdots, p_{1m}, p_{2m}, p_{3m})$. Moreover, assume that the positions of the landmarks of all shapes are in the same coordinate system. These vectors form a distribution in a $3m$-dimensional space. The goal is to approximate this distribution with a linear model of the form

$$\mathbf{x} = \hat{\mathbf{x}} + \mathbf{\Phi}\mathbf{b}, \tag{2}$$

where

$$\hat{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i \tag{3}$$

is the average landmark vector, $\mathbf{b}$ is the shape parameter vector of the model, and $\mathbf{\Phi}$ is a matrix whose columns are the principal components of the covariance matrix:

$$\mathbf{S} = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i - \hat{\mathbf{x}})(\mathbf{x}_i - \hat{\mathbf{x}})^T. \tag{4}$$

The principal components of $\mathbf{S}$ are calculated as its eigenvectors, ϕ_i, with corresponding eigenvalues, λ_i (sorted so that $\lambda_i \geq \lambda_{i+1}$). If $\mathbf{\Phi}$ contains the $t < \min\{m, n\}$ eigenvectors corresponding to the largest nonzero eigenvalues, we can approximate any shape of the training set, $\mathbf{x}$, using Equation (2), where $\mathbf{\Phi} = (\phi_1|\phi_2|\cdots|\phi_t)$ and $\mathbf{b}$ is a t-dimensional vector given by,

$$\mathbf{b} = \mathbf{\Phi}^T(\mathbf{x} - \hat{\mathbf{x}}). \tag{5}$$

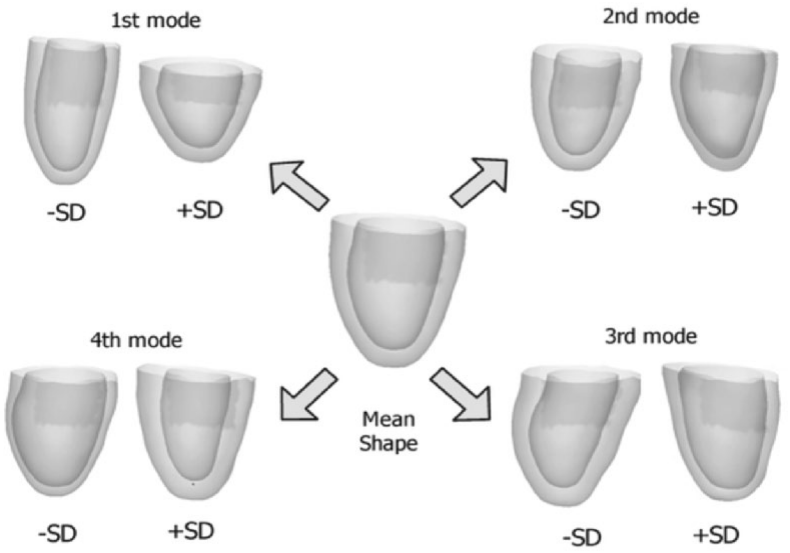


Figure 6. Principal modes of variation for the MRI (ED) PDM. The shapes are generated by varying a single model parameter, (b_i), fixing all others at zero standard deviations $(SD_i = \sqrt{\lambda_i})$ from the mean shape.

The vector $\mathbf{b}$ defines the shape parameters of the ASM. By varying these parameters we can generate different instances of the shape class under analysis using Equation (2). Assuming that the cloud of landmark vectors follows a multi-dimensional Gaussian distribution, the variance of the $i\text{th}$ parameter, b_i, across the training set is given by λ_i. By applying limits to the variation of b_i, for instance $|b_i| \leq \pm 3\sqrt{\lambda_i}$, it can be ensured that a generated shape is similar to the shapes contained in the training class.

Obtaining the m 3D landmarks and their correspondence among training examples is a non-trivial task. Such a problem still represents an open topic for the shape analysis community. Our methodology was inspired on the method proposed by Frangi and coworkers [35]. The method can easily be set to build either 1- or 2-chamber models. In this work, a single-ventricle configuration with 2848 points (1071 endocardial and 1777 epicardial) was used. The constructed PDM was trained from a population of 90 hearts with large intra- and inter-individual variability, including both healthy and pathologic examples, in ED and ES temporal phases. Thus, 180 examples were in total considered [29]. Figure 6 presents the four principal modes of variation for the MRI (ED) PDM.

2.3. Appearance Model

Our 3D-ASM implementation uses a simple appearance model, which can be regarded as a 3D extension of the one used in the original 2D-ASM [24]. In that formulation, the image structure around each landmark is represented from first-derivative profiles perpendicular to the object contour, sampled with bilinear interpolation. Taking k positions to each size, the total length is $2k+1$. The effect of global intensity changes is reduced by normalizing the patches such that the sum of the absolute intensity values equals 1:

$$\hat{\mathbf{g}}_i = \frac{g_i}{\sum_{k=1}^{L} |g_{ik}|}, \tag{6}$$

with $\mathbf{g}_{ik}$ the $k\mathrm{th}$ component in the $i\mathrm{th}$ profile. Denoting these normalized derivative profiles as $\hat{\mathbf{g}}_1, \ldots, \hat{\mathbf{g}}_s$, the mean profile $\overline{\mathbf{g}}$ and the covariance matrix $\Sigma_{\mathbf{g}}$ are computed for each landmark. The Mahalanobis distance,

$$f(\hat{\mathbf{g}}_i) = (\hat{\mathbf{g}}_i - \overline{\mathbf{g}})\Sigma_{\mathbf{g}}^{-1}(\hat{\mathbf{g}}_i - \overline{\mathbf{g}}), \tag{7}$$

between new boundary patches $\mathbf{g}_i$ to the mean of learned profiles $\overline{\mathbf{g}}$ is used as a boundary measure. Minimizing $f(\hat{\mathbf{g}}_i)$ is equivalent to maximizing the probability that $\hat{\mathbf{g}}_i$ is originated from the same mulitivariate Gaussian distribution assumed for the training profiles. $\Sigma_{\mathbf{g}}$ weights the variation among corresponding points in the training profiles.

Our aim is to have a general framework capable of segmenting the same organ in dynamic data volumes of different modalities and acquisition protocols, using an arbitrary number, position, and orientation of image slices or stacks, as well as any voxel dimension and anisotropy degree. In such a general and diverse scenario, it is unfeasible to have, neither from a geometrical nor from a computational point of view, a gray-level model for each point in the shape model, like in the 2D counterpart.

Therefore, we chose to follow the following approach. By dividing the shape model into regions, we calculated a mean profile and a covariance matrix of normalized gray-level profiles, for each region, and from the set of manual contours in all available image slices in the training set. The profiles were assigned to the different regions by closest proximity. The appearance models were built in a multi-resolution fashion, so there is a mean profile and a covariance matrix for every region and resolution level (see Section 2.4). In our implementation, a total of 33 appearance models were considered (17 epicardial and 16 endocardial). These regions coincide with AHA's anatomical segments [36].

In high-resolution cardiac images like MSCT and MRI, many details of the image are depicted. Consequently, the LV boundary can be obscured by surrounding tissue of similar gray value and structures with strong edges in close proximity. In addition, the size and appearance of corresponding LV regions and neighboring

structures vary considerably between patients as well as in one patient over time. Therefore, a problem inherent to a linear gray-level model is that the underlying assumption of a normal intensity distribution in practice does not hold. The problem would persist if other local image features for boundary localization are employed, but normalized gray-level profiles seem to be the most appropriate for a linear approach [37].

2.4. Matching Procedure

The 3D extension of the ASM algorithm (3D-ASM) has been put forward by van Assen et al. [38]. This methodology allows for using sparsely and arbitrarily oriented data [32].

This algorithm can be described as follows:

1. *Shape positioning*: The algorithm starts by roughly positioning the mean shape at the center of the heart. It is important to scale the initial shape so that it spans through all image slices and to roughly orient it perpendicular to the stack of short axis slices (Figure 7).

2. *Image planes intersect model's mesh*: The current shape model instance bisects the image planes at a set of intersection points that do not necessarily belong to the model's mesh. The intersection points can be assigned by closest proximity to each of the anatomical regions defined in the shape model. This action yields stacks of 2D contours composed of the intersections of image planes with individual mesh triangles (Figure 7).

3. *Search for candidate displacements in every image slice*: This is achieved by evaluating the Mahalanobis distance between the actual gray-level profiles and the mean profile stored for each region. Elected points will be given by the lowest Mahalanobis distance and lying on the intersected image planes.

4. *Mesh deformation*: Elected points will act as forces that are propagated to the rest of the mesh nodes in the following fashion. A set of source update vectors v_w are available at some nodes ϖ of the mesh, which are in closest proximity to the candidate points found in the image planes. These vectors are propagated to the rest of the mesh with a weight in the receiving nodes λ according to the geodesic distance ($\|\lambda - \varpi\|^2$) between the source and receiving nodes, weighted by a Gaussian-shaped kernel, the width of which is given by σ_p. The resulting weight is

$$\omega(\lambda, \varpi) = e^{-\frac{\|\lambda - \varpi\|^2}{4\sigma_p^2}}. \tag{8}$$

   Note that the actual weight for the source node ($\lambda = w$) is unity. To avoid propagation of an update over the entire shape model surface, propagation

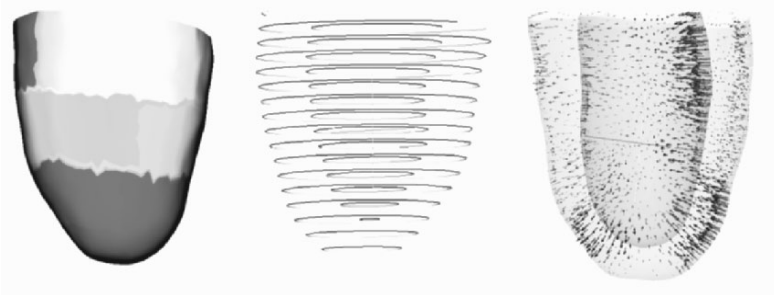


Figure 7. Convergence of the algorithm. Initialization (left), intermediate step (center), and convergence (right). See attached CD for color version.

is stopped when the geodesic distance exceeds a fixed threshold $\chi \equiv 3\sigma_p$. After all propagations stop, a total update per node is computed by summing over all contributions and normalizing with the sum of weights.

To facilitate through-plane motion, force vectors are projected to the surface's normals, which also have a component perpendicular to the image planes. Otherwise, the propagated forces would tend to retain the orientation of the intersected image planes.

5. *New valid instance*: The resulting deformed mesh is projected on the model's subspace, yielding a new instance of the shape model. This means finding the shape and the pose parameters of the shape model that best fits it, as explained in Section 2.2 (Figure 7).

Steps 2 to 5 are repeated either for a fixed number of iterations or until convergence is achieved, given some decision rule. A flowchart can be found in Figure 8.

2.4.1. Initialization

The initialization of segmentation methods is frequently regarded as a given precondition. In practice, however, initialization is usually performed manually or by some heuristic preprocessing steps. Therefore, it is of great importance to have a simple and effective initialization method at one's disposal.

Although the 3D-ASM is relatively insensitive to the initial shape model instance, a fully- or almost fully-automatic initialization is needed to produce consistent results. We followed a very simple mechanism to scale and position the mean shape of the model. The operator defines six epicardial points at the basal level and a seventh one in the apex. The centroids of corresponding anatomical regions in the mean shape are aligned to these points, with a similarity transformation. Figure 9 illustrates examples of the shape model initialization. Key benefits

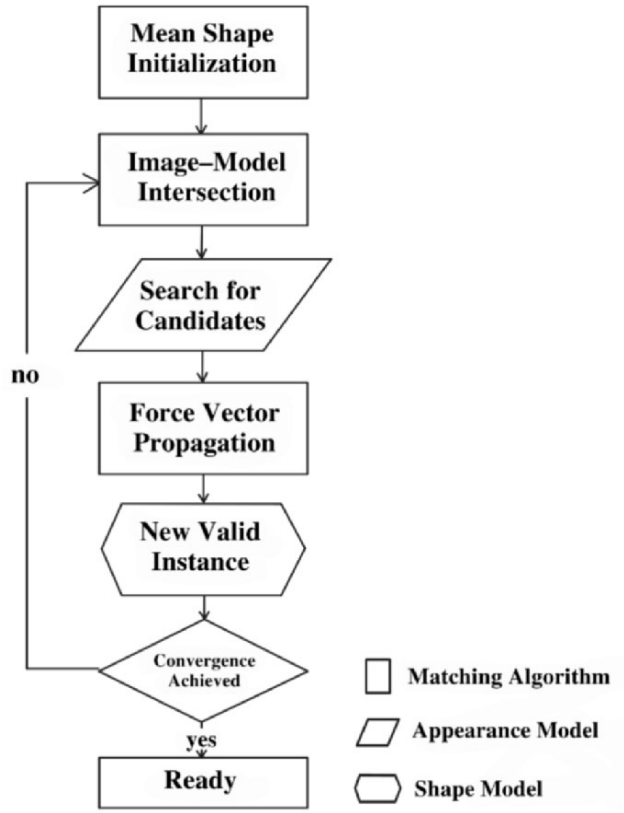
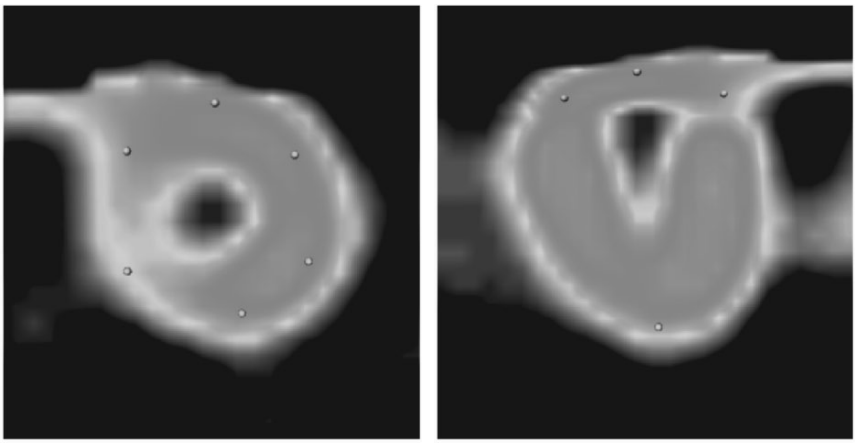


Figure 8. 3D-ASM matching procedure flowchart.

of such initialization are threefold. First, it provides a convenient initial position, orientation, and scale of the mean shape. This helps the robustness and speed of convergence of the algorithm because the first shape model instance is very close to the target. Second, it defines the basal plane. This helps producing more consistent calculations of volumes and derived parameters. Third, it positions the anatomical regions in the correct order. This is very important since anatomical regions should occupy, at least roughly, the same region both during training and matching.

2.4.2. Multi-Phase Segmentation

In segmenting multi-phase dynamic studies, only the first temporal phase needs the aforementioned semiautomatic initialization. To segment the rest of the



(a) (b)

Figure 9. Initialization points: (a) SA and (b) biplane views. See attached CD for color version.

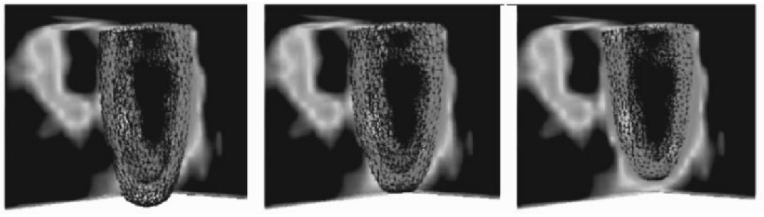


Figure 10. Convergence of the algorithm. Initialization (left), intermediate step (center), and convergence (right). See attached CD for color version.

phases in the study, the fitted shape of the previous phase is used, as well as less resolutions levels and iterations.

2.4.3. Convergence Criteria

A simple convergence test can be used to stop the deformation automatically when the mesh does not change significantly. Two convergence detection criteria can be adopted: the difference in LV volume and the total mesh displacement between two iterations. Results are robust with respect to both criteria. In our implementation we use a fixed number of iterations, 10 per resolution, which is large enough to prevent early interruption of the deformation and small enough to prevent useless iterations (Figure 10).

2.4.4. Multi-Resolution Strategy

A multi-resolution framework increases the capture range of the algorithm [24]. Blurred versions of the image result by applying a Gaussian pyramid filter. The iterative process starts at a coarse resolution and moves progressively to finer resolution levels (typically 2 for SPECT images), reducing the length of the target search region when approaching convergence. The finest level is the original data. A multi-resolution shape model could be used as well, as an effective way of decomposing the surface geometry into several levels of details at the different stages of the matching.

3. MATERIALS AND METHODS

3.1. Gated-SPECT Studies

Our dataset comprised 30 subjects. Two rest gated-SPECT studies were acquired on the same day with a 30-minute interval, for a total of 60 studies, at a rate of eight frames per cardiac cycle. This allows for validating each patient with him/herself since clinical situation remains constant. Patients were imaged 1 hour after administration of $^{99\mathrm{m}}Tc$-tetrofosmin using a Siemens ECAM SPECT (Siemens Medical Systems, IL, USA) system with a double-detector at 90° with high-resolution collimators. Sixty-four projections of a 64×64 matrix over a 180° arc were acquired with 6.6 mm/pixel of acquisition resolution for all rest gated studies. In the reconstruction with FBP (filtered back-projection) the image data were reformatted into standard short-axis slices with a voxel dimension of 4.8 mm^3, using the software provided by the acquisition system.

3.2. Patient Database

The database was constituted with patients referred for routine rest/rest myocardial perfusion evaluation at the Vall d'Hebron University Hospital of Barcelona. It included a total of 30 subjects (19 males, 11 females), of which 15 were healthy and 15 pathologic patients. The diagnoses for these patients consisted of: (a) myocardial infarction ($n = 6$), of which 3 were extensive (dilated $n = 2$, nondilated $n = 1$) and 3 moderate (dilated $n = 1$, ischemic $n = 1$); (b) ischemia ($n = 8$), of which minor ($n = 4$), moderated ($n = 3$), and intensive ($n = 1$), reached at various locations, and (c) non-ischemic dilated cardiomyopathy ($n = 1$). Table 1 offers a description of the database.

3.3. Experiments

For this application task the previously described PDM was applied without further training (see Section 2.2). Aiming to match the profile of intensities particular to SPECT modality, the appearance model was retrained from manually delineated endocardial and epicardial borders.

Table 1. Clinical Condition of Patients in Database

Description			Number
Healthy			15
Myocardial Infarction	Extensive	Dilated	2
		Non-Dilated	1
	Moderated	Dilated	1
		+Ischemia	1
		Non-Dilated	1
Ischemia	Minor		4
	Moderated		3
	Intensive		1
Dilated Cardiomyopathy	Non Ischemic		1

Table 2. 3D-ASM Settings

General
- **s**: number of training sets from which the shape model was built (180)
- **a**: number of training sets from which the appearance model was built (1)

Shape Model
- **n**: number of landmark points (2848)
- **fv**: part of variance to be explained by the shape model (0.50)
- **t**: number of modes in the shape model (4)
- **m**: bounds on shape model parameters (3.0)

Appearance Model
- **k**: number of points in profile on either side of the landmark point, giving profiles of length 2k + 1 (5)

Matching Procedure
- **ns**: number of new positions to evaluate at each side of current landmark position; the total number of positions is 2ns + 1 (7)
- **L**: number of levels in the multi-resolution strategy (2)
- **N**: number of iterations per resolution level (20)

In order to retrieve the optimal settings, and hence increase the efficiency of the process, several parameter assemblies where evaluated. Final 3D-ASM settings are summarized in Table 2.

3.4. Functional Analysis

Once segmentation is achieved, it is highly desirable to obtain functional parameters. Hence, both systolic and diastolic functional analysis were performed. Subsequently, interstudy variability was evaluated. In order to achieve this, the cavity volume of the fitted shape at each temporal stage was computed for the two rest studies of each patient. VTCs were constructed through cubic spline interpolation. From these curves EDV, ESV, EF, PFR and TTPF were calculated (see Section 1.3). Performed statistical analysis is outlined in Section 3.6.

3.5. Quantitative Analysis Tools

Other automatic quantitative analysis and visualization tools for regional functional assessment are available in our platform. Further improvement by incorporating normal ranges for patient assessment is mandatory. Their accuracy is yet to be evaluated. These tools may be described as follows:

1. *Myocardial Central Surface*: For every point in the endocardial surface of the fitted shape, the shortest distance to the epicardial surface is assessed. A myocardial center surface is built by collecting the middle points in all these segments and using the same mesh connectivity of the endocardial surface (Figure 11).

2. *Wall Thickness*: A wall thickness map is constructed by simply defining a scalar for every point in the middle surface with a value proportional to the longitude of the corresponding segment (Figure 12).

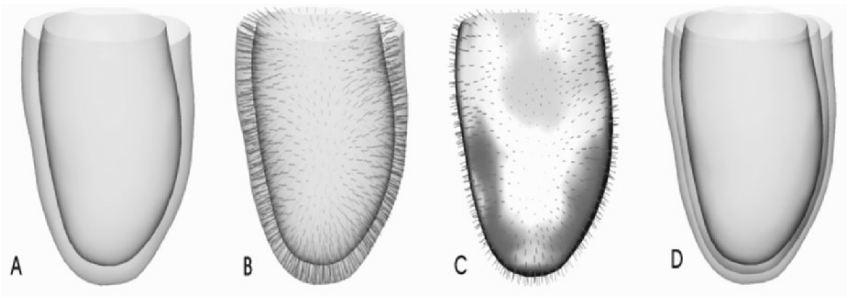


Figure 11. Central surface calculation. For every point in the endocardial surface, the shortest distance to the epicardial surface is assessed. The middle points in all these segments are collected and based on the same mesh connectivity of the endocardial surface; a new mesh is built. (a) Original surface; (b–c) segment evaluation, and (d) generated central surface. See attached CD for color version.

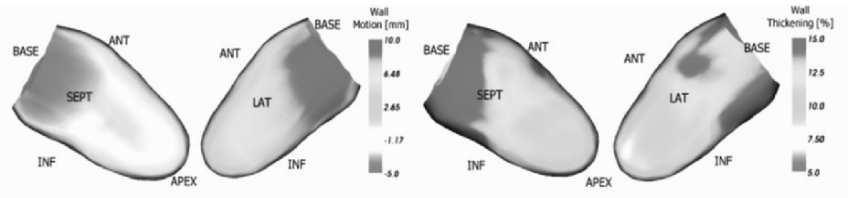


Figure 12. Left: wall motion, ED and ES, respectively. **Right**: wall thickening assessment, ED and ES, respectively. See attached CD for color version.

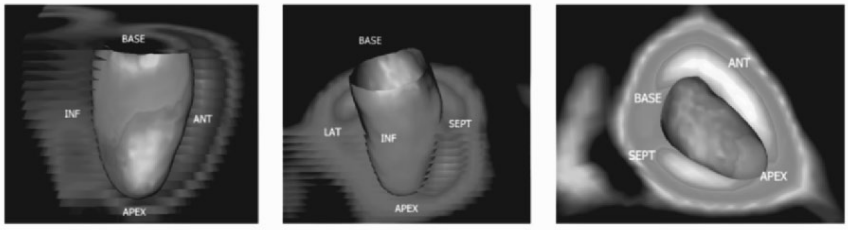


Figure 13. Perfusion map corresponding to ES displayed in different views. See attached CD for color version.

3. *Wall Motion*: The endocardial surface of the ED fitted shape is set as a reference for motion calculation. The distance between the reference shape and the remaining fitted shapes is computed. The distance grants the surface a scalar value for further visualization (Figure 12).
4. *Perfusion Map*: A scalar is defined for every intersection point between the endocardial surface and the image planes, with a value proportional to the average of counts along the corresponding profile. The rest of the points in the mesh are assigned an interpolated value resulting from averaging the nearest intersection points, weighted by the geodesic distance to them (Figure 13).
5. *Endocardial Excursion*: In our approach it is possible to fix the position either of the shape model centroid, the LV long axis, or none of them. Any combination of the previous maps can easily be rendered in the 3D animation tool (Figure 14).
6. *Perfusion Defect Area Assessment*: A surface representation is built only considering those portions of the perfusion map that are outside the normal captation range defined from a population of normal studies or with respect to rest/stress states of the same patient.

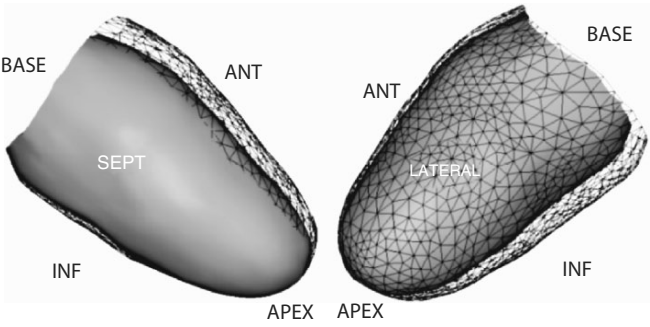


Figure 14. Endocardial Excursion. See attached CD for color version.

Any combination of the previous maps can easily be rendered in the 3D animation tool. The minimum and maximum values for the scalar ranges of the previously described tools can be calculated from a selectable population of normal studies or from rest/stress states of the same patient.

3.6. Statistical Analysis

EDV, ESV, EF, PFR, and TTPF measurements for both rest studies were obtained with 3DASM and QGS. Interstudy repeatability of all parameters was assessed by means of Bland-Altman plots [39]. In these graphs, the difference of two measurements is plotted against the averages of the two measurements. For our case, each vertical and horizontal axis represent Rest 1 – Rest 2 and mean(Rest 1, Rest 2) values, respectively.

4. RESULTS

Quantitative, and a few visual, results will be presented in this section and further discussed in Section 5. Total ranges on all parameters calculations are summarized in Table 3. Mean $\pm$ SD values are shown in Table 4. Figures 15 and 16 display the results of Bland-Altman analysis. According to them, 3DASM achieved lower SD values for EF, ESV, and PFR, while QGS computed smaller SD values for EDV and TTPF.

Regarding systolic function, two main types of discrepancies between QGS and 3DASM calculations were detected during this work:

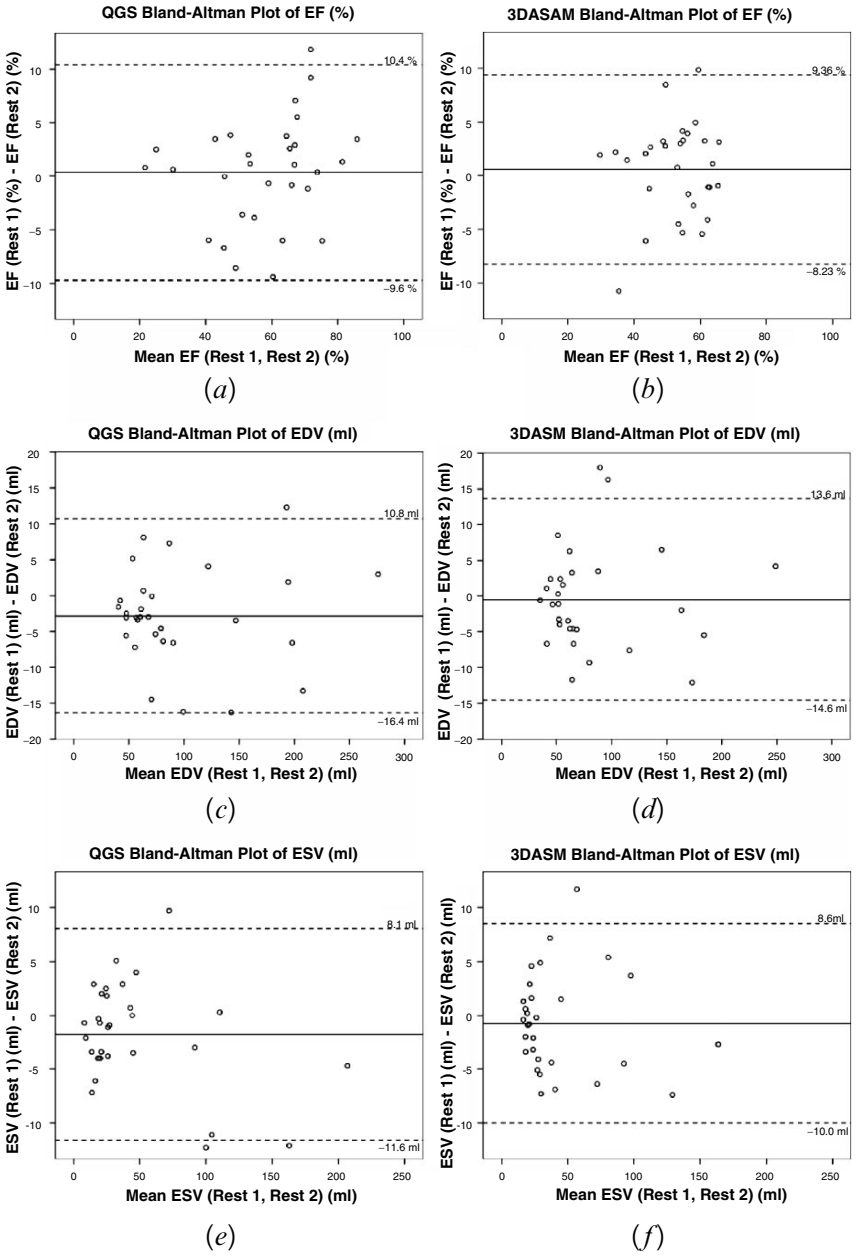


Figure 15. Bland-Altman Plots of EF (a–b), EDV (c–d), and ESV (e–f) comparing first and second rest studies estimated with QGS (left) and 3DASM (right).

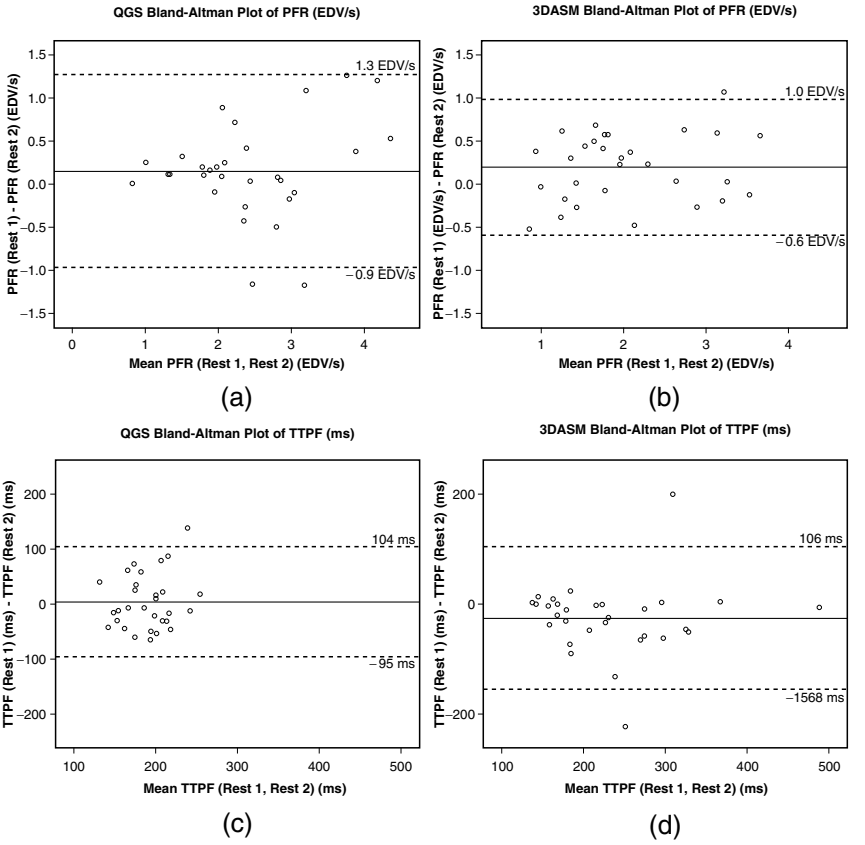


Figure 16. Bland-Altman Plots of PFR (a–b) and TTPF (c–d) comparing first and second rest studies estimated with QGS (left) and 3DASM (right).

1. *Small Hearts*: In our experience, QGS tends to underestimate the ESV of small hearts, generating an exaggerated EF. 3DASM, instead, incorporates the necessary anatomical constraints to impede further reduction of LV cavities; hence, it computes more likely results under physiological conditions. Various cases with these characteristics were present in our database. They in fact displayed the greatest difference on EF calculations by both methodologies. For these cases, QGS computed ESV values between 8 and 20 ml, with their respective EF values ranging from 59 to 88%, while our method computed ESV values ranging from 16 to 29 ml, and EF values between 50 and 67%. Samples of 3DASM segmentation results at ES for this case group are displayed in Figure 17.

Table 3. Statistical Analysis Results

	EDV ml	ESV ml	EF %	PFR EDV/s	$TTPF$ ms
$3DASM$	35 to 251	16 to 165	28.7 to 67.2	0.60 to 3.94	136 to 491
QGS	40 to 277	8 to 209	21.2 to 87.6	0.82 to 4.78	111 to 308

Values are displayed as min to max.

Table 4. Statistical Analysis Results

	EDV ml	ESV ml	EF %	PFR EDV/s	$TTPF$ ms
$3DASM$	82 ± 51	43 ± 36	52.6 ± 9.9	2.1 ± 0.84	232 ± 86
QGS	97 ± 60	47 ± 47	57.9 ± 15.9	2.4 ± 0.92	190 ± 39

Values displayed as mean $\pm$ SD.

2. *Dilated Hearts*: Lomsky et al. [22] demonstrated, using a digital phantom as the gold standard, that QGS tends to overestimate both EDVs and ESVs for large-heart datasets. Our work revealed great differences on volume calculations for dilated hearts, both on EDVs and ESVs. For these cases, QGS computed EDV values between 151 and 277 ml, and ESV values ranging from 90 to 209 ml, while our method computed EDV values ranging from 120 to 251 ml, and EDV values between 75 and 162 ml. Three examples of this type of discrepancy are illustrated in Figure 18. ED segmentation results are displayed on the left and ES results on the right.

As for diastolic function, calculations revealed larger repeatability errors for TTPF. This great variation is mainly because of two factors: time difference of ES and time difference to PFR. Due to the low resolution of clinical routine datasets, the filling rate curve is highly dependent on VTC interpolation. The amplitude and shape of the first-derivative curve varies drastically with distinct interpolation techniques. Evaluating smoothness-related variations on the curve generated, it was observed that both the PFR and TTPF calculations vary considerably. Figure 19 shows examples of the interpolation obtained with (a–b) restrictive interpolation, which generates a first derivative curve with more bends and greater amplitude and (c–d) smoother interpolation, which generates a more regular derivative and smaller peak value.

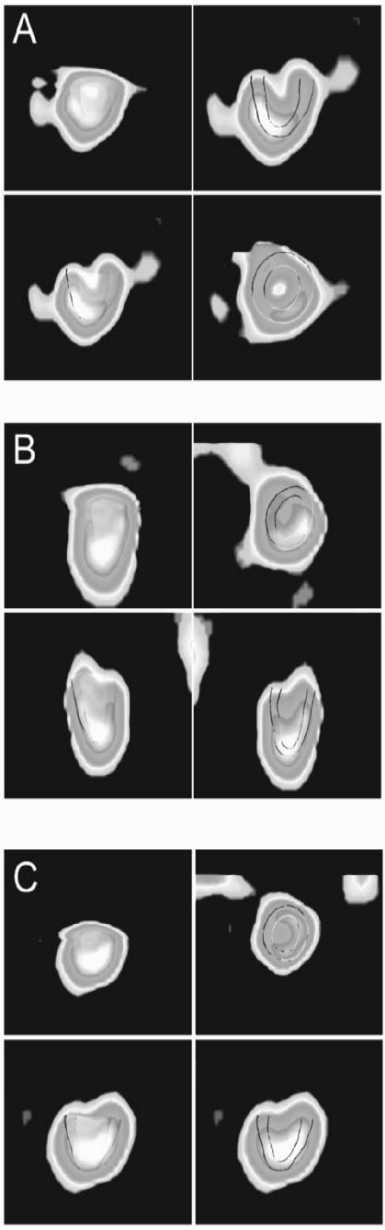


Figure 17. (a–c) Some examples of end systole segmentation results obtained with 3DASM of cases with the greatest difference on EF calculations. See attached CD for color version.

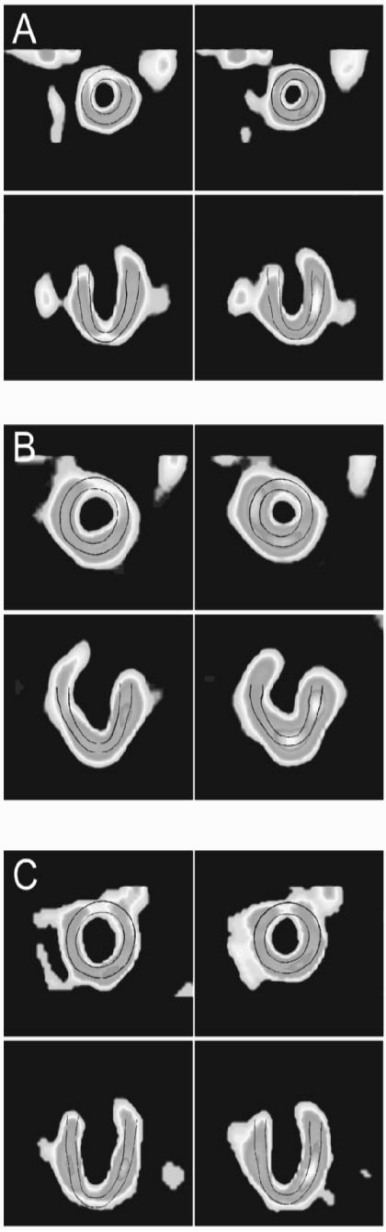


Figure 18. (a–c) Segmentation results obtained with 3DASM of cases with the greatest difference on EDV calculations. ED segmentation results are displayed on the left and ES results on the right. See attached CD for color version.

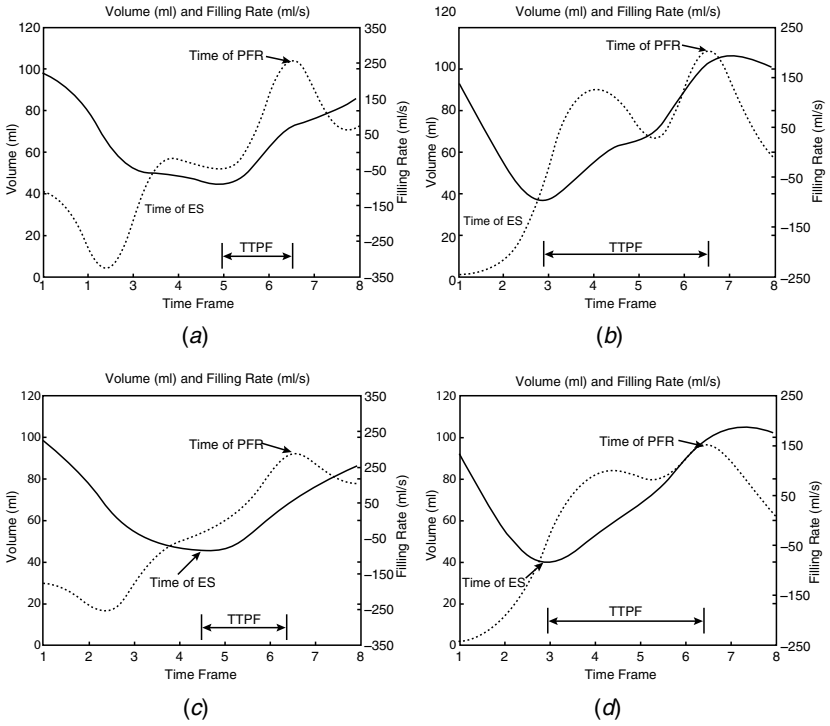


Figure 19. Volume and filling rate curves with different interpolation techniques. (a–b) Top row shows restrictive interpolation results and (c–d) bottom row displays smooth curves. **Left**: both plots (a–c) were obtained with data from the same patient. **Right**: both plots (b–d) were obtained with data from the same patient. Notice the difference in time of ES and PFR, length of TTPF (arrow), amplitude, and shape.

5. DISCUSSION

During the initial development of a new method, visual inspection of the epi- and endocardial boundaries produced by the algorithm are used to assess its correctness. For further improvement on the methodology, quantitative results against a gold standard or ground truth are needed to evaluate its accuracy. Commonly used gold standards for gated SPECT are cine MRI and digital phantoms. Unfortunately, the use of a gold standard was not possible during the development of this work, but will be incorporated into future experiments. Apart from a merely intuitive visual inspection, the first approach to numeric evaluation included inter-study repeatability analysis. This allows for evaluating the same subject under the same clinical conditions. One limitation of using two separate datasets to evaluate algorithm accuracy is that added errors could be present in the dataset itself, due to artifacts (see Section 1.4).

Comparisons with the most widespread clinical analysis tool (QGS) were made in order to have a general idea of the expected results for each patient. These comparisons were performed taking into account previously published studies that describe QGS tendencies, along with our own experience during clinical practice. It should be noted that this study did not aim to compare the accuracy of 3DASM and QGS. QGS computations were included only to highlight problems with existing methods as an incentive to develop new methodologies.

As of datasets with extensive signal drops due to perfusion defects, our method provided a correct anatomical interpolation in the apex with a smooth connection between non-infarcted portions of the wall, based on reliable information from other parts of the image and prior statistical knowledge. There is a clear difference in the LV shape recovered with the two algorithms.

We have identified determination of the mitral valve plane as the main cause of LV volume estimation discrepancies. Our shape model was built from a training set of MRI studies, thereby providing high-resolution anatomical constraints to LV shape recovery. Since expert segmentations spanned from the base to the apex, the deformation process itself made the shape deform and grow toward an acceptable position of the basal plane and apex. As demonstrated in Section 1.4, these kinds of anatomical constraints are not present in SPECT datasets. Therefore, the basal plane defined with initialization points is used to provide an upper limit for cavity volume calculation.

6. CONCLUSIONS

A 3D statistical model-based algorithm was used for segmentation of the left ventricle in dynamic perfusion SPECT studies. The developed statistical model can separate geometric (shape) from gray-level (appearance) models. In this way it is possible to train the algorithm on image data sets of different modalities. The high resolution of functional MRI studies allowed for using an anatomically well-described LV shape representation. The method was applied successfully to 60 clinical studies, providing reproducible segmentation results. The method is robust with difficult cases combining dilated cardiomyopathy, extended myocardial infarction, and ischemia. Once the automatic segmentation of every phase is obtained, global and regional functional parameters can be calculated.

7. ACKNOWLEDGMENTS

Our work is supported by grants TIC2002-04495-CO2 and ISCIII G03/185 from the Spanish Ministries of Science and Technology, and Health, respectively. C.T.G. holds a UPF scholarship, S.O. holds a MEyC FPU grant (AP2002-3955), and A.F. holds an MEyC Ramón y Cajal fellowship.

8. REFERENCES

1. Schaefer W, Lipke C, Standke D, Kulh HP, Nowak B, Kaiser HJ, Koch KC, Buell U. 2005. Quantification of left ventricular volumes and ejection fraction from gated ^{99m}Tc-MIBI SPECT: MRI validation and comparison of the Emory Cardiac Tool Box with QGS and 4D-MSPECT. *J Nucl Med* **46**(8):1256–1263.
2. Lum DP, Coel MN. 2003. Comparison of automatic quantification software for the measurement of ventricular volume and ejection fraction in gated myocardial perfusion SPECT. *Nucl Med Commun* **24**(4):259–266.
3. Nakajima K, Higuchi T, Taki J, Kawano M, Tonami N. 2001. Accuracy of ventricular volume and ejection fraction measured by gated myocardial SPECT: comparison of 4 software programs. *J Nucl Med* **42**:1571–1578.
4. Faber TL, Vansant JP, Pettigrew RI, Galt JR, Blais M, Chatzimavroudis G, Cooke CD, Folks RD, Waldrop SM, Gurtler-Krawczynska E, Wittry MD, Garcia E. 2001. Evaluation of left ventricular endocardial volumes and ejection fractions computed from gated perfusion SPECT with magnetic imaging: comparison of two methods. *J Nucl Med* **40**:645–651.
5. Vaduganathan P, He Z, Vick III GW, Mahmarian JJ, Verani MS. 1998. Evaluation of left ventricular wall motion, volumes, and ejection fraction by gated myocardial tomography with technetium 99m-labeled tetrofosmin: a comparison with cine magnetic resonance imaging. *J Nucl Cardiol* **6**:3–10.
6. Lipke CSA, Kuhl HP, Nowak B, Kaiser HJ, Reinartz P, Koch KC, Buell U, Schaefer WM. 2004. Validation of 4D-MSPECT and QGS for quantification of left ventricular volumes and ejection fraction from gated ^{99m}Tc-MIBI SPET: comparison with cardiac magnetic resonance imaging. *Eur J Nucl Med Mol Imaging* **31**(4):482–490.
7. Tadamura E, Kudoh T, Motooka M, Inubushi M, Okada T, Kubo S, Hattori N, Matsuda T, Koshiji T, Nishimura K, Komeda M, Konishi J. 1999. Use of technetium-99m sestamibi ECG-gated single-photon emission tomography for the evaluation of left ventricular function following coronary artery bypass graft: comparison with threedimensional magnetic resonance imaging. *Eur J Nucl Med* **26**:705–712.
8. Thorley PJ, Plein S, Bloomer TN, Ridgway JP, Sivananthan UM. 2003. Comparison of 99mtc tetrofosmin gated SPECT measurements of left ventricular volumes and ejection fraction with MRI over a wide range of values. *Nucl Med Commun* **24**:763–769.
9. Germano G, Kavanagh P, Su H, Mazzati M, Kiat H. 1995. Automatic reorientation of 3-dimensional transaxial myocardial perfusion SPECT images. *J Nucl Med* **36**:1107–1114.
10. Faber T, Cooke C, Folks R, Vansant J, Nichos K, DePuey E, Pettigrew R, Garcia E. 1999. Left ventricular function and perfusion from gated SPECT perfusion images: an intergrated method. *J Nucl Med* **40**:650–659.
11. Akincioglu C, Berman DS, Nishina H, Kavanagh PB, Slomka PJ, Abidov A, Hayes S, Friedman JD, Germano G. 2005. Assessment of diastolic function using 16-frame ^{99m}Tc-Sestamibi gated myocardial perfusion SPECT: normal values. *J Nucl Med* **46**:1102–1108.
12. Boyer J, Thanigaraj S, Schechtman K, Perez J. 2004. Prevalence of ventricular diastolic dysfunction in asymtomatic, normotensive patients with diabetes mellitus. *Am J Cardiol* **93**:870–875.
13. Yuda S, Fang Z. 2003. Association of severe coronary stenosis with subclinical left ventricular dysfunction in the absence of infarction. *J Am Soc Echocardiogr* **16**:1163–1170.
14. Yamada H, Goh PP, Sun JP, Odabashian J, Garcia MJ, Thomas JD, Klein AL. 2002. Prevalence of left ventricular diastolic dysfunction by doppler echocardiography: clinical application of the canadian consensus guidelines. *J Am Soc Echocardiogr* **15**:1238–1244.

15. Bayata S, Susam I, Pinar A, Dinckal M, Postaci N, Yesil M. 2000. New doppler echocardiographic applications for the evaluation of early alterations in left ventricular diastolic function after coronary angioplasty. *Eur J Echocardiogr* **1**:105–108.

16. Matsumura Y, Elliott P, Virdee M, Sorajja P, Doi Y, McKenna W. 2002. Left ventricular diastolic function assessed using doppler tissue imaging in patients with hypertrophic cardiomyopathy: relation to symptoms and exercise capacity. *Heart* **87**(4):247–251.

17. Galderisi M, Cicala S, Caso P, De Simone L, D'Errico A, Petrocelli A, de Divitiis O. 2002. Coronary flow reserve and myocardial diastolic dysfunction in arterial hypertension. *Am J Cardiol* **90**:860–864.

18. Germano G. 2001. Technical aspects of myocardial SPECT imaging. *J Nucl Med* **42**:1499–1507.

19. Germano G, Berman D. 1999. Quantitative gated perfusion SPECT. In *Clinical gated cardiac SPECT*, pp. 115–146. Ed G Germano, D. Berman. Armonk, NY: Futura Publishing.

20. Cauvin J, Boire J, Bonny J, Zanca M, Veyre A. 1992. Automatic detection of the left ventricular myocardium long axis and center in thallium-201 single photon emission computed tomography. *Eur J Nucl Med* **19(21)**:1032–1037.

21. Germano G, Kiat H, Kavanagh B, Moriel M, Mazzanti M, Su H, Van Train KF, Berman D. 1995. Automatic quantification of ejection fraction from gated myocardial perfusion SPECT. *J Nucl Med* **36**: 2138–2147.

22. Lomsky M, El-Ali H, Astrom K, Ljungberg M, Richter J, Johansson L, Edenbrandt L. 2005. A new automated method for analysis of gated- SPECT images based on a three-dimensional heart shaped model. *Clin Physiol Funct Imaging* **25**(4):234–240.

23. Frangi AF, Niessen WJ, Viergever MA. 2000. Three-dimensional modeling for functional analysis of cardiac images: a review. *IEEE Trans Med Imaging* **20**(1):2–25.

24. Cootes TF, Taylor CJ, Cooper DH, Graham J. 1995. Active shape models — their training and application. *Comput Vision Image Understand* **61**(1):38–59.

25. Kelemen A, Szekely G, Guerig G. 1999. Elastic model-based segmentation of 3D neuroradiological data sets. *IEEE Trans Med Imaging* **18**:828–839.

26. McInerney T, Terzopoulos D. 1996. Deformable models in medical image analysis: a survey. *Med Image Anal* **1**(2):91–108.

27. Montagnat J, Delingette H. 2005. 4D deformable models with temporal constraints: application to 4D cardiac image segmentation. *Med Image Anal* **9**(1):87–100.

28. Cootes TF, Edwards GJ, Taylor CJ. 1998. Active appearance models. *Proc Eur Conf Comput Vision* **2**:484–498.

29. Ordas S, Boisrobert L, Bossa M, Laucelli M, Huguet M, Olmos S, Frangi AF. 2004. Grid-enabled automatic construction of a two-chamber cardiac PDM from a large database of dynamic 3D shapes. In *Proceedings of the 2004 IEEE international symposium on biomedical imaging*, pp. 416–419. Washington, DC: IEEE.

30. Mitchell SC, Bosch JG, Lelieveldt BPF, van der Geest RJ, Reiber JHC, Sonka M. 2002. 3D active appearance models: segmentation of cardiac MR and ultrasound images. *IEEE Trans Med Imaging* **21**(9):1167–1179.

31. Stegmann MB. 2004. *Generative interpretation of medical images*. Phd dissertation, Informatics and Mathematical Modelling, Technical University of Denmark, Lyngby.

32. van Assen HC, Danilouchkine MG, Frangi AF, Ordas S, Westenberg JJM, Reiber JHC, Lelieveldt BPF. 2005. SPASM: segmentation of sparse and arbitrarily oriented cardiac MRI data using a 3D-ASM. In *Lecture notes in computer science*, vol. 3504: 33–43. Ed AF Frangi, P Radeva, A Santos, M Hernandez. New York: Springer.

33. Bardinet E, Cohen LD, Ayache N. 1995. Superquadrics and free-form deformations: A global model to fit and track 3D medical data. In *Lecture notes in computer science*, Vol. 905, pp. 319–326. Ed N Ayache. New York: Springer.

34. Montagnat J, Delingette H. Space and time shape constrained deformable surfaces for 4D medical image segmentation. 2000. In *Lecture notes in computer science*, Vol. 1935, pp. 196–205. Ed SL Delp, AM Digioia, B Jarmaz. New York: Springer.

35. Frangi AF, Rueckert D, Schnabel JA, Niessen WJ. 2002. Automatic construction of multiple-object three-dimensional statistical shape models: application to cardiac modeling. *IEEE Trans Med Imaging* **21**(9):1151–1166.

36. American Heart Association. 1998. *American Heart Association 1999 heart and stroke statistical update*. http://www.americanheart.org, Dallas, Texas.

37. Behiels G, Maes F, Vandermeulen D, Suetens P. 2002. Evaluation of image features and search strategies for segmentation of bone structures in radiographs using active shape models. *Med Image Anal* **6**(1):47–62.

38. van Assen HC, Danilouchkine MG, Dirksen MS, Rieber JHC, Lelieveldt BPF. 2006. A 3D-ASM driven by fuzzy inference: application to cardiac CT and MR. *Med Image Anal*. In press.

39. Bland JM, Altman DG. 1986. Statistical methods for assessing agreement between two methods of clinical measurement. *Lancet* **8476**:307–310.

7

LEVEL SET FORMULATION FOR DUAL SNAKE MODELS

Gilson A. Giraldi, Paulo S.S. Rodrigues,
Rodrigo L.S. Silva, and Antonio L. Apolinário Jr.

National Laboratory for Scientific Computing
Petropólis, Brazil

Jasjit S. Suri

Biomedical Research Institute, Pocatello, Idaho, USA

Dual snake models are powerful techniques for boundary extraction and segmentation of 2D images. In these methods one contour contracts from outside the target and another one expands from inside as a balanced technique with the ability to reject local minima. Such approaches have been proposed in the context of parametric snakes and extended for topologically adaptable snake models through the Dual-T-Snakes. In this chapter we present an implicit formulation for dual snakes based on the level set approach. The level set method consists of embedding the snake as the zero level set of a higher-dimensional function and to solve the corresponding equation of motion. The key idea of our work is to view the inner/outer contours as a level set of a suitable embedding function. The mathematical background of the method is explained and its utility for segmentation of cell images discussed in the experimental results. Theoretical aspects are considered and comparisons with parametric dual models presented.

1. INTRODUCTION

Deformable models, which includes the popular *snake models* [1] and deformable surfaces [2, 3], are well-known techniques for boundary extraction and

Address all correspondence to: Gilson A. Giraldi, Laboratório Nacional de Computação Científica, Av. Getulio Vargas, 333, Quitandinha, Petropólis, Brazil, CEP: 25651-075. Phone: +55 24 2233-6088, Fax: +55 24 2231-5595. gilson@lncc.br.

tracking in 2D/3D images. Basically, these models can be classified in three categories: parametric, geodesic snakes, and implicit models. The relationships between these models have been demonstrated in several works in the literature [4, 5].

Parametric deformable models consist of a curve (or surface) which can dynamically conform to object shapes in response to internal (elastic) forces and external forces (image and constraint ones) [6]. Snake models, also called active contour models, are 2D deformable models proposed by Kass at al. [1] that have been successfully applied in a variety of problems in computer vision and image analysis [6]. Their mathematical formulation makes it easier to integrate image data, the initial estimate, the desired contour properties, and knowledge-based constraints into a single extraction process [6].

For geodesic snakes, the key idea is to construct the evolution of a contour as a geodesic computation. A special metric is proposed (based on the gradient of the image field) to allow the state of the minimal energy to correspond to the desired boundary. This approach allows one to address the parameterization dependence of parametric snake models and can be extended to three dimensions through the theory of minimal surfaces [7, 5].

Implicit models, such as the formulation of the *level set* used in [8], consist of embedding the snake as the zero level set of a higher-dimensional function and solving the corresponding equation of motion. Such methodologies are best suited for recovery of objects with unknown topologies, which is a limitation for most parametric models. In fact, despite the mentioned capabilities, parametric models in general cannot deal with topological changes. Among the approaches to deal with the topological limitations of a traditional snake model, T-Snakes has the advantage of being a general one [9].

In addition, parametric models are too sensitive to their initial conditions due to nonconvexity problems (see [10] and the references therein). To address this limitation, some authors have proposed multiscale techniques [11], dynamic programs (DPs) [12], dual methods [13], as well as a two-stage approach [14, 15]: (1) the region of interest is reduced; (2) a global minimization technique is used to find the object boundaries.

In [15–17] we proposed to address stage (1) by using the Dual-T-Snakes method. Dual-T-Snakes is a parametric snake model. The result of this method is two contours, close to the object boundary, which bound the search space. Hence, a DP algorithm [12, 18] can be used more efficiently.

In the present we review dual snakes models and present an implicit formulation for dual active contour models based on the level set approach. The mathematical background of the Dual-Level-Set (DLS) method is explained. Theoretical aspects are discussed and comparisons with parametric dual models are presented.

In Section 2 we review dual snakes approaches. We then describe the T-Snakes (Section 3) and the Dual-T-Snakes (Section 4) frameworks. In Section 5 we discuss the level set method. Section 6 presents the implicit formulation for dual

snakes. In Section 7 we describe a segmentation framework that can be used when combining dual snakes and search-based snake models. The experimental results are presented in Section 8. We offer some discussion in Section 9 and compare the presented method with related ones. In section 10 we present our conclusions. Finally, Appendix 1 focuses on theoretical aspects of the mathematical background of the level set method.

2. BACKGROUND REVIEW

Dual active contour models have been applied for cell image segmentation [17, 19] and feature extraction [18]. Although there are few examples of such approaches [17, 19, 20], their capability to reject local minima shall be better explored for segmentation and boundary extraction purposes.

The basic idea of the dual active contour models (*Dual ACM*) is to reject local minima by using two contours: one that contracts from outside the target and one that expands from inside. Such a feature makes it possible to reduce sensitivity to initialization by enabling a comparison between the energy of the two contours, which is used to reject local minima.

This methodology was first proposed in [20]. To obtain the conventional continuity and smoothness constraints, but remove the unwanted contraction force, a known problem in traditional snake models [21, 10], a scale-invariant internal energy function (*shape model*) was developed. In [20] a snake is considered as a particle system, $v_i = (x_i, y_i)$, $i = 0, ..., N-1$, whose particles are linked by internal constraints. The shape model is accomplished by the following internal energy:

$$E_{\text{int}} = \frac{1}{2} \sum_{i=0}^{N-1} \left(\frac{\|e_i\|}{h} \right)^2, \tag{1}$$

where

$$e_i = \frac{1}{2} \left(v_{i-1} + v_{i+1} \right) - v_i + \frac{1}{2} \theta_i R \left(v_{i-1} - v_{i+1} \right), \tag{2}$$

h is the average space step, R is a 90^0 rotation matrix, and θ_i is related to the internal angle φ_i in the vertex v_i by

$$\theta_i = \cot \left(\frac{\varphi_i}{2} \right). \tag{3}$$

It is clear that E_{int} has a global minimum when $e_i = 0$, $i = 0, 1, ..., N-1$. From (2)–(3) it can be shown that this happens when

$$\varphi_i = \pi \left(N - 2 \right) / 2N, i = 0, 1, ..., N-1, \tag{4}$$

which are the internal angles of a regular polygon with vertices given by the points v_i [20]. The energy (1) can also be shown to be rotation, translation, and scale

invariant [20]. As usual in snake models [2, 6], the external energy is defined by

$$E_{\text{ext}}\left(v_i\right) = -\left\|\nabla I\left(v_i\right)\right\|^2, \tag{5}$$

The total energy of the model is given by

$$E = \sum_i \left(\frac{\lambda}{N} E_{\text{int}}\left(v_i\right) + \left(1-\lambda\right) E_{\text{ext}}\left(v_i\right)\right), \tag{6}$$

where λ is a smoothing parameter that lies between 0 and 1 [20].

With the internal energy given by expression (1), the curve is biased toward a regular polygon [20]. Therefore, this fact makes it easier to establish a correspondence (*matching*) between the points of the two contours because the form of the snake during evolution is limited by the energy (1). The methodology takes advantage of this correspondence by proposing the *driving force*:

$$F_{driving} = g\left(t\right) \frac{u_i - v_i^t}{\left\|u_i - v_i^t\right\|}, \tag{7}$$

where v_i^t is the contour being processed at time t, u_i is the contour remaining at rest, and $g(t)$ is the strength of the force. The termination condition adopted in [20] is the following one, based on the low-velocity criterion:

$$\max_i \ \left\|v_i^{t+1} - v_i^t\right\| < \delta, \tag{8}$$

where δ is a termination parameter.

The dual approach consists in making the inner and outer contours evolve according the following algorithm: the contour with the highest energy is selected. If its motion remains below some termination condition, then the driving force (7) is increased until it moves at a rate greater than the chosen threshold δ. When the energy begins to decrease, the added driving force is removed and the contour allowed to come into equilibrium. The procedure is then repeated until both contours have found the same equilibrium position.

The shape model given by expression (1) limits the application of the method for general shapes and topologies. In addition, the need for a match between the two contours places restrictions to extend the technique for 3D. These limitations are addressed by the Dual-T-Snakes [15, 17] approach, developed by some of us, which is described in Section 4. The experience with this method shows that it is very useful for reducing the search space. We therefore proposed in [15] a two-stage segmentation approach: (1) the region of interest is reduced using Dual-T-Snakes; and (2) a global minimization technique, the Viterbi algorithm [17, 18], is used to find the object boundaries.

In fact, the Viterbi algorithm was also used in [14] and sometimes it is called a *non-evolutionary* dual model, in the sense that it is not based on curve evolution [22]. Before describing Dual-T-Snakes, we shall discuss the T-Snakes method, which is its background.

3. T-SNAKES MODEL

The T-Snakes approach is basically composed of four components [9]: (1) a simple CF triangulation of the image domain; (2) projection of each snake over the grid; (3) a binary function called a *characteristic function*, χ, defined on the grid nodes, which distinguishes the interior from the exterior of a snake; and (4) a discrete snake model.

To clarify our ideas, consider the characteristic functions (χ_1 and χ_2) relative to the two contours depicted in Figure 1. The vertices marked are those where $\max\{\chi_1, \chi_2\} = 1$. Note that the overlap of the curves belongs to the triangles in which the characteristic function changes value.

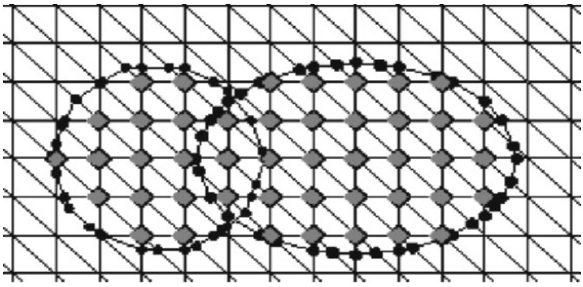


Figure 1. Two snakes colliding with the inside grid nodes and snake point (*snaxels*) marked. Reprinted with permission from Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003. Copyright ©2003, Elsevier.

Thus, from the data obtained with step (2), we can choose a set of N points, $\{v_i = (x_i, y_i)\,,\ \ i = 0, ..., N-1\}$ to be connected to form a closed contour (T-Snake). These points are called *snaxels*.

In [16] we proposed to evolve a T-Snake based on a tensile (smoothing) force (B_i), a normal (balloon-like) force (F_i), and an external (image) force (f_i) [9]. These forces are given respectively by the following expressions:

$$B_i = b_i \left(v_i - \frac{1}{2}\left(v_{i-1} + v_{i+1}\right) \right). \tag{9}$$

$$F_i = k_i \left(\text{sign}_i\right) n_i; \quad f_i = \gamma_i \nabla P, \tag{10}$$

where n_i is the normal at the snaxel v_i and b_i, k_i, γ_i are force scale factors, $P = -\left\|\nabla I\right\|^2$, $\text{sign}_i = 1$ if $I\left(v_i\right) \geq T$ and $\text{sign}_i = 0$ otherwise (T is a threshold for the image I). Region-based statistics can be also used [9].

Hence, we update the T-Snake position according to the following evolution equation:

$$v_i^{(t+\Delta t)} = v_i^t + h_i \left(B_i^t + F_i^t + f_i^t\right), \tag{11}$$

where h_i is an evolution step.

The T-Snakes model also incorporates an *entropy condition*: "once a node is *burnt* (passed over by the snake) it stays burnt" [9].

A termination condition is defined based on the number of deformations steps (*temperature*) during which a triangle is cut by a T-Snake. A T-Snake is considered to have reached its equilibrium state when the temperature of all the snaxels falls below a preset value, called the "freezing point" in the T-Snakes literature [9, 23].

4. DUAL-T-SNAKES ALGORITHM

The key idea behind this method is to explore the T-Snakes framework to propose a *generalized dual active contour model*: one T-Snake contracts and splits from outside targets and another ones expand from inside targets.

To make the outer snake contract and the inner ones expand, we assign an inward normal force to the first and an outward normal force to the others according to expressions (10). Also, to make the T-Snakes evolution interdependent we use the *image energy* and an *affinity* restriction.

We use two different definitions for image energy: one for the outer contour, (E_{outer}), and the other for the set of inner contours enclosed by it, (E_{inner}):

$$E_{\text{outer}} = \sum_{i=0}^{N-1} \left(-\left\|\nabla I\left(v_i\right)\right\|^2\right)/N, \tag{12}$$

$$E_{\text{inner}} = \frac{1}{m}\left(\sum_{k=0}^{m}\left(\sum_{i=0}^{N_k-1}\left(-\left\|\nabla I\left(v_i\right)\right\|^2\right)/N_k\right)\right), \tag{13}$$

where m is the number of inner curves, and N, N_k represent the number of snaxels of the outer and inner (k) snakes, respectively. The energies are normalized in order to be compared. Otherwise, the snake energy would be a decreasing function of the number of snaxels and comparisons would not make sense.

Following the dual approach methodology [20], if $E_{\text{inner}} > E_{\text{outer}}$, an inner curve must be chosen. To accomplish this, we use an *affinity operator*, which estimates the pixels of the image most likely to lie on the boundaries of the objects. Based on this operator, we can assign to a snaxel the likelihood that it is close to a boundary. That likelihood is thresholded to obtain an *affinity function* that assigns

to the snaxel a value of 0 or 1: 0 for snaxels most likely to lie away from the target boundaries and 1 otherwise.

Then, the inner curve with the highest number of snaxels with an affinity function value of null is chosen. If $E_{\text{outer}} > E_{\text{inner}}$, the outer snake is evolved if the corresponding affinity function has null entries.

In addition, the balance between the energy/affinity of the outer and inner snakes allows avoiding local minima. For instance, if a T-Snake has been frozen, we can increase the normal force at the snaxels where the affinity function is zero. The self-intersections that may happen during evolution of a snake when increasing the normal force are naturally resolved by the T-Snakes model. This is the way we can use the added normal force to play the role of the driving force used by Gunn and Nixon (avoiding the matching problem required in [20]).

To evaluate the similarity of two contours, we use the difference between the characteristic function of the outer snake and the characteristic functions of the inner ones (*Characteristic_Diff*). For example, in the case of CF triangulation in Figure 1, we can stop the motion of all snaxels of an inner snake inside a triangle σ if any of its vertices $v \in \sigma$ has the two following properties: (a). All the 6 triangles adjacent to v have a vertex where Characteristic_Diff=0; and (b) one of these triangles is crossed by the outer contour.

The freezing point is used to indicate that a T-Snake has found an equilibrium position. In the following algorithm we call *Dual Snake* a list of T-Snakes where the first one is an outer contour and the other ones are inner contours. The algorithm is summarized as on the following page.

This method can be efficient for images with artifacts and noise. In [17] it was improved by using a multigrid approach and a region-growing method based on the grid. Other improvements can be found in [16]. Once Dual-T-Snakes stops, a global minimization method [15] or a simple greedy technique can be used to find the object boundaries in order to complete the segmentation (see Section 7 below). Next, we develop the basic elements of the level set in order to be able to present the implicit formulation for dual approaches (Section 6).

5. LEVEL SET

In this section we review some of the details of the level set formulation [8]. The main idea of this method is to represent a deformable surface (or curve) as a level set, $\left\{x \in \Re^3 | G(x) = 0\right\}$, of an embedding function,

$$G : \Re^3 \times \Re^+ \rightarrow \Re, \tag{14}$$

such that the deformable surface (also called the *front* in this formulation) at $t = 0$ is given by a surface S:

$$S(t = 0) = \left\{x \in \Re^3 | G(x, t = 0) = 0\right\}. \tag{15}$$

Algorithm 1 Dual-T-Snakes

Put all the dual snakes into a queue.
repeat
  Pop out a dual snake from the queue;
  Use the energies (equations (12) and (13)) and the affinity function to decide the snake to be processed;
  if all snaxels of that snake are frozen **then**
    repeat
      increase the normal force at those with affinity zero
    until the snake energy starts decreasing.
    Remove that added normal force;
    repeat
      Evolve the snake
    until the temperature of all snaxels falls below the freezing point;
    Analyze the *Characteristic_Diff* of the current snake;
    if the snake being processed is close to a snake of the other type (inner/outer) **then**
      remove the dual snake from the queue.
    else
      mount the resulting dual snake(s) and go to the beginning.
    end if
  end if
until the queue is empty

The next step is to find an Eulerian formulation for the front evolution. Following Sethian [8], let us suppose that the front evolves in the normal direction with velocity $\overrightarrow{F}$, which may be a function of the curvature, normal direction, etc.

We need an equation for the evolution of $G(x,t)$, considering that the surface S is the level set given by:

$$S(t) = \left\{ x \in \Re^3 | G(x,t) = 0 \right\}. \tag{16}$$

Let us take a point, $x(t)$, $t \in \Re^+$, of the propagating front S. From its implicit definition given above we have

$$G(x(t), t) = 0. \tag{17}$$

Now, we can use the chain rule to compute the time derivative of this expression:

$$G_t + F|\nabla G| = 0, \tag{18}$$

where $F = \|dx/dt\|$ is called the *speed function*. An initial condition, $G(x, t = 0)$, is required. A straightforward (and expensive) technique to define this function is to compute a signed-distance function as follows:

$$G(x, t = 0) = \pm d, \tag{19}$$

where d is the distance from x to the surface $S(x, t = 0)$ and the sign indicates if the point is interior (-) or exterior (+) to the initial front.

Finite-difference schemes, based on an uniform grid, can be used to solve Eq. (18). The same entropy condition of T-Surfaces (*once a grid node is burnt it stays burnt*) is incorporated in order to drive the model to the desired solution (in fact, T-Surfaces was inspired by the level set model [24]).

In this higher-dimensional formulation, topological changes can be efficiently implemented. Numerical schemes are stable, and the model is general in the sense that the same formulation holds for 2D and 3D, as well as for merge and splits. In addition, the surface geometry is easily computed. For example, the front normal ($\overrightarrow{n}$) and curvature (K) are given, respectively, by:

$$\overrightarrow{n} = \nabla G(x, t), \quad K = \nabla \cdot \left(\frac{\nabla G(x, t)}{\|\nabla G(x, t)\|} \right), \tag{20}$$

where the gradient and divergent ($\nabla \cdot$) are computed with respect to x.

The update of the embedding function through expression (18) can be made cheaper if the *narrow-band* technique is applied. The key idea of this method comes from the observation that the front can be moved by updating the level set function at a small set of points in the neighborhood of the zero set instead of updating it at all the points on the domain (see [8, 25] for details).

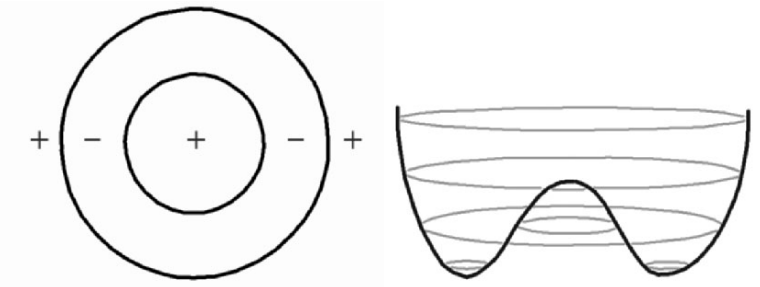
To implement this scheme we need to pre-set a distance Δd to define the narrow band. The front can move inside the narrow band until it collides with the narrow band frontiers. Then, the function G should be reinitialized by treating the current zero set configuration as the initial one.

This method can also be made cheaper by observing that the grid points that do not belong to the narrow band can be treated as sign holders [8], following the same idea of the characteristic function used in Section 3. This observation will be used in Section 6.2. We are now able to present the level set formulation for the dual snake model.

6. DUAL-LEVEL-SET APPROACH

In this section we examine the philosophy of Dual-T-Snakes: one snake contracts and splits from outside the targets and another expands from inside the targets. However, the snake model will be an implicit one.

To clarify ideas, let us consider Figure 2a, which shows two contours bounding the search space, and Figure 2b, which pictures a bi-dimensional surface where the zero level set is the union of the two contours just presented.



(a) (b)

Figure 2. (a) Dual snakes bounding the search space. (b) Initial function where the zero level set is the two contours presented.

If the surface evolves such that the two contours get closer, we can obtain the same behavior of Dual-T-Snakes. That is the key idea of our proposal. In order to accomplish this goal, we must define a suitable speed function and an efficient numerical approach. For simplicity, we consider the one-dimensional version of the problem depicted in Figure 3. In this case, the evolution equation can be written as

$$G_t + \frac{\partial G}{\partial x} F = 0. \tag{21}$$

The main point is to design the speed function F such that $G_t > 0$. Therefore, if we set the sign of F opposite to that of G_x, we attain this goal once

$$G_t = -\frac{\partial G}{\partial x} F. \tag{22}$$

Hence, the desired behavior can be obtained by distribution of the sign of F shown in Figure 3.

However, we should note that $G_x = 0$ for singular points. The values of G therefore remain constant over these points because G_t becomes null. Thus, we should be careful about surface evolution nearby the singular points because anomalies may happen. One possible way to avoid this problem is to stop front evolution before it gets close to this point. Another possibility could be to change the evolution equation in order to allow $G_t \neq 0$ over singular points. Such proposal implies that the isolines may be not preserved, that is, they become a function of time also. Thus,

$$G\left(x\left(t\right), t\right) = y\left(t\right); \tag{23}$$

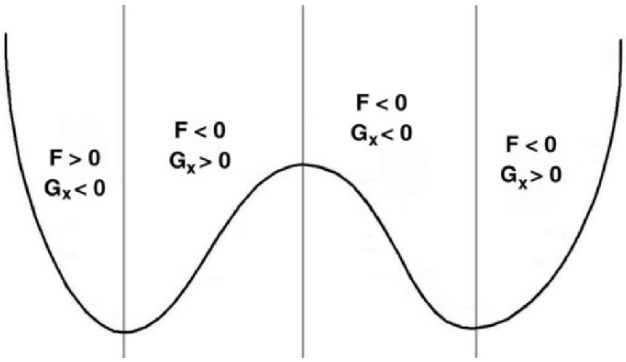


Figure 3. Sign of the speed function.

consequently, by applying the chain rule:

$$G_t + \frac{\partial G}{\partial x}\frac{dx}{dt} = \frac{dy}{dt}. \tag{24}$$

Therefore, we should provide a speed function in the y direction:

$$G_t + \left(\frac{\partial G}{\partial x}, -1\right) \cdot \left(\frac{dx}{dt}, \frac{dy}{dt}\right) = 0. \tag{25}$$

We can write this expression as

$$G_t + F\,|(\nabla G, -1)| = 0,$$

where F is the speed function. For fronts in 3D we get

$$G_t + \frac{\partial G}{\partial x}\frac{dx}{dt} + \frac{\partial G}{\partial y}\frac{dy}{dt} = \frac{dz}{dt}, \tag{26}$$

and, therefore,

$$G_t + \left(\frac{\partial G}{\partial x}, \frac{\partial G}{\partial y}, -1\right) \cdot \left(\frac{dx}{dt}, \frac{dy}{dt}, \frac{dz}{dt}\right) = 0. \tag{27}$$

One way to deal with these models is through *viscous conservation laws*. For example, expression (24) becomes

$$G_t + \frac{\partial G}{\partial x}\frac{dx}{dt} = \varepsilon \frac{\partial^2 G}{\partial x^2}, \tag{28}$$

if dy/dt is replaced by G_{xx}. For 2D we will have

$$G_t + \left(\frac{\partial G}{\partial x}, \frac{\partial G}{\partial y}\right) \cdot \left(\frac{dx}{dt}, \frac{dy}{dt}\right) = \varepsilon \nabla^2 G, \tag{29}$$

where ∇^2 means the Laplacian operator, that is:

$$\nabla^2 G = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}. \tag{30}$$

In our model we will maintain the idea that the front evolves in the normal direction. Thus, expression (29) can be rewritten as

$$G_t + F\,|\nabla G| = \varepsilon \nabla^2 G, \tag{31}$$

following the same development to derive expression (18). Such a model has been studied in the context of front propagation in [8, 25, 26] and can be used to achieve the desired result.

Once our application focus is shape recovery in a image I, we must choose a suitable speed function F as well as a convenient stopping term S to be added to the right-hand side of Eq. (31). Among the possibilities [27], the following ones have been suitable for our Dual-Level-Set:

$$F = \frac{1 + \alpha k}{1 + |\nabla I|^2}, \tag{32}$$

$$S = \beta \nabla P \cdot \nabla G, \tag{33}$$

where β is a scale parameter and P is defined just after expression (10). Therefore, we are going to deal with the following level set model:

$$G_t = \left(\frac{1 + \alpha k}{1 + |\nabla I|^2}\right) |\nabla G| + \varepsilon \nabla^2 G + \beta \nabla P \cdot \nabla G, \tag{34}$$

where $P = -|\nabla I|^2$, as in Eq. (10). The evolution of the fronts is interdependent due to the embedding function. After initialization (see Section 6.2), all the fronts evolve following expression (34). However, once the evolution is halted, we must evaluate the similarity between the two contours and apply a *driving velocity* instead of the driving force of Section 4. We next develop these elements of the Dual-Level-Set method.

6.1. Numerical Methods

Expression (34) can be written in a general form as

$$G_t + H\left(G_x, G_y\right) = \varepsilon \nabla^2 G, \tag{35}$$

where H may be a non-convex *Hamiltonian*. Therefore, following [25], we can use a first-order numerical scheme given by

$$G_{i,j}^{n+1} = G_{i,j}^{n} - \Delta t \left[H\left(\frac{D_{i,j}^{-x}G + D_{i,j}^{+x}G}{2}, \frac{D_{i,j}^{-y}G + D_{i,j}^{+y}G}{2} \right) \right] - \tag{36}$$

$$\varepsilon \frac{\Delta t}{2} \left(\frac{D_{i,j}^{+x}G - D_{i,j}^{-x}G}{2} + \frac{D_{i,j}^{+y}G - D_{i,j}^{-y}G}{2} \right), \tag{37}$$

where $D_{i,j}^{-x}$ and $D_{i,j}^{+x}$ are first-order finite-difference operators defined by:

$$D^{-x}G = \frac{G(x,y) - G(x - \Delta x, y)}{\Delta x}, \tag{38}$$

$$D^{+x}G = \frac{G(x + \Delta x, y) - G(x,y)}{\Delta x}, \tag{39}$$

following the same idea for $D_{i,j}^{-y}$ and $D_{i,j}^{+y}$. This scheme can be obviously extended to 3D. Besides, we constrain the change of the discrete field $G_{i,j}$ as follows:

$$\left| G_{i,j}^{n+1} - G_{i,j}^{n} \right| < \delta, \tag{40}$$

where δ is a parameter (set at 1.0 in the tests below) for controlling the rate of change of G. With such a procedure we get a more stable front evolution and the choice of parameter values for the numerical scheme becomes easier. In Appendix 1 we offer an additional discussion about numerical methods for differential equations like expression (35).

6.2. Initialization

In shape recovery problems, discontinuities in image intensity are significant and important. In fact, such discontinuities of an image I can be modeled as step functions like in [28]:

$$s(x,y) = \sum_{i=1}^{n} a_i S_i(x,y), \tag{41}$$

where S_i is a step function defined on a closed subset Ω_i of the image domain, which means

$$\begin{aligned} S_i(x,y) &= 1 \quad if \quad (x,y) \in \Omega_i, \\ S_i(x,y) &= 0 \quad otherwise. \end{aligned} \tag{42}$$

Therefore, an image I is not a differentiable function in the usual sense. The usual way to address this problem is to work with a coarser-scale version of the image obtained by convolving this signal with a Gaussian kernel:

$$I_g(x,y) = K_\sigma(x,y) * I(x,y), \tag{43}$$

where $K_\sigma(x, y)$ is a Gaussian function with standard deviation σ. Therefore, we must replace the function I by I_g in expression (34).

This point is interesting for our context due to initialization of the Dual-Level-Set approach. A very simple way to do this would be through step functions like expression (41). For instance, Figure 4 shows such an example based on the sign distribution of Figure 2a, in which we decompose the image domain in three closed subsets, Ω_1, Ω_2, and Ω_3. The step function would be given by expression (41) with $a_1 = 1$, $a_2 = -1$, and $a_3 = 1$.

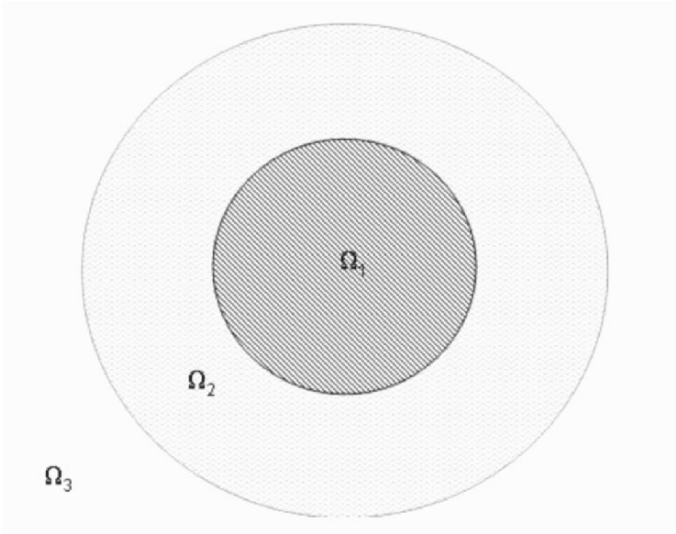


Figure 4. Support of a step function given by Eq. (41) with $a_1 = 1$, $a_2 = -1$, and $a_3 = 1$.

Such a proposal is computationally efficient but cannot be accepted due to discontinuities of the function s. From a functional analysis viewpoint, we should use a differentiable function to initialize the method. We can address this problem following the same idea behind expression (43), that is, we can define a step function through expression (41), and then take a convoluted version of it obtained by replacing image I by function $s(x, y)$ in expression (43). Such a smoothed function will be suitable for our purposes. If we add boundary conditions to Eq. (34), we get a parabolic problem that may have complex behaviors from the numerical viewpoint, as we will show in Appendix 1.

6.3. The Narrow Band Method and the Termination Condition

As usual in level set approaches, we are using the narrow-band method, that is, only the values of G within a tube placed around the front (Figure 5) are updated. This simplified approach can drop the complexity of the algorithm from $O\left(N^3\right)$ in three dimensions, to $O\left(mN^2\right)$, where m is the number of cells in the width of the narrow band [25]. When the front moves near to the edge of the tube boundary, the computation is stopped and a new tube is built with the zero level set at the center.

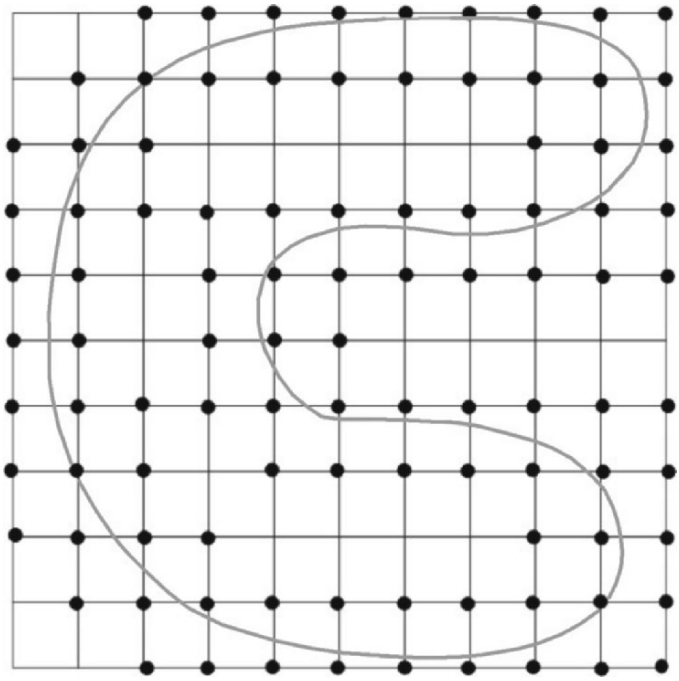
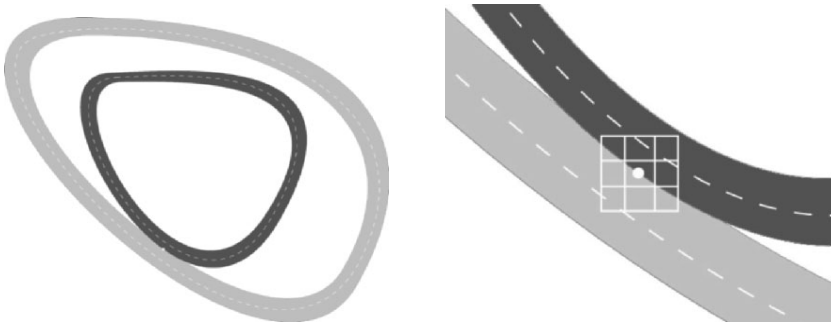


Figure 5. Narrow band around a propagating front.

To construct the narrow band, we define two polygons. Each polygon is associated with an element of the curve that approximates the zero level set of the previous iteration. Each edge and vertex defines two polygons, for the inside and outside parts of the narrow band. Edge polygons are quadrilaterals, built by a shift of the edge in the direction of the normal vector, by a displacement defined by the narrow bandwidth. Similarly, to each vertex we construct triangles, using the normal vectors of each adjacent edge and with height defined also by the narrow bandwidth. This approach is similar to the procedure used by Maunch [29] and

Sigg and Peikert [30] to calculate the Euclidean distance function using a scan process. Our scan process is executed by the GPU of the video card, during the rasterization pipeline phase. The result of the rendering process is returned to the application as a texture, where the cells in the narrow band are associated with the pixels generated by each polygon rasterization.

For our dual approach, the narrow band is attractive not only for computational aspects but also because it allows an efficient way to evaluate similarity between two contours. In fact, instead of using the criterion of Section 4, we take the procedure depicted in Figure 6. First, the intersection point is computed (Figure 6a); we then take a neighborhood of this point (Figure 6b) and stop to update the function G in all the grid points inside it, or we can set to zero the velocity function for these points. We say that those grid points are *frozen* ones.



(a) (b)

Figure 6. (a) Narrow bands touching each other. (b) Neighborhood to define similarity between fronts.

The number of non-frozen grid points inside a narrow band offers a criterion that indicates that the Dual-Level-Set has found an equilibrium position because, if this number does not change for some interactions, we can be sure that the zero level set remains unchanged. So, we must stop the evolution.

6.4. Driving Velocity

Once the fronts stop moving, we must decide at which grid points to add a *driving velocity*. It is an extra velocity term whose goal is the same as that of the driving force in Section 4, that is, to keep fronts moving again. Therefore, we

employ a less sensitive model of the initial position of the fronts. To accomplish this task we can add an extra term to Eq. (34), which has a general form:

$$V_{driv} = \lambda_1 F_p \left|\nabla G\right| + \lambda_2 V_{\text{shape}} + \lambda_3 \mathbf{F}_{\text{adv}} \cdot \nabla G, \tag{44}$$

the terms of which will be explained next. We must be careful when choosing the grid points to apply this expression. As in the case of Dual-T-Snakes, the fronts may be near the boundary somewhere, but far from the target at another point. We should automatically realize this fact when the fronts stop moving. To do so, we can use the *affinity operator* explained in Section 4. Based on this operator, we can define an *affinity function* that assigns to a grid point inside the narrow band a value of 0 or 1: 0 for the grid points most likely to lie away from the target boundaries and 1 otherwise. Like in Dual-T-Snakes, such affinity operator can be defined through fuzzy segmentation methods [31–33], image transforms [34, 35], region statistics [27], etc. For example, we can turn on the V_p term wherever the affinity function is 0 by using Suri's proposal [27], given by

$$F_p = 1 - 2u\left(x, y\right), \tag{45}$$

where u is the fuzzy membership function that has values within the range $[0, 1]$.

The $\mathbf{F}_{\text{adv}}$ vector field can be implemented through an extra force field. In this work, we propose Gradient Vector Flow (GVF) to define this term. GVF is a vector diffusion method that was introduced in [36] and can be defined through the following diffusion reaction equation [37]:

$$\begin{aligned} \frac{\partial \mathbf{v}}{\partial t} &= \nabla \cdot \left(g \nabla \mathbf{v}\right) + h\left(\nabla f - \mathbf{v}\right), \\ \mathbf{v}\left(x, 0\right) &= \nabla f, \end{aligned} \tag{46}$$

where f is a function of the image gradient (e.g., $P = -\left\|\nabla I\right\|^2$), and $g\left(x\right), h\left(x\right)$ are nonnegative functions defined on the image domain. The field obtained by solving the above equation is a smooth version of the original one that tends to be extended very far from the object boundaries. When used as an external force for deformable models, it makes the methods less sensitive to initialization [36] and improves their convergence to the object boundaries. However, it is limited in the presence of noise and artifacts. Despite this problem, the result can be worthwhile near the boundaries of interest. We can therefore use the usual external field until the evolution stops. The application of GVF as an extra velocity can then push the fronts toward the boundaries. In this work, we simply set $\mathbf{F}_{\text{adv}} = \mathbf{v}$, with $\mathbf{v}$ given by the GVF solution (Eq. (46)). In [38] we found another possibility for combining GVF and level set models.

We can also implement the driving velocity based on global shape information. In this case, there are training data that offer the prior shape information [39]. First, we must find the shape and pose parameters, represented by vectors α and β,

respectively. Such parameters can be found by employing Principal Component Analysis (PCA). To accomplish this task, Leventon [39] took n curves and, for each, defined an embedding function ϕ_i, $i = 1, .., n$. Then, following the standard PCA procedure (see [40], p. 163), we compute the mean shape and the mean offset maps, given respectively by

$$\mu = \frac{1}{n}\sum_{i=1}^{n}\phi_i, \tag{47}$$

$$\widehat{\phi}_i = \phi_i - \mu, \quad i = 1, .., n. \tag{48}$$

The mean shape gives the pose parameters, that is, $\beta = \mu$. Next, we write each matrix $\widehat{\phi}_i$ in the form of a column vector of dimension N, and form a matrix M of dimension $N \times n$, as well as the covariance matrix given by

$$R = \frac{1}{n}MM^T. \tag{49}$$

We then apply singular-value decomposition (SVD) to represent the covariance matrix R as

$$R = U\sum U^T,$$

where U is a matrix whose column vectors represent the set of orthogonal modes of shape variation arranged according to the decreasing order of its eigenvalues, and $\sum$ is a diagonal matrix of corresponding singular values. Let U_k be the matrix composed by the k first columns of U (the k principal components). Thus, given a field u, we can compute the shape parameters α by [39]

$$\alpha = U_k^T\left(u - \mu\right). \tag{50}$$

Using the Gaussian distribution, the prior shape model can be computed as

$$P\left(\alpha\right) = \frac{1}{\sqrt{\left(2\pi\right)\left|\sum_k\right|}}\exp\left(\frac{1}{2}\alpha^T\Sigma_k^{-1}\alpha\right), \tag{51}$$

where $\sum_k$contains the first k rows and columns of Σ.

This expression is used in [39] to compute the optimized G^*, which is defined as

$$G^*\left(t\right) = \arg\max P\left(G^*|G\left(t\right), \nabla I\right), \tag{52}$$

where G is the embedding function at time t and ∇I is the image gradient. With G^* we can define V_{shape} as

$$V_{\text{shape}} = G^*\left(t\right) - G\left(t\right). \tag{53}$$

7. SEGMENTATION FRAMEWORK

As long as the two fronts reach an equilibrium state, we can assure that the boundary of the object we are trying to identify is located inside the two half narrow bands associated with the inside and outside fronts. Therefore, the Dual-Level-Set method can be combined with a search-based technique (a dynamic program or greedy technique), resulting in a boundary extraction procedure composed by four steps [15, 16, 41]: (a) the user initializes the inner/outer fronts; (b) computation of the affinity operator and affinity function; (c) application of the Dual-Level-Set algorithm; and (d) finding the final boundaries using a search-based approach.

The same methodology was applied for the Dual-T-Snakes method [17]. As the boundary is enclosed by fronts obtained at the end of the Dual-Level-Set method (see Figure 7), the Viterbi algorithm [13, 18] may be suitable. In this algorithm the search space is constructed by discretizing each curve into N points and establishing a match between them. Each pair of points is then connected by a segment that is subdivided into M points. This process provides a *discrete search space* with NM points (Figure 7).

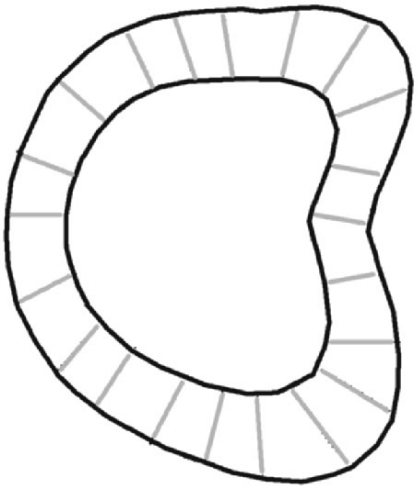


Figure 7. Search space obtained through a matching between inner and outer snakes.

The target boundary is then determined by minimizing the following energy functional [18]:

$$E_{snake} = \sum_{i=0}^{N-1} E_i; \tag{54}$$

$$E_i = \alpha E_{\text{int}}\left(v_{i-1}, v_i, v_{i+1}\right) + \beta E_{\text{ext}}\left(v_i\right) + \lambda E_{\text{line}}\left(v_i\right), \tag{55}$$

with E_{int}, E_{ext}, and E_{line} defined as follows:

$$E_{\text{int}}\left(v_{i-1}, v_i, v_{i+1}\right) = \left(\frac{v_{i+1} - 2v_i + v_{i-1}}{\left\|v_{i+1} - v_{i-1}\right\|}\right)^2, \tag{56}$$

$$E_{\text{ext}}\left(v_i\right) = -\left\|\nabla I\left(v_i\right)\right\|^2; \quad E_{\text{line}}\left(v_i\right) = \pm I\left(v_i\right), \tag{57}$$

where α, β and λ are parameters to be chosen in advance. The term E_{line} will attract the solution to light (signal $-$) or dark lines (signal $+$). The term E_{ext} will attract the solution to points with a high gradient, and E_{int} is a smoothing term. The corresponding recurrence relation is the classical one [12]:

$$S_i\left(v_{i+1}, v_i\right) = \min_{v_{i-1}} \left\{S_{i-1}\left(v_i, v_{i-1}\right) + E_{i-1}\right\}. \tag{58}$$

where $S_i\left(v_{i+1}, v_i\right)$ is the *energy of the optimum path connecting* v_i *to* v_0.

If the two fronts are too close to each other, there is another search-based method that can be efficient and less expensive then the Viterby algorithm. We first compute a curve located in between the two fronts. This curve could be defined as the skeleton of the Euclidian distance field, induced by the positive narrow band of the inside front and the negative narrow band of the outside front. The skeleton is defined as the set of points where the gradient of the distance field is null — in other words, the points located at the same distance from the external and internal fronts.

In order to avoid the cost of building the distance field, we use a simpler process that can generate an approximation of some points on the skeleton curve. We establish a matching between the two fronts, as in Figure 7, but take only the midpoint of each segment.

The obtained curve can be used to initialize a snake model based on a greedy algorithm that works as follows. For each snaxel v_i we take a 3×3 neighborhood V, and for each pixel $p \in V$ we compute

$$E\left(p\right) = \alpha E_{\text{int}}\left(v_{i-1}, p, v_{i+1}\right) + \beta E_{\text{ext}}\left(p\right) + \lambda E_{\text{line}}\left(p\right), \tag{59}$$

We then solve the following problem:

$$p_o = \arg\min\left\{E\left(p\right); \;\; p \in V\right\}. \tag{60}$$

If $E\left(p_o\right) < E\left(v_i\right)$, then $v_i \leftarrow p$. The algorithm can then be summarized as shown below.

The snake position is updated following this procedure for all snaxels. A termination condition is achieved when snaxels no longer move (equilibrium position). Greedy algorithms have been used in the context of snake models in order to improve computational efficiency [41, 42]. The proposed algorithm is simple to implement and efficient in extracting the desired boundary when the dual model stops.

Algorithm 2 Greedy Snake Procedure

for all snaxel v_i **do**
  take a 3×3 neighborhood V
  $p_o = \arg\min \{E(p); \quad p \in V;$
  if $E(p_o) < E(v_i)$ **then**
    $v_i \leftarrow p_o$.
  end if
end for

8. DUAL-LEVEL-SET RESULTS

In this section we present results, obtained through the Dual-Level-Set method, for 2D images. At initialization, the user sets the initial fronts. The setting of parameters is through experimentation. The whole Dual-Level-Set algorithm can be summarized as follows: (1) Initialization through step functions. (2) Evolution until fronts stop. (3) Evaluation of similarity. If frozen, stop. (4) Add V_{drive} for some time steps. (5) After that, turn off V_{drive}. Go to step 2.

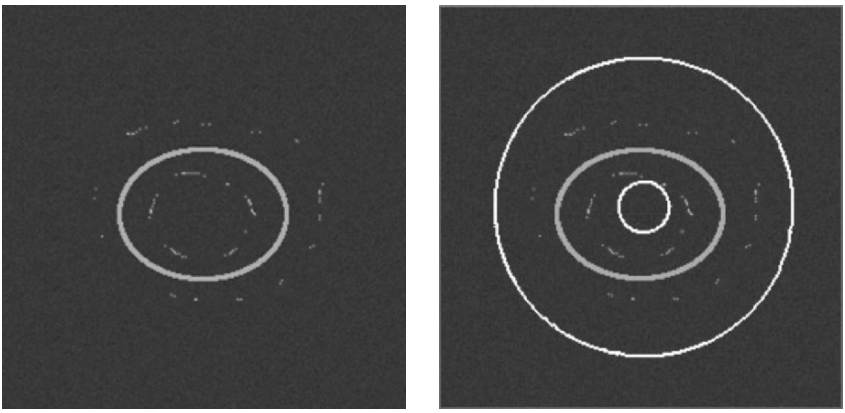
The embedding function is initialized by a step function following the discussion of Section 6.2. When the dual method stops, we apply the greed-based approach to obtain the final result (Section 7). In the following presentation, we call each update of the embedding function G an *interaction step*, according to expression (37). The termination condition used is based on the number of non-frozen points inside the narrow bands. If this number is low (30 in the following tests) and it does not change for some interactions, we consider that the zero level set remains unchanged and stop the evolution. The parameter δ in expression (40) was set to 1.0 in all tests.

In addition, active contour models are in general applied after some kind of image processing methods for noise reduction and edge detection [11]. We demonstrate below that bandpass and low-pass filtering can be useful in the proposed segmentation framework.

8.1. Synthetic Images

In the following example, the input image is depicted in Figure 8a, which shows an ellipse embedded in Gaussian noise. Figure 8b shows the initial fronts. The Dual-Level-Set parameters are $\alpha = 0.1$, $\varepsilon = 0.5$, $\beta = 0.5$, $T = 60$, and $\Delta t = 0.05$.

Figure 9a shows an instant when the two fronts stop moving according to the above condition. We then apply the affinity operator based on the threshold that characterizes the background ($T = 60$). Therefore, if p is a non-frozen grid point, inside a narrow band, which satisfies $I(p) < 60$, then we simply set $\beta = 0$ for a



(a) (b)

Figure 8. (a) Original image. (b) Initial fronts.

while (10 iterations in this case). Therefore, fronts obtain velocity and Figure 9b shows their new positions 10 iterations later. The same happens in Figures 9c–d. This example emphasizes an ability to avoid local minima with the Dual-Level-Set method. The total number of interaction steps was 1618 in this example, and the image resolution is 256×256 pixels.

The result using the Dual-Level-Set method is shown in Figure 9d. We can see that the method achieved the desired target, which means that it was able to obtain two contours close to the object boundary that bound the search space. An important aspect to be considered is that the width of the search space remains almost the same over its extension. It is interesting because it makes computation of the search space with the Viterbi algorithm much easier. However, the greedy model can be also efficient because we are very close to the target. This will be our choice, which is presented next. Figure 10 shows the final result depicted by the red curve. We can observe that the boundary extraction framework achieves the desired goal. Initialization of the greedy snake model was performed following the presentation in Section 7.

8.2. Blood Cell

The following example is useful to understand how filtering methods fits well with our Dual-Level-Set technique. Figure 11a shows a blood cell with a resolution of 300×300 pixels, obtained using an electronic microscope. When a passband filter is applied, we get an edge map that resembles a ribbon whose thickness depends on the kernel size of the used filter. That is an ideal situation for applying the segmentation framework of Section 7 because the Dual-Level-Set can extract

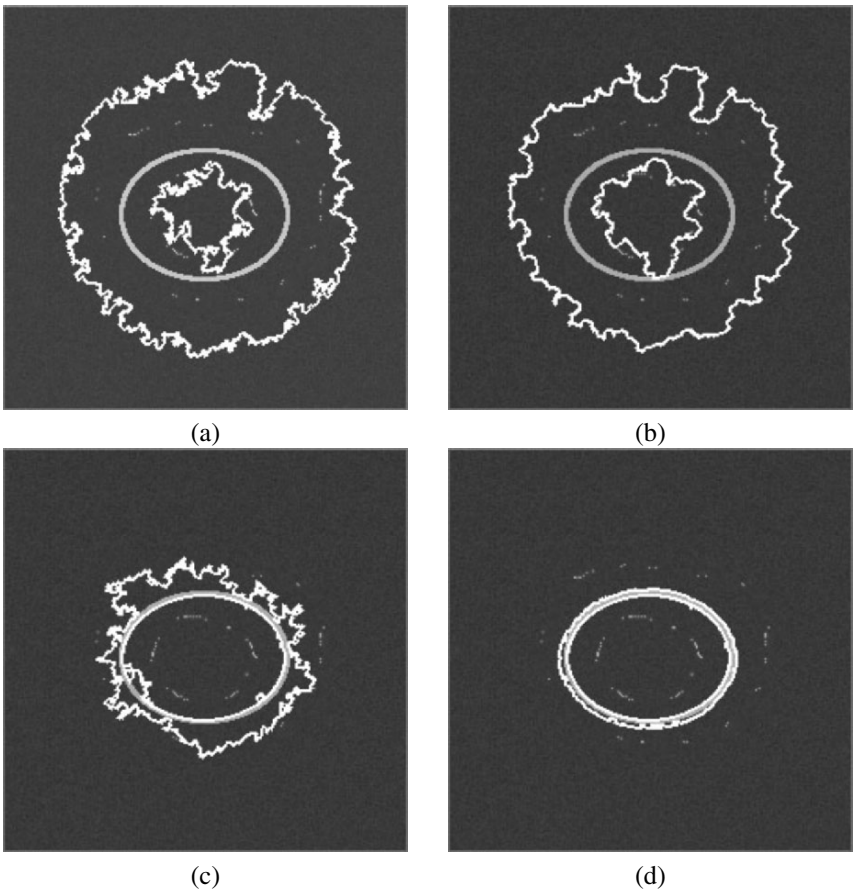


Figure 9. (a) Fronts stop moving at interaction 453 due to the low-velocity criterion. (b) Front position in interaction 463 when the stopping term is turned on. (c) Configuration at interaction 1347. (d) Final front positions.

the ribbon, and, a search based model, like the Viterby or the greedy one, can then be applied to yield the final result. In Figure 11b we depict the bandpass version of the original image as well as the initial fronts for the dual method. In this case, we apply the affinity operator based on the threshold that characterizes the boundaries ($T = 150$): if p is a non-frozen grid point inside a narrow band that satisfies $I(p) > 150$, then we set $\beta = 0.0$ for a while (10 iterations in this case).

In this case we observed also that the fronts stop moving, far from each other in some places, but the dual approach was able to go outside the local minimum and extract the ribbon. Figure 12 shows another example using just a low-pass

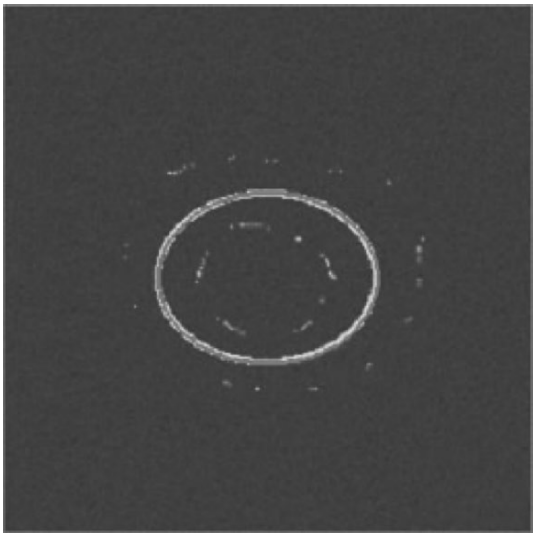


Figure 10. Final result obtained with the greedy snake model. See attached CD for color version.

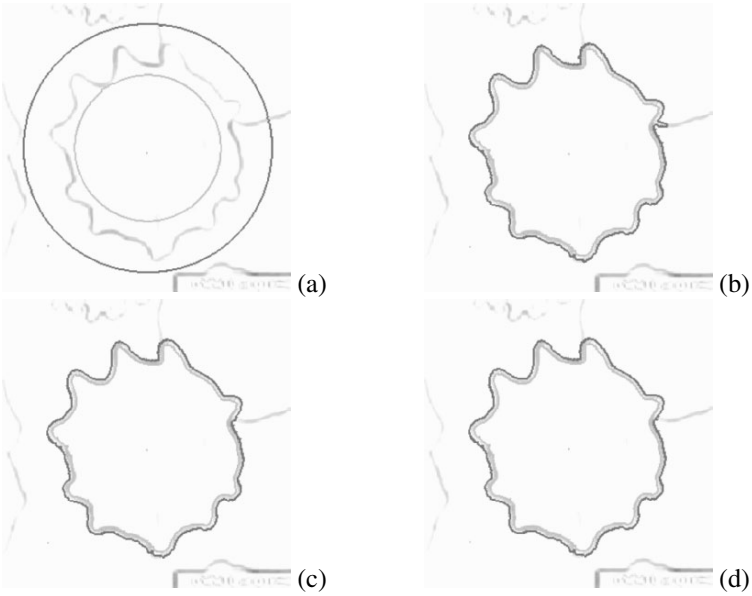


Figure 11. (a) Image and initial fronts. (b) Bandpass-filtered image and initial fronts. (c) Fronts stop moving at interaction 534 due to the low-velocity criterion. (d) Dual-Level-Set result obtained after 641 interaction steps. See attached CD for color version.

filter to smooth the same image. In this case, we observed an increase in the number of interactions if compared with the previous case. This happens because the bandpass filter can remove some textures inside the cell, but the Gaussian one is not able to do so. The Dual-Level-Set parameters are $\alpha = 0.1, \varepsilon = 2.0, \beta = 0.1$, $T = 150$, $\Delta t = 0.05$.

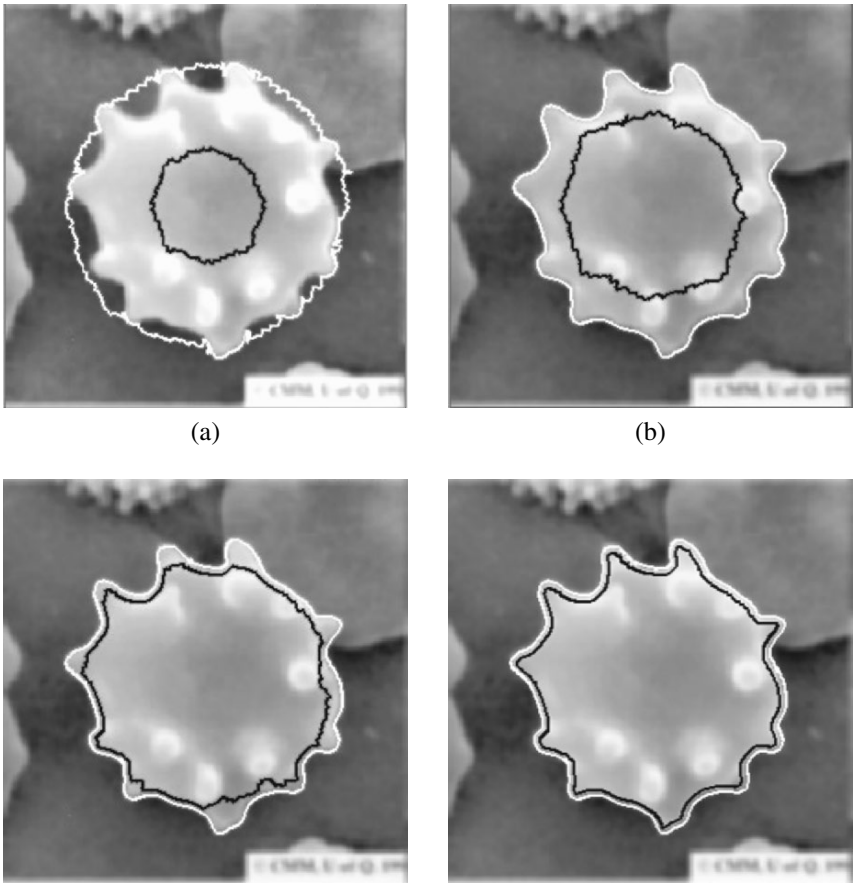
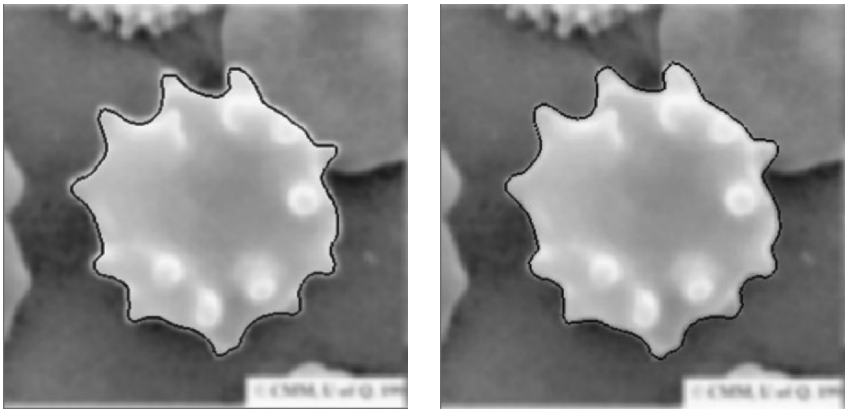


Figure 12. (a) Position of the fronts at interaction 200. (b) Interaction 400. (c) Fronts in interaction 600. (d) Dual-Level-Set result obtained after 926 interaction steps. Reprinted with permission from Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003. Copyright ©2003, Elsevier.

Figure 13a shows initialization for the greedy model, and Figure 13b depicts the final result.



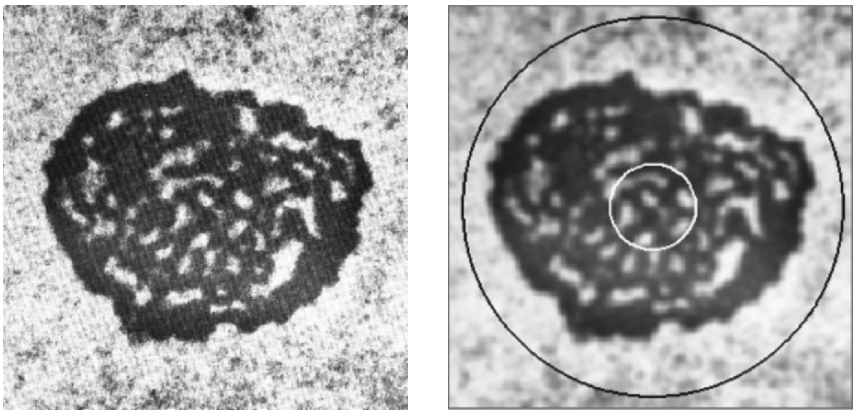
(a) (b)

Figure 13. (a) Initialization of the greedy snake model. (b) Final result. Adapted with permission from Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003. Copyright ©2003, Elsevier.

8.3. Electron Micrography Image

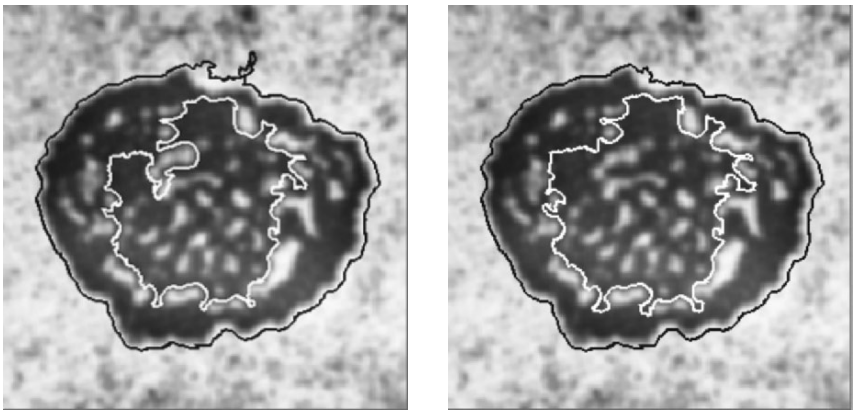
In this example we show the robustness of the Dual-Level-Set method against artifacts and noise. Let us consider Figure 14a, which depicts the image of a cell nucleolus whose resolution is 300×300. We can see that the interested boundary have points with high curvature. Figure 14b shows initialization of the Dual-Level-Set. Its parameters are $\alpha = 0.10$, $\varepsilon = 2.0$, $\beta = 0.03$, $T = 80$, and $\Delta t = 0.05$.

This example is interesting in that it shows how the topological abilities of the level set method can be useful to pass over artifacts in the image. To perform this task, we use the center of the inner front as a *reference point*. When a front undergoes a topological change, we take the obtained fronts and just discard the one(s) that does not contains the reference point. The entropy condition guarantees that this procedure is consistent. From Figures 15-a,b we can observe the fact that topological changes usually happen due to artifacts and that the front(s) that enclose the artifacts do not contain the reference point, and so we can discard them. Certainly, we should be careful if we have more then one object of interest. Such situations will be considered in further works.



(a) (b)

Figure 14. (a) Original image with a cell nucleolus. (b) Initial fronts. Adapted with permission from Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003. Copyright ©2003, Elsevier.



(a) (b)

Figure 15. (a) Outer and inner fronts just before the time point at which they undergo topological changes. (b) Fronts right after the topological changes. Adapted with permission from Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003. Copyright ©2003, Elsevier.

Figure 16 shows three evolution steps of the model as well as the dual result, shown in Figure 16d. In Figure 16b the fronts stop evolving. The driving velocity is then applied. However, different from the previous examples, we adopt the

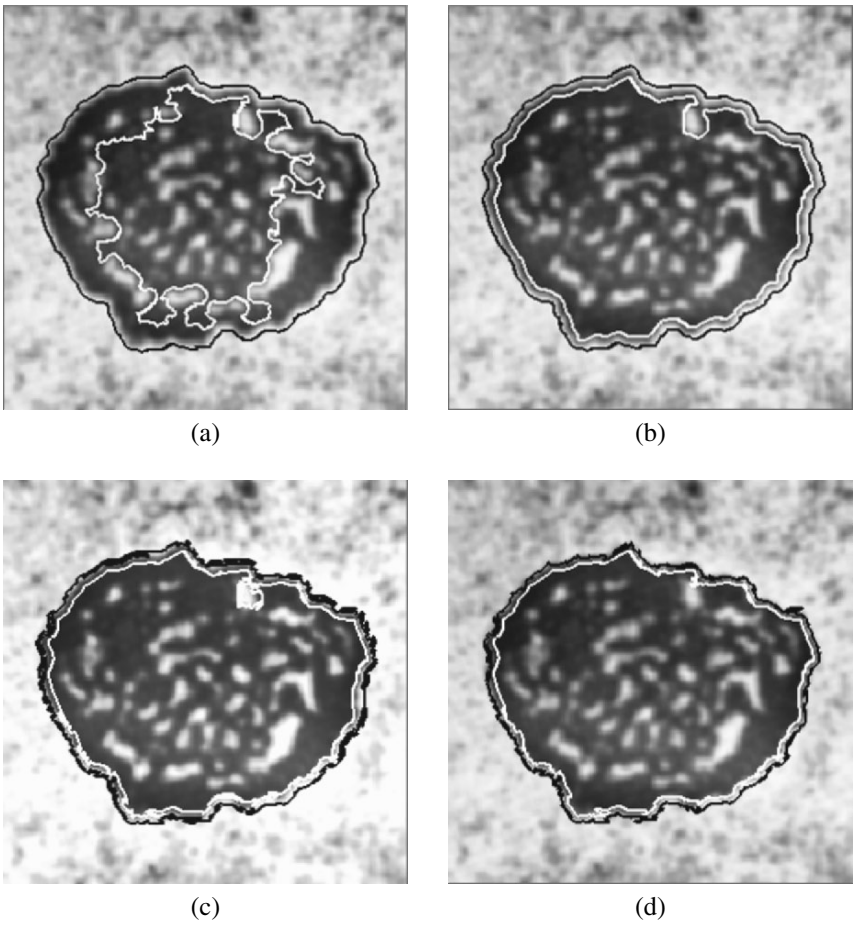


Figure 16. (a) Position of the fronts after 500 interactions. (b) Fronts stop moving due to artifacts. (c) Result after turning on the driving velocity for 20 interactions. (d) Dual-Level-Set result. Adapted with permission from Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003. Copyright ©2003, Elsevier.

following methodology. We first take a radial vector field, depicted in Figure 17, whose center is the one of the inner front of Figure 14b. We use this field to define the driving velocity given by

$$V_{drive} = |\nabla P \cdot \overrightarrow{n}|\,, \tag{61}$$

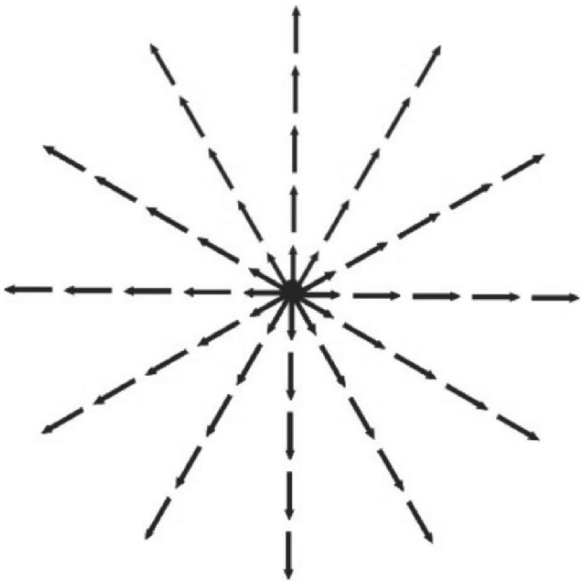


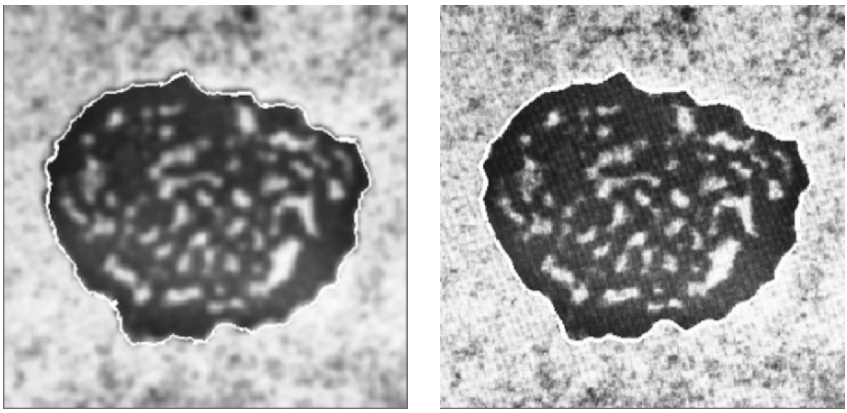
Figure 17. Radial field used to define driving velocity.

The key idea behind this field is to explore user initialization to define a vector field that can be used for perturbation of the fronts when they stop far from the target. The main point that we want to highlight is that, even with an ad-hoc procedure, the Dual-Level-Set method can go outside a local minimum and obtain the target. Figure 16c shows the result obtained when the driving velocity is turned on, given by expression 61 during 20 interactions. We observe that the fronts are less smooth now, but they are closer to undergoing a topological change and then passing over the artifact as we desire. Certainly, a more efficient driving force could be used to avoid loss of smoothness.

Figure 18a shows initialization for the greedy model, and Figure 18b depicts the extracted boundary. We can see that the method was able to get the details of the cell boundary.

9. DISCUSSION

It is important to compare our approach to the coupled constrained one presented in [27]. These methods have been used in applications where a volume has three tissue types, say $T1$, $T2$, and $T3$, and where tissue $T2$ is embedded in between tissues $T1$ and $T3$. Such an example is seen in the human brain where the GM (gray matter) is embedded between the WM (white matter) and the CSF. The



(a) (b)

Figure 18. (a) Original image with a cell nucleolus. (b) Initial fronts. Adapted with permission from Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003. Copyright ©2003, Elsevier.

key point is that there is a coupling, a thickness constraint, between the WM-GM and GM-CSF volumes. In fact, the cortical mantle has nearly constant thickness [43, 44]. Zeng proposed a level set method that begins with two embedded surfaces in the form of concentric sphere sets. The inner and outer surfaces were then evolved, driven by their own image-derived information, while maintaining the coupling in between through a thickness constraint. The model evolution is based on the following system:

$$\frac{\partial G_{in}}{\partial t} + F_{in}\,|\nabla G_{in}| = 0, \tag{62}$$

$$\frac{\partial G_{out}}{\partial t} + F_{out}\,|\nabla G_{out}| = 0, \tag{63}$$

where G_{in} and G_{out} are the embedding functions of the inner and outer surfaces, respectively. In this model, coupling between the two surfaces is obtained through the speed terms F_{in}, F_{out}, based on the distance between the two surfaces and the propagation forces. We must observe that there are two embedding functions, while in the Dual-Level-Set method there is only one. In addition, we can implement coupled constrained approaches, based on the dual method, by incorporating the constant thickness constraint in the termination condition.

The Dual-Level-Set method is also a topologically adaptable approach, just like its counterpart, the Dual-T-Snakes model, discussed in Section 4. Therefore, we must provide some comparison between these methods. The characteristic function plays a similar role to that of the embedding function in the sense that

it provides the framework for topological changes. However, an update of the former is based on exhaustive tests [9, 24], while the latter is just a consequence of the model evolution based on Eq. (34). As a practical consequence, we observed that it is easier to implement the Dual-Level-Set method than Dual-T-Snakes.

Both methods are suitable for a shared memory implementation if we observe that in the implicit formulation we must distribute the narrow band evolution while in the parametric case we should focus on the distribution of T-Snakes processing. Multi-grid techniques can also be implemented with both methods through Adaptive Mesh Refinement schemes [25].

Automatic initialization of the dual approaches in general, and for Dual-Level-Set in particular, may be a drawback of the technique because we need to set the inner and outer fronts at the beginning. This may be a tedious procedure if performed by a user, especially when the target is closer to other structures in the image. Anatomical information sometimes can be used in order to automatically perform initialization [14]. However, in general, the development of a semiautomatic system that combines techniques to assist in this task would be very useful to extend the range of applications for dual approaches.

10. CONCLUSIONS AND FUTURE WORKS

In this chapter we presented an implicit formulation for dual active contour models based on the level set approach. The so-called Dual-Level-Set method was explained and its mathematical background developed. Following previous works using another dual approach, we proposed a segmentation framework that uses the dual method to reduce the search space, and then, some suitable technique was applied to extract the boundary. The experimental results show the efficiency of such a framework in the context of cell image segmentation.

A further direction for this work will be to extend the proposed model for 3D images as well as to explore its topological abilities for segmentation of multiple objects. For the latter, we will explore the methodology developed in [17], but now using the implicit version of the dual approach.

We must also offer a quantitative comparison between Dual-Level-Set and Dual-T-Snakes. The precision and computational cost are parameters that we should consider at this point. The theoretical elements presented in Appendix 1 should be explored in order to chart the future direction of level set theory. The application of GPU techniques [45] is yet another point to be considered in coming works in order to improve the performance of the Dual-Level-Set method.

11. ACKNOWLEDGMENTS

We would like to acknowledge the Brazilian agency for scientific development (CNPq) and PCI-LNCC for financial support of this work.

12. APPENDIX

THEORETICAL PERSPECTIVES

In this section we present some theoretical elements not covered in the traditional level set literature. Our goal is to point out problems inherent in the numerical analysis and a possible direction to be explored for level set theory. Therefore, we start with numerical considerations, in the context of weak solutions, for a simplified version of our problem. Then, two mathematical tools — the generalized solutions and relaxation models for front propagation problems — are discussed within the context of interest.

Expression (35) is a second-order partial differential equation (in space) that depends on the parameter ε. If we add boundary conditions, we obtain a parabolic problem that must be analyzed with some care because complex behaviors, from the numerical viewpoint, may happen. Such an analysis is derived from the standard level set literature [25, 27] because such difficulties seem to be reduced when using the narrow band approach.

A simple example of the static problem, $F\left(x, G, G_x, G_{xx}\right) = 0$, is useful for understanding the kind of complex behaviors that may occur. We shall thus consider the following singular diffusion problem in one dimension, defined in the unit interval

$$\sigma^2 G - \varepsilon^2 G_{xx} = 0, \tag{64}$$

$$G\left(0\right) = 0, \tag{65}$$

$$G\left(1\right) = 1. \tag{66}$$

The exact solution to this problem can be written as

$$G\left(x\right) = \frac{\exp\left(-\frac{\sigma}{\varepsilon}x\right) - \exp\left(\frac{\sigma}{\varepsilon}x\right)}{\exp\left(-\frac{\sigma}{\varepsilon}\right) - \exp\left(\frac{\sigma}{\varepsilon}\right)}, \tag{67}$$

which is well behaved for $\varepsilon > 0$. However, numerically, things are not so simple. Finite-difference schemes are particular cases of weak solutions, which can be simply stated as follows. Let us consider the space $H^1\left(0,1\right)$, composed by functions that have a square-integrable first derivative, and the following spaces (see [46] and references therein):

$$H = \left\{G \in H^1\left(0,1\right); \quad G\left(0\right) = 0, G\left(1\right) = 1\right\}, \tag{68}$$

$$H_0^1 = \left\{G \in H^1\left(0,1\right); \quad G\left(0\right) = G\left(1\right) = 0\right\}. \tag{69}$$

The weak formulation of problem (64)–(66) is: find $G \in H$ such that:

$$\int_0^1 \sigma^2 G V dx + \int_0^1 \varepsilon^2 G_x V_x dx = 0, \quad \forall V \in H_0^1. \tag{70}$$

It can be shown that solution of Eqs. (70) and (64), with conditions (65)–(66), are the same. However, formulation (70) allows development of approximation schemes, well known in the field of finite-element methods [47]. Thus, let us consider a partition of the unit interval $0 = x_0 < x_1 < ... < x_{n_{\rm el}}$, where $n_{\rm el}$ is the total number of elements $\Omega^e = (x_{e-1}, x_e)$, $e = 1, ..., n_{\rm el}$. We have also a mesh parameter, $h = \max(\Omega^e)$, $e = 1, ..., n_{\rm el}$. Then consider the set of all polynomials of degree not greater than k and denote its restriction to Ω^e by $P_k(\Omega^e)$. Let

$$S_h^k(0,1) = \{G_h| \quad G_e = P_k(\Omega^e), e = 1, ..., n_{\rm el}\} \cap H, \tag{71}$$

$$V_h^k(0,1) = \{G_h| \quad G_e = P_k(\Omega^e), e = 1, ..., n_{\rm el}\} \cap H_0^1, \tag{72}$$

where G_e is the restriction of G_h to element Ω^e. The standard Galerkin method then consists of finding $G_h \in S_h^k(0,1)$ such that

$$\int_0^1 \sigma^2 G_h V_h dx + \int_0^1 \varepsilon^2 G_{h,x} V_{h,x} dx = 0, \quad \forall V_h \in V_h^k(0,1). \tag{73}$$

A critical parameter in this formulation is

$$\alpha = \frac{\sigma^2 h^2}{6\varepsilon^2}. \tag{74}$$

To see this, consider a uniform mesh, and let G_I denote the nodal value of the approximate solution at an arbitrary interior node I. Employing piecewise linear shape functions, i.e., $k = 1$, then the Ith equation derived from expression (73) is

$$(1-\alpha) G_{I-1} - 2(1+2\alpha) G_I + (1-\alpha) G_{I+1} = 0. \tag{75}$$

The general difference solution for this expression is of the form

$$G_I = cr^I. \tag{76}$$

Substituting (76) into (75) implies the following second-order equation in r, namely

$$(1-\alpha) r^2 - 2(1+2\alpha) r + (1-\alpha) = 0, \tag{77}$$

with roots

$$r_1 = \frac{1 + 2\alpha + \sqrt{(1+2\alpha)^2 - (1-\alpha)^2}}{(1-\alpha)}, \tag{78}$$

$$r_2 = \frac{1}{r_1}. \tag{79}$$

We can note that both roots are positive if $\alpha < 1$ and both are negative for $\alpha > 1$. Therefore, by (76) the difference solution will oscillate from one node to the next, which demonstrates a spurious behavior of the approximate solution as compared to the smooth exact solution in expression (67).

The main point is to check the effect of parameter α for the level set in general, and for Dual-Level-Set in particular, even with the narrow-band approach. In addition, if we take embedding function G as the image field itself, as in [5], we obtain an approach that combines color anisotropic diffusion and shock filtering, leading to a possible image flow for segmentation/simplification (see also [48]). In this case, the narrow-band technique is not useful, and so it is not safe to apply the method without the above discussion.

Another possibility to be investigated would be to work with generalized solutions of the level set problem. First, we must observe that level set Eq. (34) has the following general form:

$$G_t + F\left(x, G, D_x G, D_x^2 G\right) = 0, \tag{80}$$

where $D_x G$, $D_x^2 G$ represent the gradient and Hessian of G, respectively. For the dual model we must have

$$F\left(x, G, D_x G, D_x^2 G\right) \leq 0. \tag{81}$$

Let us suppose that F has the property

$$F(x, r, p, X) \leq F(x, s, p, Y), \quad if \quad r \leq s, Y \leq X. \tag{82}$$

Let us also suppose that G is a solution of $F = 0$. If we take a function $\varphi \in C^2$ such that $G - \varphi$ has a local maximum at a point $\widehat{x}$, then we must have $D_x G(\widehat{x}) = D_x \varphi(\widehat{x})$ and $D_x^2 G(\widehat{x}) \leq D_x^2 \varphi(\widehat{x})$. Therefore, property (82) implies

$$F\left(\widehat{x}, G(\widehat{x}), D_x \varphi(\widehat{x}), D_x^2 \varphi(\widehat{x})\right) \leq F\left(\widehat{x}, G(\widehat{x}), D_x G(\widehat{x}), D_x^2 G(\widehat{x})\right) \leq 0. \tag{83}$$

It follows that $G(x) \leq G(\widehat{x}) - \varphi(\widehat{x}) + \varphi(x)$, and, from the Taylor series, we know that

$$G(x) \leq G(\widehat{x}) + p \cdot (x - \widehat{x}) + \frac{1}{2} X \cdot (x - \widehat{x})^2 + O\left((x - \widehat{x})^2\right), \tag{84}$$

for $x \to \widehat{x}$, where $p = D_x \varphi(\widehat{x})$ and $X = D_x^2 \varphi(\widehat{x})$. Moreover, if (84) holds for some (p, X) and G is a twice-differentiable function at $\widehat{x}$, then $p = D_x G(\widehat{x})$ and

$D_x^2 G(\widehat{x}) \leq X$. Thus, if condition (82) holds, we have $F(\widehat{x}, G(\widehat{x}), p, X) \leq F(\widehat{x}, G(\widehat{x}), D_x G(\widehat{x}), D_x^2 G(\widehat{x})) \leq 0$; and, therefore,

$$F(\widehat{x}, G(\widehat{x}), p, X) \leq 0, \tag{85}$$

whenever (84) holds. We must observe that this expression does not depend on the derivatives of G. Thus, it points toward a definition of generalized solutions for Eq. (81). In fact, roughly, we can say that G is a viscosity solution of Eq. (80) if, for all smooth test functions φ

1) If $G - \varphi$ has a local maximum at a point (x, t), then:

$$\varphi_t + F\left(x, G, D_x\varphi, D_x^2\varphi\right) \leq 0,$$

2) If $G - \varphi$ has a local minimum at a point (x, t), then

$$\varphi_t + F\left(x, G, D_x\varphi, D_x^2\varphi\right) \geq 0.$$

Although this theory is outside the scope of this paper, we must emphasize that viscosity solutions theory is intimately connected with numerical analysis. It provides mathematical tools to perform convergence analysis as well as indicates how to build discretization methods or schemes for classical and more general boundary conditions (see [49] and the references therein).

Another interesting point is the application of relaxation models for front propagation problems. For instance, let us consider a Cauchy problem for the following simplified version of the Hamilton-Jacobi equation in $\Re^n$:

$$G_t + H(\nabla G) = 0, \tag{86}$$

$$G(x, 0) = G_0(x). \tag{87}$$

We can associate with this problem a system of conservation laws obtained by differentiating expression (86):

$$\mathbf{p}_t + \nabla H(\mathbf{p}) = 0, \tag{88}$$

$$\mathbf{p}(x, 0) = \mathbf{p}_0(x) = \nabla G_0(x), \tag{89}$$

where

$$\mathbf{p}(x, t) = \nabla G(x, t). \tag{90}$$

We are now going to present the relaxation approximation for this problem described in [50]:

$$\mathbf{p}_t + \nabla w = 0, \tag{91}$$

$$w_t + a\nabla \cdot \mathbf{p} = -\frac{1}{\varepsilon}(w - H(\mathbf{p})), \tag{92}$$

$$G_t + w = 0, \tag{93}$$

where w is an auxiliary function and $\varepsilon > 0$ is the relaxation time. Also, the constant a must satisfy the stability condition:

$$a > \left|\nabla_{\mathbf{p}} H\left(\mathbf{p}\right)\right|^2. \tag{94}$$

By Chapman-Enskog expansion it can be shown that, if the $O\left(\varepsilon^2\right)$ terms are ignored, expressions (91)–(93) yield

$$G_t + H\left(\nabla G\right) = \varepsilon\left(a\Delta G - \nabla_{\mathbf{p}} H\left(\mathbf{p}\right)^T \nabla_{\mathbf{p}}\left[\nabla_{\mathbf{p}} H\left(\mathbf{p}\right)\right]\right). \tag{95}$$

How do we interpret such a result in the context of front propagation? To answer this question we shall consider a front $S\left(t\right)$ propagating with normal velocity $F = -1$. The Hamilton-Jacobi equation will then be given by expression (86), and, therefore, the relaxation scheme presented yields

$$G_t + \left|\nabla G\right| = \varepsilon Tr\left(I - \left[\frac{\nabla G \otimes \nabla G}{\left|\nabla G\right|^2}\right]\Delta G\right), \tag{96}$$

where we choose $a = 1$. This equation is the level set formulation of the propagation of a front $S\left(t\right)$ with normal velocity

$$F = -1 - \varepsilon k, \tag{97}$$

where k is the mean curvature given by the formula

$$k = \nabla \cdot \mathbf{n} = -\frac{1}{\left|\nabla G\right|} Tr\left(I - \left[\frac{\nabla G \otimes \nabla G}{\left|\nabla G\right|^2}\right]\Delta G\right). \tag{98}$$

In [50] analytical comparisons are presented between models (96) and (31) as well as numerical schemes to solve system (91)–(93). Compared to the second-order, singular and degenerate level set Eq. (96), relaxation system (91)–(93) is a first-order semilinear hyperbolic system without singularity. Moreover, it retains the advantages of the level set equation, such as the ability to handle complicated geometries and changes in topology, and it allows one to capture the curvature dependency automatically without directly solving the singular curvature term numerically. Once it is semilinear we can numerically solve it in a strikingly simple way by avoiding the Riemann solvers.

13. REFERENCES

1. Kass M, Witkin A, Terzopoulos D. 1988. Snakes: active contour models. *Int J Comput Vision* **1**(4):321–331.
2. McInerney T, Terzopoulos D. 1996. Deformable models in medical image analysis: a survey. *Med Image Anal* **1**(2): 91–108.
3. Cohen LD, Cohen I. 1993. Finite-element methods for active contour models and balloons for 2D and 3D images. *IEEE Trans Pattern Anal Machine Intell* **15**(11): 1131–1147.
4. Niessen WJ, ter Haar Romery BM, Viergever MA. 1998. Geodesic deformable models for medical image analysis. *IEEE Trans Med Imaging* **17**(4):634–641.
5. Sapiro G. 1997. Color snakes. *Comput Vision Image Understand* **68**(2):247–253.
6. Black A, Yuille A, eds. 1993. *Active vision*. Cambridge: MIT Press.
7. Caselles V, Kimmel R, Sapiro G. 1997. Geodesic active contours. *Int J Comput Vision* **22**(1):61–79.
8. Malladi R, Sethian JA, Vemuri BC. 1995. Shape modeling with front propagation: a level set approach. *IEEE Trans Pattern Anal Machine Intell* **17**(2):158– 175.
9. McInerney T, Terzopoulos D. 1999. Topology adaptive deformable surfaces for medical image volume segmentation. *IEEE Trans Med Imaging* **18**(10):840–850.
10. Giraldi GA, Oliveira AF. 1999. *Convexity analysis of snake models based on hamiltonian formulation*. Technical report, Universidade Federal do Rio de Janeiro, Departamental Engenharia de Sistemas e Computação, http://www.arxiv.org/abs/cs.CV/0504031.
11. Leymarie F, Levine MD. 1993. Tracking deformable objects in the plane using and active contour model. *IEEE Trans Pattern Anal Machine Intell* **15**(6):617–634.
12. Amini AA, Weymouth TE, Jain RC. 1990. Using dynamic programming for solving variational problems in vision. *IEEE Trans Pattern Anal Machine Intell* **12**(9):855–867.
13. Giraldi GA, Vasconcelos N, Strauss E, Oliveira AF. 2001. Dual and topologically adaptable snakes and initialization of deformable models. Technical report, National Laboratory for Scientific Computation, Petropolis, Brazil, http://www.lncc.br/proj-pesq/relpesq-01.htm.
14. Bamford P, Lovell B. 1997. A two-stage scene segmentation scheme for the automatic collection of cervical cell images. In *Proceedings of TENCON '97: IEEE region 10 annual conference on speech and image technologies for computing and telecommunications*, pp. 683–686. Washington, DC: IEEE.
15. Giraldi GA, Strauss E, Oliveira AF. 2000. A boundary extraction method based on Dual-T-snakes and dynamic programming. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR '2000)*, Vol. 1, pp. 44–49. Washington, DC: IEEE.
16. Giraldi GA, Strauss E, Oliveira AF. 2001. Improving the dual-t-snakes model. In *Proceedings of the international symposium on computer graphics, image processing and vision (SIBGRAPI'2001), Florianópolis, Brazil, October 15–18*, pp. 346–353.
17. Giraldi GA, Strauss E, Oliveira AF. 2003. Dual-T-snakes model for medical imaging segmentation. *Pattern Recognit Lett* **24**(7):993–1003.
18. Gunn SR. 1996. *Dual active contour models for image feature extraction*. Phd dissertation, University of Southampton.
19. Bamford P, Lovell B. 1995. Robust cell nucleus segmentation using a viterbi search based active contour. In *Proceedings of the fifth international conference on computer vision (ICCV'95), Cambridge, MA, USA* pp. 840–845.
20. Gunn SR, Nixon MS. 1997. A robust snake implementation; a dual active contour. *IEEE Trans Pattern Anal Machine Intell* **19**(1):63–68.

21. Xu G, Segawa E, Tsuji S. 1994. Robust active contours with insensitive parameters. *Pattern Recognit* **27**(7):879–884.

22. Gunn SR, Nixon MS. 1996. Snake head boundary extraction using global and local energy minimisation. *Proc IEEE Int Conf Pattern Recognit* **2**:581–585.

23. McInerney T, Terzopoulos D. 2000. T-snakes: topology adaptive snakes. *Med Image Anal* **4**(2):73–91.

24. McInerney TJ. 1997. *Topologically adaptable deformable models for medical image analysis*. PhD dissertation. University of Toronto.

25. Sethian JA. 1996. *Level set methods: evolving interfaces in geometry, fluid mechanics, computer vision and materials sciences*. Cambridge: Cambridge UP.

26. Osher S, Sethian JA. 1988. Fronts propagation with curvature-dependent speed: algorthms based on Hamilton-Jacobi formulations. *J Comput Phys* **79**:12–49.

27. Suri JS, Liu K, Singh S, Laxminarayan S, Zeng X, Reden L. 2002. Shape recovery algorithms using level sets in 2d/3d medical imagery: a state-of-the-art review. *IEEE Trans Inf Technol Biomed* 6(1):8–28.

28. You Y-L, Xu W, Tannenbaum A, Kaveh M. 1996. Behavioral analysis of anisotropic diffusion in image processing. *IEEE Trans Image Process* **5**(11):1539–1553.

29. Mauch S. 2000. *A fast algorithm for computing the closest point and distance function*. Unpublished technical report. California Institute of Technology, September. http://www.acm.caltech.edu/ seanm/projects/cpt/cpt.pdf.

30. Sigg C, Peikert R. 2005. Optimized bounding polyhedra for GPU-based distance transform. In *Scientific visualization: the visual extraction of knowledge from data*, pp. 65–78. Ed GP Bonneau, T Ertl, GM Nielson. New York: Springer.

31. Udupa J, Samarasekera S. 1996. Fuzzy connectedness and object definition: theory, algorithms and applications in image segmentation. *Graphical Models Image Process* **58**(3), 246–261.

32. Bezdek JC, Hall LO. 1993. Review of MR image segmentation techniques using pattern recognition. *Med Phys* **20**(4):1033–1048.

33. Xu C, Pham D, Rettmann M, Yu D, Prince J. 1999. Reconstruction of the human cerebral cortex from magnetic resonance images. *IEEE Trans Med Imaging* **18**(6):467–480.

34. Pohle R, Behlau T, Toennies KD. 2003. Segmentation of 3D medical image data sets with a combination of region based initial segmentation and active surfaces. In *Progress in biomedical optics and imaging*. *Proc SPIE Med Imaging* **5203**:1225–1231.

35. Falcão AX, da Cunha BS, Lotufo RA. 2001. Design of connected operators using the image foresting transform. *Proc SPIE Med Imaging* **4322**:468–479.

36. Xu C, Prince J. 1998. Snakes, shapes, and gradient vector flow. *IEEE Trans Image Process* **7**(3):359–369.

37. Xu C, Prince JL. 2000. Global optimality of gradient vector flow. In *Proceedings of the 34th annual conference on information sciences and systems (CISS'00)*. iacl.ece.jhu.edu/pubs/p125c.ps.gz.

38. Hang X, Greenberg NL, Thomas JD. 2002. A geometric deformable model for echocardiographic image segmentation. *Comput Cardiol* **29**:77–80.

39. Leventon ME, Eric W, Grimson L, Faugeras OD. 2000. Statistical shape influence in geodesic active contours. *Proc IEEE Int Conf Pattern Recognit* 1:316–323.

40. Jain AK. 1989. *Fundamentals of digital image processing*. Englewood Cliffs, NJ: Prentice-Hall.

41. Poon CS, Braun M. 1997. Image segmentation by a deformable contour model incorporating region analysis. *Phys Med Biol* **42**:1833–1841.

42. Williams DJ, Shah M. 1992. A fast algorithm for active contours and curvature estimation. *Comput Vision Image Understand* **55**(1):14–26.

43. Zeng X, Staib LH, Schultz RT, Duncan JS. 1999. Segmentation and measurement of the cortex from 3d MR images. *IEEE Trans Med Imaging* **18**:100–111.

44. Zeng X, Staib LH, Schultz RT, Duncan JS. 1999. Segmentation and measurement of the cortex from 3d MR images using coupled-surfaces propagation. *IEEE Trans Med Imaging* **18**(10):927–937.

45. Cates JE, Lefohn AE, Whitaker RT. 2004. *GIST: an interactive, GPU-based level set segmentation tool for 3D medical images*. http://www.cs.utah.edu/research/techreports/2004/pdf/UUCS-04-007.pdf.

46. Franca LP, Madureira AL, Valentin F. 2003. *Towards multiscale functions: Enriching finite element spaces with local but not bubble-like functions*. Technical Report no. 26/2003, National Laboratory for Scientific Computing, Petropolis, Brazil.

47. Hughes TJR. 1987. *The finite element method: linear static and dynamic finite element analysis*. Englewood Cliffs: Prentice-Hall.

48. Sapiro G. 1995. *Color snakes*. Technical report, Hewlett-Packard Laboratories (guille@hpl.hp.com).

49. Crandall MG, Ishii H, Lions P-L. 1992. User's guide to viscosity solutions of second-order partial differential equations. *Bull Am Math Soc* **27**:1–67.

50. Jin S, Katsoulakis MA, Xin Z. 1999. Relaxation schemes for curvature-dependent front propagation. *Commun Pure Appl Math* **52**(12):1557–1615.

8

ACCURATE TRACKING OF MONOTONICALLY ADVANCING FRONTS

M. Sabry Hassouna and A.A. Farag

Computer Vision and Image Processing Laboratory
University of Louisville, Louisville, Kentucky

A wide range of computer vision applications — such as distance field computation, shape from shading, shape representation, skeletonization, and optimal path planning — require an accurate solution of a particular Hamilton-Jacobi (HJ) equation, known as the eikonal equation. Although the fast marching method (FMM) is the most stable and consistent method among existing techniques for solving such an equation, it suffers from a large numerical error along diagonal directions, and its computational complexity is not optimal. In this chapter, we propose an improved version of the FMM that is both highly accurate and computationally efficient for Cartesian domains. The new method is called the multi-stencils fast marching (MSFM) method, which computes a solution at each grid point by solving the eikonal equation along several stencils and then picks the solution that satisfies the fast marching causality relationship. The stencils are centered at each grid point x and cover its entire nearest neighbors. In a 2D space, two stencils cover the eight neighbors of x, while in a 3D space six stencils cover its 26 neighbors. For those stencils that do not coincide with the natural coordinate system, the eikonal equation is derived using directional derivatives and then solved using a higher-order finite-difference scheme.

1. INTRODUCTION

The *eikonal equation* is a first-order nonlinear partial differential equation (PDE) whose solution tracks the motion of a monotonically advancing front, which is found to be very useful in a wide range of computer vision and graphics

Address all correspondence to: Dr. Aly A. Farag, Professor of Electrical and Computer Engineering, University of Louisville, CVIP Lab, Room 412, Lutz Hall, 2301 South 3rd Street, Louisville, KY 40208, USA. Phone: (502) 852-7510, Fax: (502) 852-1580. aafara01@louisville.edu.

applications, such as computing distance fields from one or more source points, shape from shading, shape offsetting, optimal path planning, and skeletonization.

Several methods have been proposed to solve the eikonal equation [1–6]. The most stable and consistent method among those techniques is the fast marching method (FMM), which is applicable to both Cartesian [5, 7] and triangulated surfaces [8, 9]. The FMM combines entropy satisfying upwind schemes and a fast sorting technique to find the solution in a one-pass algorithm. Unfortunately, the FMM still has two limitations: (1) At each grid point $\boldsymbol{x}$, the method employs a 4-point stencil to exploit only the information of the adjacent four neighbors to $\boldsymbol{x}$, thus ignoring the information provided by diagonal points. As a consequence, the FMM suffers from a large numerical error along diagonal directions. (2) The computational complexity of the method is high because it stores the solutions in a narrow band that is implemented using a sorted heap data structure. The complexity of maintaining the heap is $O(\log n)$, where n is the total number of grid points. Therefore, the total complexity of the method is $O(n \log n)$.

Few methods have been proposed to improve the FMM so as to be either computationally efficient [10, 11] or more accurate [7, 12]. In [7], *The Higher Accuracy Fast Marching Method* was introduced to improve the accuracy of the FMM by approximating the gradient by a second-order finite-difference scheme whenever the arrival times of the neighbor points are available and to revert to first-order approximation in other cases.

Danielsson and Lin [12] improved the accuracy of the FMM by introducing the *shifted Grid Fast Marching* method, a modified version of the FMM. The main idea of this algorithm is to sample the cost function at half-grid positions; therefore, the cost is dependent on the marching direction. The update strategy for computing the arrival time at a neighbor point of a *known* point under the new scheme is derived from optimal control theory in a similar fashion to Tsitsiklis [13]. Therefore, the solution cannot make use of any higher-order finite-difference scheme. They proposed two solution models. The first uses 4-connected neighbors and gives the same result as the FMM, while the second employs 8-connected neighbors and gives better results than the FMM. In both models, the idea is the same, where the neighborhood around $\boldsymbol{x}$ is divided into either 4 quadrants or 8 octants. The *known* points of each quadrant/octant computes the arrival time at $\boldsymbol{x}$, which is then assigned the minimum value over all quadrant/octants.

The Group Marching Method (GMM) [10] is a modified version of the FMM that advances a group of grid points simultaneously rather than sorting the solution in a narrow band. The GMM reduces the computational complexity of the FMM to $O(n)$, while maintaining the same accuracy. The method works as follows: a group of points G are extracted from the narrow band such that their travel times do not alter each other during the update procedure. The neighbor points of G join the narrow band after computing their travel times, while the G points are tagged as known.

In [11], another method was proposed to improve the computational efficiency of the FMM by reducing its complexity to $O(n)$. This method implements the narrow band by a special data structure called an *untidy priority queue*, whose maintenance (insert/delete) operations cost $O(1)$, as opposed to a sorted heap, whose cost is $O(\log n)$. The queue is implemented using a dynamic circular array. Each computed arrival time value is quantized and used as an index to the dynamic array. Each entry (bucket) of the circular array contains a list of points with similar T values. The quantization is used only to place the grid point in the queue, while the actual T value is used to solve the eikonal equation when the grid point is selected. Therefore, errors can only occur due to wrong selection order. The authors have shown that the error introduced due to a wrong selection order is of the same order of magnitude as with the FMM.

In this chapter, we propose a highly accurate method for tracking a monotonically advancing front. The new method is called the *multi-stencils fast marching* (MSFM) method, which builds over the FMM and computes the solution at a point $\boldsymbol{x}$ by solving the eikonal equation along several stencils and then pick the solution that satisfies the fast marching causality relationship. The stencils are centered at $\boldsymbol{x}$ and cover all its neighbors. Along each stencil, the eikonal equation is derived using directional derivatives and then solved using a higher-order finite-difference scheme.

2. THE FAST MARCHING METHOD

Consider a closed interface Γ (e.g., boundary) that separates one region from another. Γ can be a curve in 2D or a surface in 3D. Assume that Γ moves in its normal direction with a known speed $F(\boldsymbol{x})$ that is either increasing or decreasing. The equation of motion of that front is given by

$$\begin{aligned} \|\nabla T(\boldsymbol{x})\| F(\boldsymbol{x}) &= 1 \\ T(\Gamma_0) &= 0, \end{aligned} \tag{1}$$

where $T(\boldsymbol{x})$ is the arrival time of Γ as it crosses each point $\boldsymbol{x}$. If the speed depends only on the position $\boldsymbol{x}$, then the equation reduces to a nonlinear first-order partial differential equation, which is known in geometrical optics as the eikonal equation. The FMM [5] solves that equation in one pass algorithm as follows. For clarity, let's consider the 2D version of the FMM. The numerical approximation of $\|\nabla T\|$ that selects the physically correct vanishing viscosity weak solution is given by Godunov [14] as

$$\max(D_{ij}^{-x}T, -D_{ij}^{+x}T, 0)^2 + \max(D_{ij}^{-y}T, -D_{ij}^{+y}T, 0)^2 = \frac{1}{F_{ij}^2}. \tag{2}$$

If the gradient ∇T is approximated by the first-order finite-difference scheme, Eq. (2) can be rewritten as

$$\max\left(\frac{T_{i,j}-\min(T_{i-1,j},T_{i+1,j})}{\Delta x_1},0\right)^2+ \max\left(\frac{T_{i,j}-\min(T_{i,j-1},T_{i,j+1})}{\Delta x_2},0\right)^2 = \frac{1}{F_{ij}^2}, \tag{3}$$

where, Δx_1 and Δx_2 are the grid spacing in the x and y directions, respectively. Let $T_1=\min(T_{i-1,j},T_{i+1,j})$, and $T_2=\min(T_{i,j-1},T_{i,j+1})$; then

$$\max\left(\frac{T_{i,j}-T_1}{\Delta x_1},0\right)^2+\max\left(\frac{T_{i,j}-T_2}{\Delta x_2},0\right)^2=\frac{1}{F_{ij}^2}. \tag{4}$$

If $T_{i,j}>\max(T_1,T_2)$, then Eq. (4) reduces to a second-order equation of the form

$$\sum_{k=1}^{2}\left(\frac{T_{i,j}-T_k}{\Delta x_k}\right)^2=\frac{1}{F_{ij}^2}; \tag{5}$$

otherwise,

$$T_{i,j}=\min(T_k+\frac{\Delta x_k}{F_{ij}})\ \ k=1,2. \tag{6}$$

The idea behind the FMM is to introduce an order in the selection of grid points during computing their arrival times, in a way similar to the Dijkstra shortest path algorithm [15]. This order is based on a causality criterion, where the arrival time T at any point depends only on the adjacent neighbors that have smaller values. During the evolution of the front, each grid point $\boldsymbol{x}$ is assigned one of three possible tags: (1) *known*, if $T(\boldsymbol{x})$ will not be changed latter; (2) *narrowband*, if $T(\boldsymbol{x})$ may be changed later, and finally, (3) *far*, if $T(\boldsymbol{x})$ is not yet computed. The FMM algorithm can be summarized as follows. Initially, all boundary points are tagged as *known*. Then, their nearest neighbors are tagged as *narrowband* after computing their arrival time by solving Eq. (4). Among all narrow-band points,

1. Extract the point with minimum T and tag it as *known*.
2. Find its nearest neighbors (*far*, or *narrowband*).
3. Update their arrival times by solving Eq. (4).
4. Go back to first step.

As a result of the update procedure, either a *far* point is tagged as a *narrowband*, or a *narrowband* point gets assigned a new arrival time that is less than its previous value.

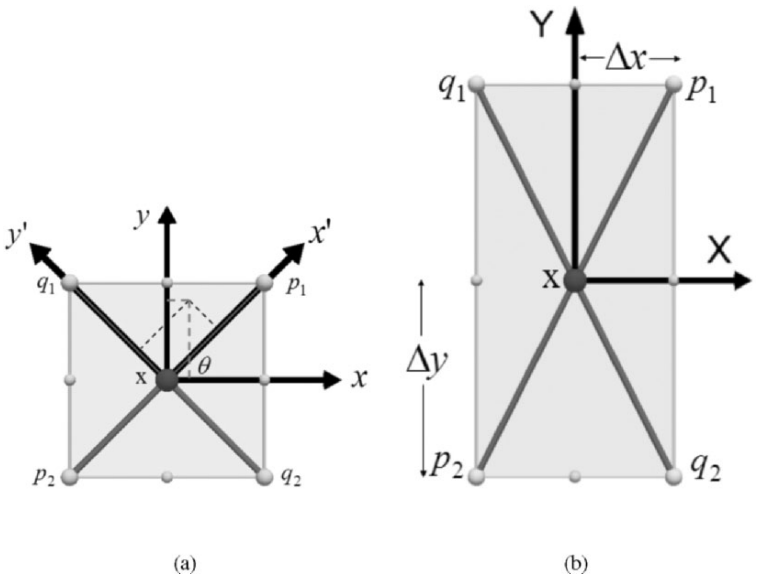


Figure 1. (a) The coordinate system is rotated by an angle θ to intersect the grid at lattice points. (b) The stencil S_j is centered at $\boldsymbol{x}$ and intersects the grid at lattice points. See attached CD for color version.

3. METHODS

All related methods except that of [12] ignore the information provided by diagonal points and hence suffer from a large error along diagonal directions. One can make use of diagonal information by either: (1) using only one stencil that always coincides with the natural coordinate system and then rotate the coordinate system several times such that the stencil intersects the grid at diagonal points, as shown in Figure 1a, or (2) use several stencils at $\boldsymbol{x}$ that cover the entire diagonal information and then approximate the gradient using directional derivatives. In both cases, the eikonal equation is solved along each stencil; then the solution at $\boldsymbol{x}$ that satisfies the causality relationship is selected. Although both methods give the same results, the first method is limited to isotropic grid-spacing, and so the second method is more general.

3.1. 2D Multi-Stencils Fast Marching Method

Consider the stencil S_j that intersects the two-dimensional Cartesian domain at lattice points p_1, p_2, q_1, and q_2, as shown in Figure 1b. Let $\overrightarrow{r}_1 = [r_{11}\ r_{12}]^T$ and $\overrightarrow{r}_2 = [r_{21}\ r_{22}]^T$ be the unit vectors along $\overline{p_2p_1}$ and $\overline{q_2q_1}$, respectively, and

D_{r_1} and D_{r_2} be the directional derivatives along $\overrightarrow{r}_1$ and $\overrightarrow{r}_2$, respectively:

$$D_{r_1} = \nabla T(\boldsymbol{x}_i) \cdot \overrightarrow{r}_1 = r_{11} T_x + r_{12} T_y, \tag{7}$$
$$D_{r_2} = \nabla T(\boldsymbol{x}_i) \cdot \overrightarrow{r}_2 = r_{21} T_x + r_{22} T_y, \tag{8}$$

which can be rewritten as

$$\begin{pmatrix} D_{r_1} \\ D_{r_2} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} \begin{pmatrix} T_x \\ T_y \end{pmatrix}. \tag{9}$$

Thus,

$$D_r = R\, \nabla T(\boldsymbol{x}_i), \tag{10}$$
$$\nabla T(\boldsymbol{x}_i) = R^{-1} D_r, \tag{11}$$
$$\left(\nabla T(\boldsymbol{x}_i)\right)^T = \left(R^{-1} D_r\right)^T = {D_r}^T\, R^{-T}. \tag{12}$$

Since

$$\|\nabla T(\boldsymbol{x}_i)\|^2 = \left(\nabla T(\boldsymbol{x}_i)\right)^T\ \nabla T(\boldsymbol{x}_i), \tag{13}$$

then,

$$\|\nabla T(\boldsymbol{x}_i)\|^2 = D_r^T\, R^{-T}\, R^{-1}\, D_r \tag{14}$$
$$= D_r^T\, \left(R R^T\right)^{-1} D_r. \tag{15}$$

$$R R^T = \begin{pmatrix} \|\overrightarrow{r}_1\|^2 & \overrightarrow{r}_1 \cdot \overrightarrow{r}_2 \\ \overrightarrow{r}_2 \cdot \overrightarrow{r}_1 & \|\overrightarrow{r}_2\|^2 \end{pmatrix} = \begin{pmatrix} 1 & \cos(\phi) \\ \cos(\phi) & 1 \end{pmatrix}, \tag{16}$$

where ϕ is the angle between the directional vectors. By substituting Eq. (16) into Eq. (14), the gradient of the arrival time along an arbitrary stencil is given by

$$\|\nabla T\|^2 = D_{r_1}^2 - 2 D_{r_1} D_{r_2} \cos(\phi) + D_{r_2}^2 = \frac{\sin^2(\phi)}{F^2(\boldsymbol{x}_i)}. \tag{17}$$

The first-order approximation of the directional derivative, D_{r_j}, that obeys the viscosity solution given the arrival time of a *known* neighbor point $\boldsymbol{x}_j$ is

$$D_{r_j} = \max\left(\frac{T(\boldsymbol{x}_i) - T(\boldsymbol{x}_j)}{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}, 0\right), \quad j = 1, 2. \tag{18}$$

Also, the second-order approximation of the directional derivative, D_{r_j}, that obeys the viscosity solution given the arrival times of the *known* neighbor points $\boldsymbol{x}_j$ and $\boldsymbol{x}_{j-1}$, where $\boldsymbol{x}_{j-1} \leq \boldsymbol{x}_j$ [7] is

$$D_{r_j} = \max\left(\frac{3\, T(\boldsymbol{x}_i) - 4\, T(\boldsymbol{x}_j) + T(\boldsymbol{x}_{j-1})}{2\, \|\boldsymbol{x}_i - \boldsymbol{x}_j\|}, 0\right)$$
$$= \max\left(\frac{3}{2\, \|\boldsymbol{x}_i - \boldsymbol{x}_j\|} \left[T(\boldsymbol{x}_i) - T(v_j)\right], 0\right), \tag{19}$$

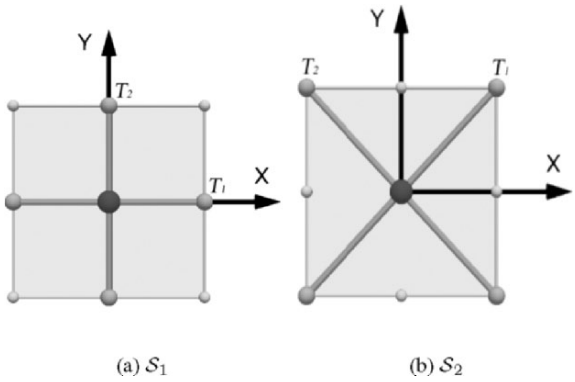


Figure 2. The proposed stencils for a 2D Cartesian domain. See attached CD for color version.

where

$$T(v_j) = \frac{4\,T(\boldsymbol{x}_j) - T(\boldsymbol{x}_{j-1})}{3}. \tag{20}$$

In 2D, we use two stencils $\mathcal{S}_1$ and $\mathcal{S}_2$. The nearest neighbors are covered by $\mathcal{S}_1$, while $\mathcal{S}_2$ covers the diagonal ones. To simplify discussion, let's assume isotropic grid spacing with $\Delta x = \Delta y = h$, then $\phi = 90^\circ$. Let

$$\begin{aligned} T_1 &= \min\left(T(p_1), T(p_2)\right), \\ T_2 &= \min\left(T(q_1), T(q_2)\right). \end{aligned} \tag{21}$$

The $\mathcal{S}_1$ stencil is aligned with the natural coordinate system as shown in Figure 2a, while the $\mathcal{S}_2$ stencil is aligned with the diagonal neighbor points as shown in Figure 2b. Since $\phi = 90^\circ$, then $(R\,R^T)^{-1} = I$, and hence,

$$D_r{}^T\, I\, D_r = D_{r_1}^2 + D_{r_2}^2 = \frac{1}{F^2(\boldsymbol{x}_i)}. \tag{22}$$

For both stencils, if

$$T(\boldsymbol{x}_i) > \max\left(T_1, T_2\right), \tag{23}$$

or

$$T(\boldsymbol{x}_i) > \max\left(T(v_1), T(v_2)\right), \tag{24}$$

Table 1. Coefficients of the quadratic equation for both $\mathcal{S}_1$ and $\mathcal{S}_2$ using different order approximations of the directional derivatives

Stencil	1st-order $\tau_j = T_j$	2nd-order $\tau_j = T(v_j)$
$\mathcal{S}_1$	$g(h) = 1/h^2$	$g(h) = 9/4h^2$
$\mathcal{S}_2$	$g(h) = 1/2h^2$	$g(h) = 9/8h^2$

then, Eq. (22) is generalized to

$$\sum_{j=1}^{2} g(h)(T(\boldsymbol{x}_i) - \tau_j)^2 = g(h)\left(\sum_{j=1}^{2} a_j T^2(\boldsymbol{x}_i) + b_j T(\boldsymbol{x}_i) + c_j\right) = \frac{1}{F^2(\boldsymbol{x}_i)}, \tag{25}$$

where $a_j = 1$, $b_j = -2\tau_j$, and $c_j = \tau_j^2$. The values of $g(h)$ and τ_j are given in Table 1. They depend on the stencil shape and the order of approximations of the directional derivatives.

3.1.1. Upwind Condition

The solution of the quadratic equation must satisfy the FMM causality relationship, which states that $T(\boldsymbol{x}_i) > \max(\tau_1, \tau_2)$; and then

$$T(\boldsymbol{x}_i) - \tau_j = \frac{-b + \sqrt{b^2 - 4ac}}{2a} - \tau_j > 0, \quad j = 1, 2. \tag{26}$$

After some algebra, it is easy to show that the following conditions must be satisfied:

$$a\tau_j^2 + b\tau_j + c < 0, \quad j = 1, 2. \tag{27}$$

The first- and second-order directional derivatives can be put in the form

$$D_{r_j} = k_j T(\boldsymbol{x}_i) - n_j, \quad j = 1, 2. \tag{28}$$

The values of k_j and n_j depend on the order of approximation and the stencil's shape. Substituting Eq. (28) into (17), we get the coefficient values of Eq. (27):

$$\begin{aligned} a &= k_1^2 - 2k_1k_2\cos(\phi) + k_2^2, \\ b &= -2k_1n_1 + 2(k_1n_2 + k_2n_1)\cos(\phi) - 2k_2n_2, \\ c &= n_1^2 - 2n_1n_2\cos(\phi) + n_2^2 - \sin^2(\phi)F^{-2}(\boldsymbol{x}_i). \end{aligned} \tag{29}$$

Table 2. Coefficients of the upwind condition for both $\mathcal{S}_1$ and $\mathcal{S}_2$ using different-order approximations of the directional derivatives

Stencil	1st-order $\tau_j = T_j$	2nd-order $\tau_j = T(v_j)$
$\mathcal{S}_1$	$f(\Delta x, \Delta y) = \min(\Delta x, \Delta y)$	$f(\Delta x, \Delta y) = 2\min(\Delta x, \Delta y)$
$\mathcal{S}_2$	$f(\Delta x, \Delta y) = \sqrt{\Delta^2 x + \Delta^2 y}$	$f(\Delta x, \Delta y) = 2\sqrt{\Delta^2 x + \Delta^2 y}$

Substituting Eq. (29) into (27), we get the following upwind condition:

$$|\tau_1 - \tau_2| < \frac{f(\Delta x, \Delta y)\sin(\phi)}{F(\boldsymbol{x}_i)}. \tag{30}$$

The values of $f(\Delta x, \Delta y)$ and τ_j for different stencils are given in Table 2.

3.2. 3D Multi-Stencils Fast Marching Method

The Multi-Stencils Fast Marching Method can be extended easily to the three-dimensional Cartesian domain. Consider the stencil S_j that intersects the grid at lattice points l_1, l_2, p_1, p_2, q_1, and q_2. Let $\overrightarrow{r}_1 = [r_{11}\ r_{12}\ r_{13}]^T$, $\overrightarrow{r}_2 = [r_{21}\ r_{22}\ r_{23}]^T$, and $\overrightarrow{r}_3 = [r_{31}\ r_{32}\ r_{33}]^T$ be the unit vectors along $\overline{l_2 l_1}$, $\overline{p_2 p_1}$, and $\overline{q_2 q_1}$, respectively, α the angle between $\overrightarrow{r}_1$ and $\overrightarrow{r}_2$, β the angle between $\overrightarrow{r}_2$ and $\overrightarrow{r}_3$, and γ the angle between $\overrightarrow{r}_1$ and $\overrightarrow{r}_3$, and finally, D_{r_1}, D_{r_2}, and D_{r_3} the directional derivatives along $\overrightarrow{r}_1$, $\overrightarrow{r}_2$, and $\overrightarrow{r}_3$, respectively. In 3D we use six stencils $\mathcal{S}_j$, where $j \in [1, 6]$. The nearest neighbor points are covered by $\mathcal{S}_1$, while the rest of stencils cover the diagonal ones as shown in Figure 3. According to the chosen stencils, $\overrightarrow{r}_1 \perp \overrightarrow{r}_2$ and $\overrightarrow{r}_1 \perp \overrightarrow{r}_3$; then $\alpha = \gamma = 90°$; therefore,

$$RR^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & \cos(\beta) \\ 0 & \cos(\beta) & 1 \end{pmatrix}. \tag{31}$$

Again, to simplify discussion, let's assume isotropic grid spacing with $\Delta x = \Delta y = \Delta z = h$. Let

$$\begin{aligned} T_1 &= \min(T(l_1), T(l_2)), \\ T_2 &= \min(T(p_1), T(p_2)), \\ T_3 &= \min(T(q_1), T(q_2)). \end{aligned} \tag{32}$$

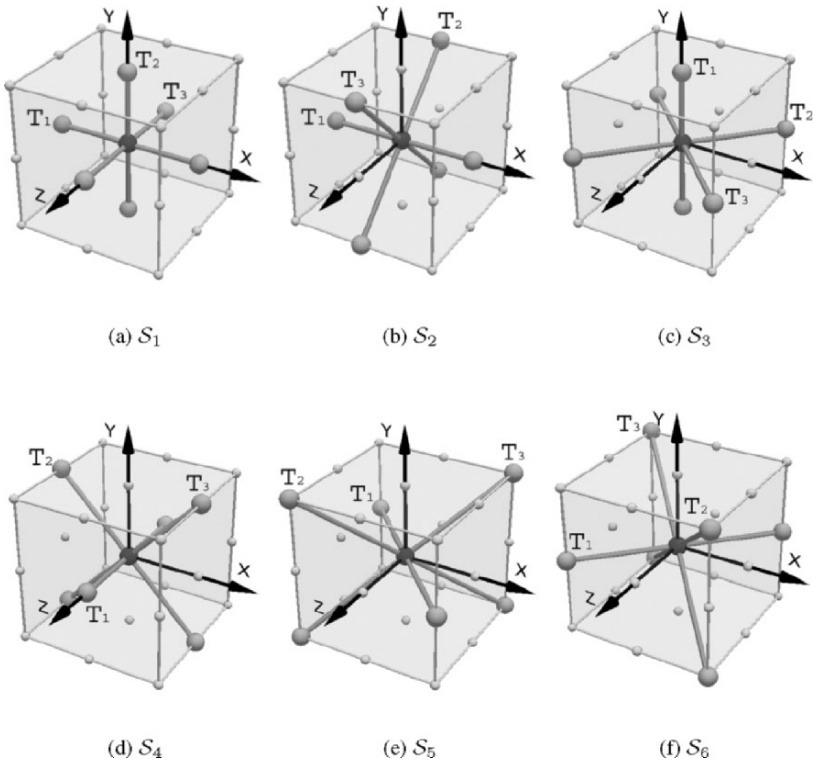


Figure 3. The proposed stencils for a 3D Cartesian domain. See attached CD for color version.

Then by substituting $\beta = 90$ in Eq. (22), we get $(R\,R^T)^{-1} = I$, and,

$$D_r{}^T\, I\, D_r = D_{r_1}^2 + D_{r_2}^2 + D_{r_3}^2 = \frac{1}{F^2(\boldsymbol{x}_i)}. \tag{33}$$

For all stencils, if

$$T(\boldsymbol{x}_i) > \max\left(T_1, T_2, T_3\right), \tag{34}$$

or

$$T(\boldsymbol{x}_i) > \max\left(T(v_1), T(v_2), T(v_3)\right), \tag{35}$$

Eq. (33) reduces to a second-order equation of the form

$$\sum_{j=1}^{3} g_j(h)\left(a_j T^2(\boldsymbol{x}_i) + b_j T(\boldsymbol{x}_i) + c_j\right) = \frac{1}{F^2(\boldsymbol{x}_i)}, \tag{36}$$

where $a_j = 1$, $b_j = -2\tau_j$, and $c_j = \tau_j^2$. The values of $g(h)$ and τ_j are given in Table 3. Again, they depend on the stencil shape and the order of approximations of the directional derivatives.

Table 3. The value of $g_j(h)$ according to stencil shape and order of approximation

Stencil	1st-order $\tau_j = T_j$			2nd-order $\tau_j = T(v_j)$		
	$g_1(h)$	$g_2(h)$	$g_3(h)$	$g_1(h)$	$g_2(h)$	$g_3(h)$
$\mathcal{S}_1$	$1/h^2$	$1/h^2$	$1/h^2$	$9/4h^2$	$9/4h^2$	$9/4h^2$
$\mathcal{S}_2$	$1/h^2$	$1/2h^2$	$1/2h^2$	$9/4h^2$	$9/8h^2$	$9/8h^2$
$\mathcal{S}_3$	$1/h^2$	$1/2h^2$	$1/2h^2$	$9/4h^2$	$9/8h^2$	$9/8h^2$
$\mathcal{S}_4$	$1/h^2$	$1/2h^2$	$1/2h^2$	$9/4h^2$	$9/8h^2$	$9/8h^2$
$\mathcal{S}_5$	$1/2h^2$	$1/3h^2$	$1/3h^2$	$9/8h^2$	$9/12h^2$	$9/12h^2$
$\mathcal{S}_6$	$1/2h^2$	$1/3h^2$	$1/3h^2$	$9/8h^2$	$9/12h^2$	$9/12h^2$

3.2.1. Upwind Condition

Although deriving the upwind condition for 3D domains is straightforward, it is very complicated, and its evaluation at every grid point is time consuming. A better way is to make sure that the computed $T(\boldsymbol{x}_i)$ is higher than the values of the three adjacent neighbors (T_1, T_2, and T_3) that participated in the solution. If the check is true, we accept the solution; otherwise, we check if $T(\boldsymbol{x}_i)$ is greater than the two remaining values of the adjacent points. If the check is true, we solve the eikonal equation based on their values and return the maximum solution; otherwise, we accept the following solution:

$$\min\left(T_j + \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{F(\boldsymbol{x}_i)}\right), \quad j = 1, 2, 3. \tag{37}$$

4. NUMERICAL EXPERIMENTS

4.1. Two-Dimensional Space

In this section, we conduct two experiments to study the accuracy of the proposed first-order multi-stencils fast marching (MSFM_1) and second-order multi-stencils fast marching (MSFM_2) methods against the first-order fast marching method (FMM_1), the second-order fast marching method (FMM_2), and the shifted grid fast marching (SGFM) method. The exact analytical solution of the eikonal equation given a particular speed model is assumed to be the gold standard. Since

the analytical solution is hard to find at least for complex speed models, we instead start from a continuous and differentiable function $T_i(\boldsymbol{x})$ as given by

$$\begin{aligned}
T_1(\boldsymbol{x}) &= \sqrt{(x-x_0)^2+(y-y_0)^2}-1, \\
T_2(\boldsymbol{x}) &= \frac{(x-x_0)^2}{100}+\frac{(y-y_0)^2}{20}, \\
T_3(\boldsymbol{x}) &= \frac{x^2}{10}\exp(\frac{y}{20})+\frac{y^2}{50}, \\
T_4(\boldsymbol{x}) &= 1-\cos(\frac{x-x_0}{20})\cos(\frac{y-y_0}{20}), \\
T_5(\boldsymbol{x}) &= \sqrt{(x-x_0)^2+(y-y_0)^2+(z-z_0)^2}-1, \\
T_6(\boldsymbol{x}) &= \frac{(x-x_0)^2}{100}+\frac{(y-y_0)^2}{20}+\frac{(z-z_0)^2}{20}.
\end{aligned} \tag{38}$$

The corresponding speed functions are derived as the reciprocal of $\|\nabla T(\boldsymbol{x})\|$. The approximated arrival time, $\acute{T}(\boldsymbol{x})$, by each method is computed and then compared with $T(\boldsymbol{x})$ using different error metrics. The iso-contours of the analytical solution of the arrival times for 2D domains are shown in Figure 4. The first speed function is given by $F_1(\boldsymbol{x})$, which corresponds to a moving front from a source point (x_0, y_0) with a unit speed. In this case, the computed arrival time corresponds to a distance field, which is of interest in many computer vision and graphics applications such as shape representation, shape-based segmentation, skeletonization, and registration. The test is performed three times from one or more source points; one source point at the middle of the grid (high curvature solution), one source point at the corner of the grid (smooth solution), and two source points within the grid (solution with shock points). To measure the error between the computed arrival time, $\acute{T}(\boldsymbol{x})$, and the analytical solution, $T_{analytic}(\boldsymbol{x})$, we employ the following error metrics:

$$L_1 = average(|\acute{T} - T_{analytic}|), \tag{39}$$

$$L_2 = average(|\acute{T} - T_{analytic}|^2), \tag{40}$$

$$L_\infty = \max(|\acute{T} - T_{analytic}|). \tag{41}$$

The numerical errors are listed in Tables 4 and 5.

In Figure 5, we show the iso-contours of the error curves when solving for the arrival time for a wave propagating with $F = 1.0$ from one and two source points, when applying FMM_2 and MSFM_2. Notice that the errors are small along the horizontal and vertical directions and increase rapidly in the regions around 45° when using FMM_2, as shown in Figure 5a,c. However, when using the proposed method, the errors are small along all the directions that are covered by all stencils, as shown in Figure 5(b,d). In Figure 6, we zoom on the the iso-contours of the exact solution (solid), FMM_2 (dashed dot), and MSFM_2 (dashed) for two different waves. The first wave propagates from the middle of the grid (51,51), while the

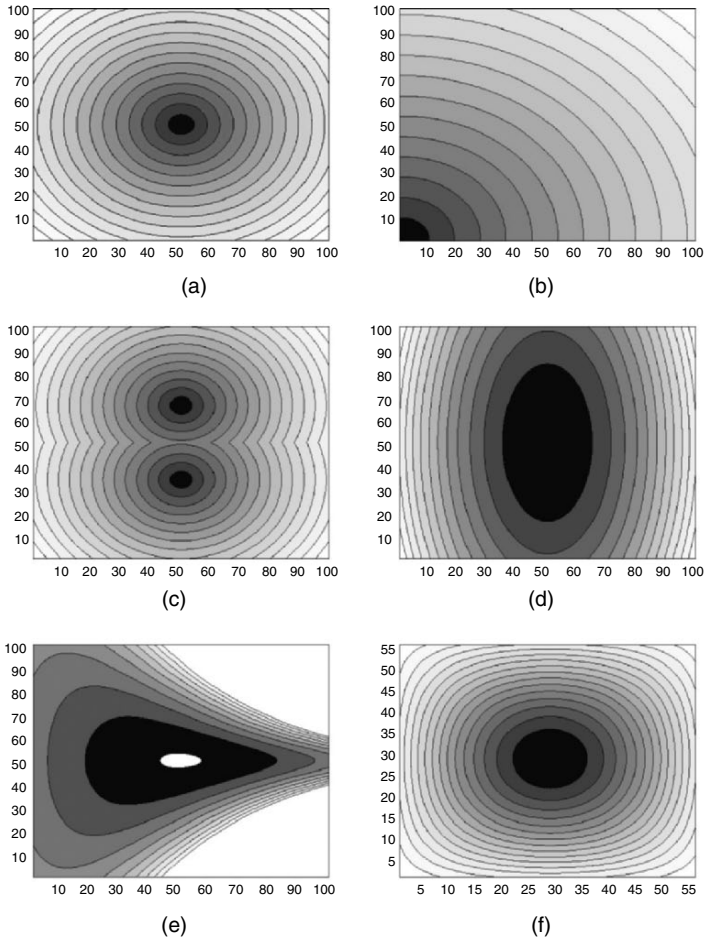


Figure 4. Iso-contours of the analytical solution of $T(\boldsymbol{x})$ for T_1 (a–c) at different source points. (d) T_2, (e) T_3, and (f) T_4.

Table 4. L_1, L_2, and L_∞ errors of the computed arrival time $\acute{T}_1(\boldsymbol{x})$ under unit speed from different source points for a grid of size 101×101

Time $T(\boldsymbol{x})$	$T_1 = \sqrt{(x-x_0)^2 + (y-y_0)^2} - 1$					
Source Point(s)	(51,51)			(1,1)		
Method/Error	L_1	L_2	L_∞	L_1	L_2	L_∞
FMM$_1$	0.746104	0.697090	1.314846	0.859562	0.951622	1.541677
MSFM$_1$	0.291622	0.112154	0.577841	0.350694	0.167881	0.719245
FMM$_2$	0.352778	0.148263	0.580275	0.346720	0.148236	0.586300
MSFM$_2$	**0.069275**	**0.007711**	**0.215501**	**0.058817**	**0.004436**	**0.212368**

The source points are located at (51,51) and (1,1).

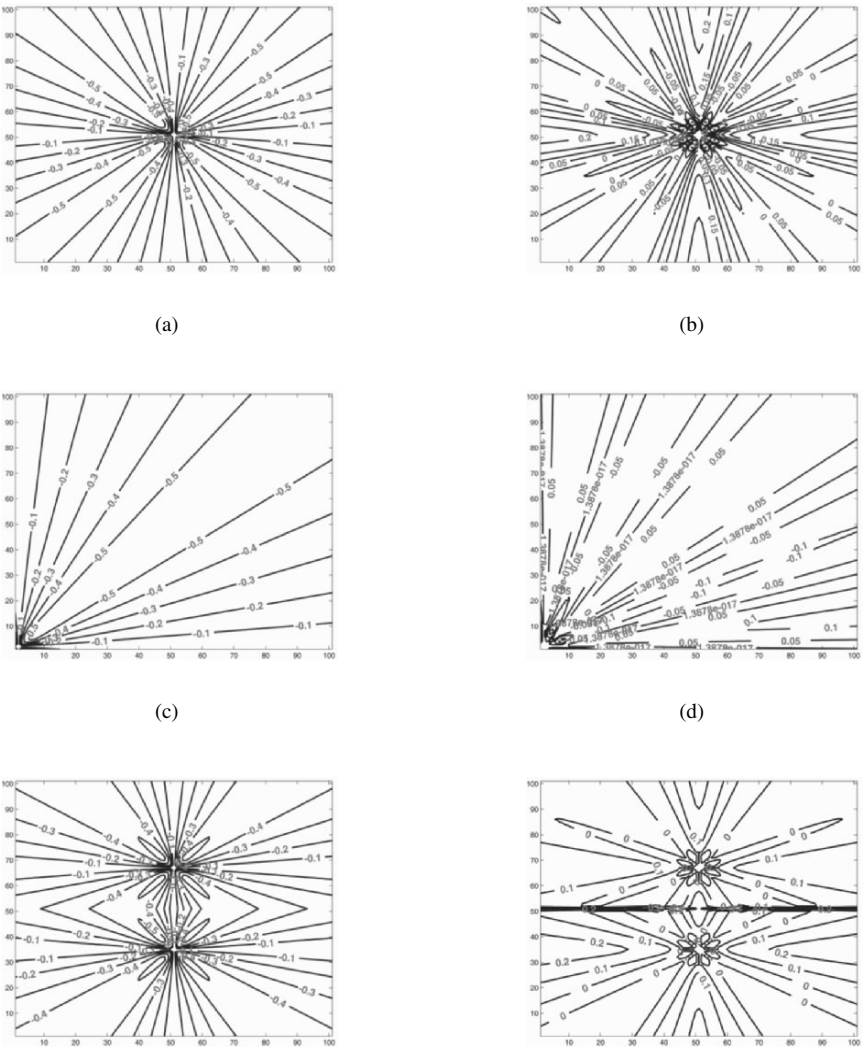
Table 5. L_1, L_2, and L_∞ errors of the computed arrival time $\acute{T}_1(\boldsymbol{x})$ under unit speed for a grid of size 101×101

Time $T(\boldsymbol{x})$	$T_1 = \sqrt{(x-x_0)^2 + (y-y_0)^2} - 1$		
Source Point(s)	(51,35) and (51,67)		
Method/Error	L_1	L_2	L_∞
FMM$_1$	0.617013	0.496484	1.200993
MSFM$_1$	0.274036	0.098130	0.577841
FMM$_2$	0.300247	0.110171	0.536088
MSFM$_2$	**0.074058**	**0.009852**	**0.420539**

Two source points are located at (51,35) and (51,67).

Table 6. L_1, L_2, and L_∞ errors of the computed arrival times $\acute{T}_2(\boldsymbol{x})$ and $\acute{T}_3(\boldsymbol{x})$ under complex speed functions for a grid of size 101×101

Time $T(\boldsymbol{x})$	$T_2 = \frac{(x-x_0)^2}{100} + \frac{(y-y_0)^2}{20}$			$T_3 = \frac{x^2}{10}\exp(\frac{y}{20}) + \frac{y^2}{50}$		
Source Point(s)	(51,51)			(51,51)		
Method/Error	L_1	L_2	L_∞	L_1	L_2	L_∞
FMM$_1$	1.514851	2.847438	3.000000	6.442591	120.800251	63.109117
MSFM$_1$	1.514851	2.847438	3.000000	6.297684	120.221017	63.109040
FMM$_2$	0.114048	0.013165	0.120108	0.388426	0.243170	1.758151
MSFM$_2$	**0.047245**	**0.002503**	**0.104651**	**0.160555**	**0.052614**	**1.356844**

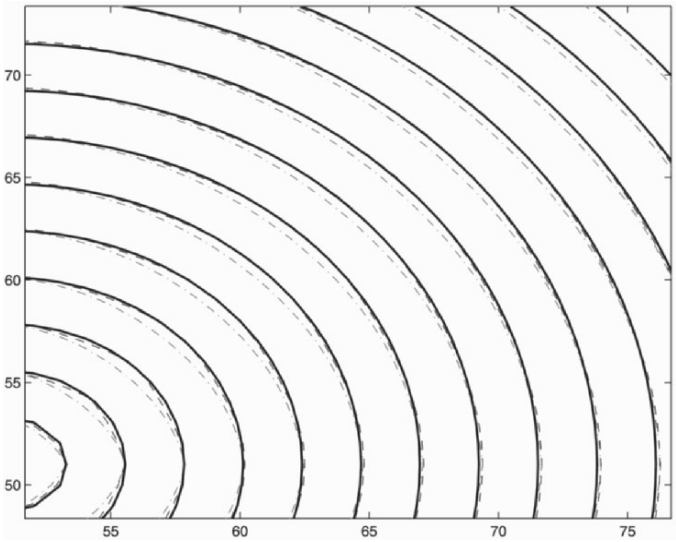


(a) (b)

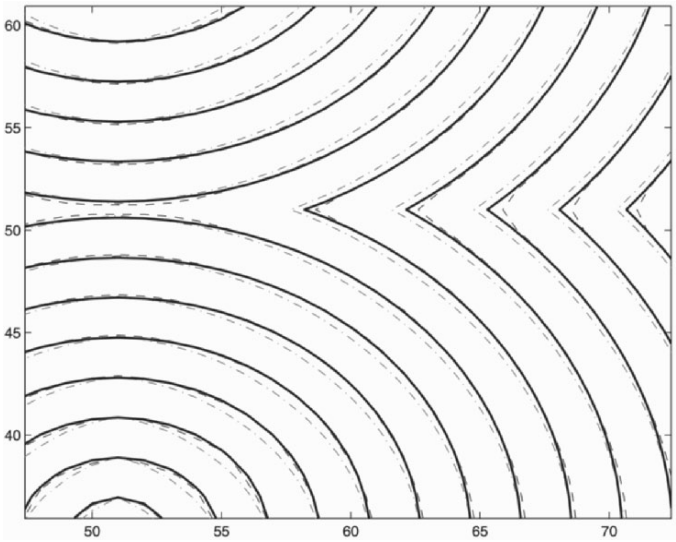
(c) (d)

(e) (f)

Figure 5. Iso-contours of the error curves when applying (a,c,e) FMM_2 and (b,d,f) the proposed method ($MSFM_2$). See attached CD for color version.



(a)



(b)

Figure 6. Iso-contours of the exact solution (solid), FMM_2 (dashed dot), and $MSFM_2$ (dashed) for a wave propagating from (a) one source point and (b) two source points. See attached CD for color version.

second one propagates from two different source points (51,35) and (51,67). It is obvious that in both cases MSFM_2 is more accurate than FMM_2. Also, the computed iso-contours by the proposed method nearly coincide with the exact analytical solution.

In Table 6, we list the numerical errors by two complex speed models (F_2 and F_3) that are functions of the spatial coordinates.

In [12], the authors have tested their method using different test functions (F_1 and F_4) on a grid of size 56×56 points with $\Delta x = \Delta y = 1$. In this experiment, we compare the proposed methods (MSFM_1 and MSFM_2) against their approach under the same experimental conditions. The numerical errors are listed in Table 7.

4.2. Three-Dimensional Space

In this experiment, we compare the proposed methods (MSFM_1 and MSFM_2) against FMM_1 and FMM_2 under different speed functions (F_5 and F_6). The size of the test grid is $41 \times 41 \times 41$ points with $\Delta x = \Delta y = \Delta z = 1$. No 3D results have been reported in [12]. The first speed function is given by $F_5(\boldsymbol{x})$, which corresponds to a moving front from a source point (x_0, y_0, z_0) with a unit speed. Again, we performed the test twice from different source points to test the quality of high curvature and smooth solutions.

5. DISCUSSION

The numerical error results of Tables 4–9 show that the proposed methods (MSFM_1 and MSFM_2) give the most accurate results among all related techniques, especially when using the second-order approximation of the directional derivatives. Since MSFM_1 is more accurate than the original FMM_1, the former method is used to initialize the points near the boundary when using second-order derivatives.

The worst case complexity of the proposed method is still $O(n \log n)$; however, the computational time is higher. The computational complexity can be reduced to $O(n)$ by implementing the narrow band as an untidy priority queue [11].

In 2D space, the computational time is nearly the same as the FMM. However, in 3D space, the computational time is approximately 2–3 times higher than the FMM. In 3D, each voxel $\boldsymbol{x}$ has 6 voxels that share a face (*F-connected*), 12 voxels that share an edge (*E-connected*), and 8 voxels that share a vertex (*V-connected*). According to the proposed stencils, $\mathcal{S}_1$ covers the *F-connected*, $\mathcal{S}_j$, where $j \in [1, 4]$ cover both *F-connected* and *E-connected*, and $\mathcal{S}_j$, where $j \in [1, 6]$ cover the entire 26-connected neighbors. To strike a balance between high accuracy and minimal computational time, our numerical experiments have shown that we can restrict the proposed method to 18 neighbors without apparent loss in accuracy.

Table 7. L_1, L_2, and L_∞ errors of the computed arrival times $\acute{T}_1(\boldsymbol{x})$ and $\acute{T}_4(\boldsymbol{x})$ by the proposed methods and SGFM under different speed functions for a grid of size 56×56

Time $T(\boldsymbol{x})$	$T_1 = \sqrt{(x-x_0)^2+(y-y_0)^2}-1$			$T_4 = 1-\cos(\frac{x-x_0}{20})\cos(\frac{y-y_0}{20})$		
Source Points(s)	(23,23)			(23,23)		
Method/Error	L_1	L_2	L_∞	L_1	L_2	L_∞
SGFM	n/a	0.0276	0.2754	n/a	0.0504	0.5245
MSFM$_1$	0.235839	0.073034	0.468195	0.023620	0.000599	0.032788
MSFM$_2$	**0.063551**	**0.006169**	**0.205003**	**0.000737**	**0.000001**	**0.002494**

Table 8. L_1, L_2, and L_∞ errors of the computed arrival time $\acute{T}_5(\boldsymbol{x})$ by the proposed methods and the FMMs under different speed functions for a grid of size $41 \times 41 \times 41$

Time $T(\boldsymbol{x})$	$T_5 = \sqrt{(x-x_0)^2+(y-y_0)^2+(z-z_0)^2}-1$					
Source Point(s)	(21,21,21)			(1,1,1)		
Method/Error	L_1	L_2	L_∞	L_1	L_2	L_∞
FMM$_1$	1.101981	1.338472	1.815051	1.141622	1.470947	1.891439
MSFM$_1$	0.560443	0.360977	0.953934	0.621374	0.451035	1.141499
FMM$_2$	0.517593	0.282983	0.724932	0.491958	0.259712	0.779457
MSFM$_2$	**0.275705**	**0.0079831**	**0.386620**	**0.170222**	**0.0037633**	**0.388761**

Table 9. L_1, L_2, and L_∞ errors of the computed arrival time $\acute{T}_6(\boldsymbol{x})$ by the proposed methods and the FMMs under different speed functions for a grid of size $41 \times 41 \times 41$

Time $T(\boldsymbol{x})$	$T_6 = \frac{(x-x_0)^2}{100}+\frac{(y-y_0)^2}{20}+\frac{(z-z_0)^2}{20}$		
Source Point(s)	(1,1,21)		
Method/Error	L_1	L_2	L_∞
FMM$_1$	1.656341	3.191547	3.390000
MSFM$_1$	1.656341	3.191547	3.390000
FMM$_2$	0.157317	0.025071	0.165000
MSFM$_2$	**0.023103**	**0.000990**	**0.118464**

6. CONCLUSION

In this chapter, we have introduced the multi-stencils fast marching method as an improved version of the original FMM for solving the eikonal equation for both 2D and 3D Cartesian domains. The idea of the method is to solve the eikonal equation at each grid point $\boldsymbol{x}$ along several stencils that cover its neighbors, and then picks the solution that satisfies the FMM causality relationship. Unlike the FMM and its variants, the proposed method is highly accurate, while its computation complexity can be reduced to $O(n)$ by implementing the narrow band as an untidy priority queue [11].

7. APPENDIX

7.1. Implementation Details

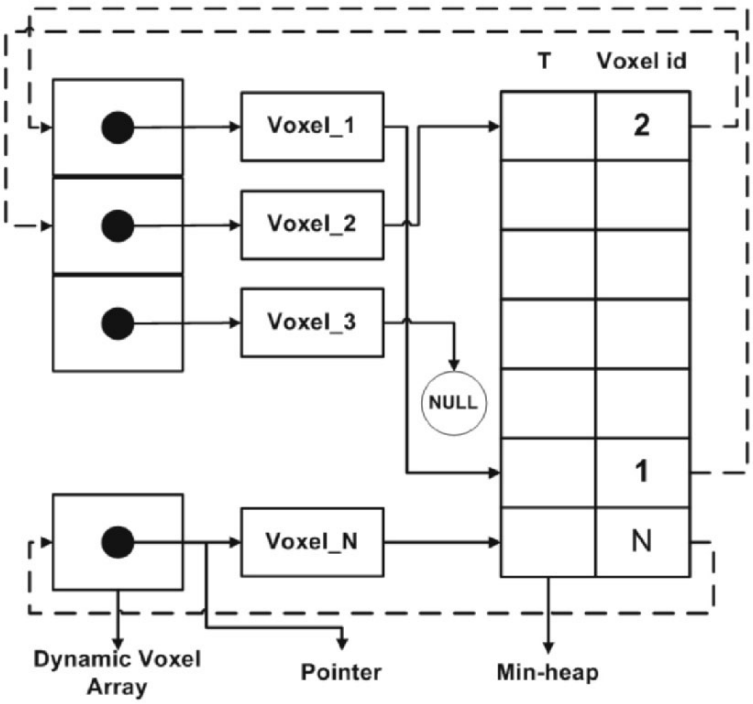
In this appendix, we present the pseudocode of the proposed multi-stencils fast marching (MSFM) method as shown in Algorithm 1. For clarity, we have presented

Algorithm 1 The Proposed MSFM Algorithm

```
VoxelArray = IdentifyBoundaryVoxels()
for i = 1 : size(VoxelArray) do
  GetNeighborDistances(VoxelArray[i])
end for
while NarrowBand.IsNotEmpty() do
  Voxel v = NarrowBand.Pop()
  GetNeighborDistances(v)
end while
```

the pseudocode for two-dimensional image space. For 3-dimensional space, the algorithm will be modified at several parts to take into account the extra stencils.

In general, the FMM involves two NarrowBand operations: searching the NarrowBand for the grid voxel v of minimum arrival time T, and finding the grid neighbors of v. It has been suggested in [5] to implement the NarrowBand using a minimum-heap data structure, which is a binary tree with the property that the value at any given node is less than or equal to the values at its children. Therefore, the root of the tree always contains the smallest element of the NarrowBand. Also, whenever a new point is inserted into the heap, the data structure takes care of resorting the tree to maintain the root as the smallest element, which can be retrieved in time $O(1)$. We have implemented the minimum-heap data structure using the *Multimap* data structure that is part of the standard template library (STL) [16].



(a)

Figure 7. The used data structure to achieve efficient implementing the MSFM method.

In Figure 7, we show the used data structure to achieve efficient implementation of the MSFM method. Each voxel of the object is represented by an object oriented *Voxel* class. The class contains several attributes of the voxel such as its coordinates, arrival time, state, flags, and others. All the voxels of the object are stored in a dynamic array. Each entry of the minimum-heap structure has two values. The first value is the minimum arrival time T, while the second value is the id of the corresponding voxel in the dynamic array. The id helps in accessing the associated voxel data, and hence fast localization of its neighbors. Each voxel has a pointer $pHeap$ that points to a heap location. If a voxel is contained in the minimum heap, then $pHeap$ is set to the address of that location; otherwise, it is set to $null$.

Algorithm 2 GetNeighborDistances(Voxel v)

```
Neighbors[8][2] = {{-1,0}, {1,0}, {0,-1}, {0,1}, {-1,-1}, {1,1}, {-1,1}, {1,-1}}
v.state = KNOWN
t = ∞
for i = 1 : 8 do
  x = v.x + Neighbors[i][1]
  y = v.y + Neighbors[i][2]
  Voxel v_n = GetVoxel(x,y)
  if ((v_n.state == NARROWBAND) ∨ (v_n.state == FAR)) then
    t = min(t, SolveEikonal(v_n, S_1))
    t = min(t, SolveEikonal(v_n, S_2))
    if (t == ∞) then
      continue
    end if
    if (v_n.state == FAR) then
      v_n.state = NARROWBAND
      NarrowBand.Insert(t, v_n.id)
    else
      NarrowBand.Update(t, v_n.id)
    end if
  end if
end for
```

Algorithm 3 SolveEikonal(Voxel v, Stencil $\mathcal{S}$, SPEED F)

```
S1_Neighbors[4][2] = {{-1,0}, {1,0}, {0,-1}, {0,1}}
S2_Neighbors[4][2] = {{-1,-1}, {1,1}, {-1,1},{1,-1}}
a = 0, b = 0, c = 0
for j = 1 : 2 do
  T1 = T2 = ∞
  for i = 1 : 2 do
    if (S == S1) then
      x = v.x + S1_Neighbors[i + 2j][1]
      y = v.y + S1_Neighbors[i + 2j][2]
    else
      x = v.x + S2_Neighbors[i + 2j][1]
      y = v.y + S2_Neighbors[i + 2j][2]
    end if
    Voxel v_n1 = GetVoxel(x,y)
    if (v_n1.state == KNOWN) then
      if (v_n1.T < T1) then
        T1 = v_n1.T
        if (S == S1) then
          x = v.x + 2×S1_Neighbors[i + 2j][1]
          y = v.y + 2×S1_Neighbors[i + 2j][2]
        else
          x = v.x + 2×S2_Neighbors[i + 2j][1]
          y = v.y + 2×S2_Neighbors[i + 2j][2]
        end if
        Voxel v_n2 = GetVoxel(x,y)
        if ((v_n2.state == KNOWN) ∧ (v_n2.T ≤ T1) ∧ (NumScheme == SecondOrder))
        then
          T2 = v_n2.T
        else
          T2 = ∞
        end if
      end if
    end if
  end for
  GetCoefficients(S, T1, T2, a, b, c)
end for
return SolveQuadratic(v, T1, T2)
```

Algorithm 4 GetCoefficients(Stencil $\mathcal{S}$, T_1, T_2, a, b, c)

if ($T_2 \neq \infty$) **then**
  T_v = ($4T_1 - T_2$) / 3
  if ($\mathcal{S} == \mathcal{S}_1$) **then**
    ratio = 9 / ($4 \times h^2$)
  else
    ratio = 9 / ($8 \times h^2$)
  end if
  a = a + ratio
  b = b - 2$\times$ ratio $\times$ T_v
  c = c + ratio$\times T_v^2$
else
  if ($\mathcal{S} == \mathcal{S}_1$) **then**
    ratio = 1 / h^2
  else
    ratio = 1 /($2 \times h^2$)
  end if
  a = a + ratio
  b = b - 2$\times$ ratio $\times$ T_1
  c = c + ratio$\times T_1^2$
end if

Algorithm 5 SolveQuadratic(v, T_1, T_2)

F = ComputeSpeed(v)
c = c - $\left(1/F^2\right)$
if ((a == 0) $\wedge$ (b==0)) **then**
  return ∞
end if
disc = b^2-4$\times$a$\times$c
if (disc $<$ 0) **then**
  return ∞
else
  t_1 = (-b + sqrt(disc))/($2 \times a$)
  t_2 = (-b - sqrt(disc))/($2 \times a$)
  return max(t_1, t_2)
end if

8. REFERENCES

1. Cerveny V. 1985. Ray synthetic seismograms for complex two- and three-dimensional structures. *J Geophys* **58**:2–26.
2. Vidale J. 1990. Finite-difference calculation of travel times in three dimensions. *J Geophys* **55**:521–526.
3. van Trier J, Symes W. 1991. Upwind finite-difference calculation of travel times. *J Geophys* **56**:812–821.
4. Podvin P, Lecomte I. 1991. Finite-difference computation of travel times in very contrasted velocity models: a massively parallel approach and its associated tools. *Geophys J Int* 105:271–284.
5. Adalsteinsson D, Sethian J. 1995. A fast level set method for propagating interfaces. *J Comput Phys* **118**:269–277.
6. Kim S. 1999. Eno-dno-ps: a stable, second-order accuracy eikonal solver. *Soc Explor Geophys* **69**:1747–1750.
7. Sethian J. 1999. *Level Sets methods and fast marching methods*, 2nd ed. Cambridge: Cambridge UP.
8. Kimmel R, Sethian J. 1998. Fast marching methods on triangulated domains. *PNAS* **95**(11):8341–8435.
9. Sethian JA, Vladimirsky A. 2000. Fast methods for the eikonal and related Hamilton-Jacobi equations on unstructured meshes. *PNAS* **97**(11):5699–5703.
10. Kim S. 2001. An o(n) level set method for eikonal equations. *SIAM J Sci Comput* **22**(6):2178–2193.
11. Yatziv L, Bartesaghi A, Sapiro G. 2006. A fast o(n) implementation of the fast marching algorithm. *J Comput Phys* **212**:393–399.
12. Danielsson P-EE, Lin Q. 2003. A modified fast marching method. In *Proceedings of the 13th Scandinavian conference, SCIA 2003*. *Lecture notes in computer science*, Vol. 2749, pp. 1154–1161. Berlin: Springer.
13. Tsitsiklis J. 1995. Efficient algorithms for globally optimal trajectories. *IEEE Trans Auto Control* **40(9)**:1528–1538.
14. Godunov S. 1959. Finite-difference method for numerical computation of discontinuous solutions of the equations of fluid dynamics. *Mat Sbornik* **47**:271–306. [Trans. from the Russian by I Bohachevsky.]
15. Dijkstra EW. 1959. A note on two problems in connexion with graphs. *Numer Mat* **1**:269–271.
16. http://www.sgi.com/tech/stl/.

9

TOWARD CONSISTENTLY BEHAVING DEFORMABLE MODELS FOR IMPROVED AUTOMATION IN IMAGE SEGMENTATION

Rongxin Li and Sébastien Ourselin

BioMedIA Lab, Autonomous Systems Laboratory
CSIRO ICT Centre, Marsfield
New South Wales 2122, Australia

Deformable models are a powerful approach to medical image segmentation. However, currently the behavior of a deformable model is highly dependent on its initialization and parameter settings. This is an obstacle to robust automatic or near-automatic segmentation. A generic approach to reducing this dependency is introduced in the present chapter based on topographic distance transforms from manually or automatically placed markers. This approach utilizes object and background differentiation through watershed theories. The implementation is based on efficient numerical methods such as the Fast Marching method and non-iterative reconstruction-by-erosion. Further extension into a multi-region coupled segmentation approach is discussed. Validation experiments are presented to demonstrate the capabilities of this approach. A preliminary application in pediatric dosimetry research is described. It is believed that the more consistent behavior will enable a higher degree of automation for segmentation employing deformable models and is particularly suited for applications that involve segmentation-based construction of organ models from image databases, especially in situations where the markers can be placed automatically based on a priori knowledge.

1. INTRODUCTION

Deformable models are a powerful approach to image analysis and are used particularly widely to segment pathological and normal anatomical structures from medical images. They include parametric deformable models (PDMs) [1, 2],

Address all correspondence to: Rongxin Li, ICT Centre, Building E6B, Macquarie University Campus, North Ryde, NSW 2113, Australia. Phone: (612) 9325 3127, Fax: (612) 9325 3200. Ron.Li@csiro.au.

geometric deformable models (GDMs) [3, 4], and statistical point distribution models (e.g., [5, 6]). PDMs are characterized by their parameterized representation of contours or surfaces in a Lagrangian framework. In comparison, GDMs are based on the theory of front evolution and represented implicitly as level sets embedded in higher-dimensional functions. Numerical computation of the evolution of the GDM is carried out with the Eulerian approach, that is, no parameterization of the model on a fixed Cartesian grid is performed.

Deformable models derived from the framework of functional minimization have become widely used. The functional to be minimized is often referred to as the "energy," which varies with the model's geometry and either regional consistency, image gradients, or both. Although some models such as the generic GDM were developed independently of energy minimization, those derived from an energy minimization framework have been the most fruitful in image segmentation. Since it is generally not possible to perform global optimization for energy functionals (except in restricted circumstances such as when the functional is inherently one dimensional [7]), a local minimum is usually the only achievable outcome with these energy-based models. That is, in order to make such optimization computationally tractable, algorithms applicable to multivariate energy minimization find the nearest local minimum in a functional space composed of all permissible hypersurfaces. This local minimum depends on the initial position of the model in the space of all permissible models. This is called *initialization dependency*.

Furthermore, the energy minimum and the model geometry that corresponds to that minimum are dependent on the parameters. This is referred to as *parameter dependency*. For example, in the case of GDMs, the strength of the unidirectional propagation force and the number of iterations can play a vital role in the final outcome. Too few iterations or a too-weak propagation force can give rise to a partially segmented object, while too many iterations combined with a comparatively strong propagation term may result in so-called "leakage" (i.e., invasion of the zero level set into the background) through the relatively weak boundary gradients of the object, particularly if these weak parts are also close to the initial zero level set. This is the underlying reason that it is often necessary to monitor the progress of segmentation using a GDM. A similar problem exists with a PDM that incorporates an inflation or deflation force (also called a balloon force).

The initialization and parameter dependencies are an obstacle to robust automatic or near-automatic segmentation as image-specific, individualized initialization and parameter tuning are necessary to achieve the desired segmentation. This sensitivity to initialization, although varying in degree for different types of models, has in general been a significant limitation under many circumstances.

We take segmentation-based anatomical-model construction as an example. Anatomical models derived from medical images or image databases are needed for some medical research (e.g., [8]). For instance, our own current research is on accurate estimation of the amount of radiation energy deposited in various tissues within the body of a pediatric patient during a radiological procedure. Accurate

age-specific models of children for radiation dosimetry purposes are not presently available. It is hoped that this work will lead to a scientific basis for determining an optimal dose for children. Due to the generally large size of the databases, a high degree of automation is often needed in this process. The requirement for image-specific initialization and parameter tuning would prevent such automation from being achieved. In fact, for a gradient rich image it is usually non-trivial to find a set of parameters that sufficiently constrain the model yet are able to avoid over-segmentation. Further, segmentation failure can also result from poor initialization. This can occur, for example, when the initial implicit model lies on both sides of the object boundary.

Despite the limitations, however, the deformable models approach remains a powerful one for image segmentation. In this chapter, we present a generic marker-based approach aimed at reducing limitations in order to achieve a higher degree of automation for deformable model-based segmentation, especially in scenarios where the markers can be automatically placed, using a morphological or a knowledge-based method. This approach is presented after a brief review of the literature in Section 2, and can be summarized as follows.

The approach described here consists of the following steps given two automatically or manually placed markers. First, a gradient magnitude image is obtained after the original image is smoothed. This image is then modified through a Swamping Transform before geodesic topographic distances (GTDs) from the markers are computed. The background theory and computational considerations in this step is presented in Section 3. The second step is partitioning of the image based on the GTD transforms via thresholding and computation of distance gradient vectors. This is given in Section 5. Finally, the distance gradient vectors along with the partitioning information is used within parametric and geometric deformable models, as described in Section 6.

After the approach is described, the results of its validation experiments are discussed in Sections 7 and 8. An application to the determination of tissue characteristics in early childhood is presented in Section 9. Finally, some discussions and conclusions are presented.

2. BACKGROUND

Image segmentation using deformable models derived from an energy minimization framework is achieved via a process of model optimization. If the energy functional is inherently one dimensional, global optimality can be achieved via discrete decision or graph search-based techniques. A classic example is the widely used dynamic programming algorithm for parametric models first proposed by Amini et al. [7]. Global minimization of higher-dimensional energy functionals, however, is generally not possible. Moreover, even when the global minimum is achievable, the corresponding model is still dependent on the parameters of

the model, albeit to a lesser extent [7]. What is attainable, in general, is a local minimum of the energy functional via a variational framework. As the energy functional is usually non-convex, the local minimum achieved is likely to be different from the global minimum. This is the cause of parameter and initialization dependency.

First-generation deformable models [1] were particularly sensitive to initialization due to a nearby local energy minimum that corresponds to a totally collapsed version of the model. Early attempts to improve this include a balloon model [9], which applies either a constant or a gradient-adaptive force in the direction of the contour or surface normal. Based on similar principles, a curvature-independent propagation term,

$$g(|\nabla Z(x)|)|\nabla u|,$$

has been incorporated in the geodesic active contour (GAC) level set models[1], where $Z(x)$ is the image intensity, g is a monotonically decreasing function such that $g : R^+ \rightarrow (0, 1]$, as $r \rightarrow \infty$, and u is the level set function. The GAC essentially requires that the deformable model be completely inside or outside the structure of interest in order to ensure success [10], as is the case with the balloon model. Furthermore, as observed by numerous researchers (e.g., [11]), leakage through weak object boundaries may happen due to factors such as initialization and inappropriate weighting and stopping parameters.

Another widely employed approach is to modify the external force in deformable models. The gradient vector flow (GVF) method [13, 12, 10] is perhaps the most prominent example, which uses spatial diffusion of the gradient of an edge-strength map derived from the original image to supply an external force. A GVF is a vector field $\mathbf{v}(x, y) = [u(x, y), v(x, y)]$ that minimizes the energy functional

$$E = \iint \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2|\mathbf{v} - \nabla f|^2 dxdy,$$

where f is an edge map of the original image and μ is a blending parameter. This technique enables gradient forces to be extended from the boundary of the object and has an improved ability in dealing with concavities over using forces based on distances from edges [9, 12]. A major drawback of this approach, however, is that it does not discriminate between target and irrelevant edges. Such irrelevant gradients are abundant in a typical medical image. Consequently, the "effective range" of the object gradients for attracting the deformable model is only extended to the nearest non-object gradients. If the model is initialized beyond this range, an undesirable outcome attributable to the non-object gradients is likely to result.

A third approach is hybrid segmentation, where independent analysis such as thresholding, seeded region growing, a watershed transform, or a Markov Random Field or Gibbs Prior model, is employed to provide an initial segmentation, followed by a refinement stage using a deformable model (e.g., [14–16]). In this

approach, the deformable model may only be a secondary procedure used merely to refine the primary segmentation in the hope to gain more robustness to noise or other degradation. The individual components in this hybrid approach may be repeated to form an iterative process [15].

Through simultaneous statistical modeling of background regions, a region competition approach originated from Zhu and Yuille [17] offers a mechanism for PDMs to overcome the leakage problem, which was briefly described in Section 1 above. This mechanism, however, is not always suitable. The difficulties in the statistical modeling and in controlling the number of resultant regions are two issues in many situations. On the other hand, some subsequent efforts have successfully extended the mechanism for use with other frameworks such as the GDM. In [11], the authors employ posterior probability maps of the tumor and the background derived from fitting Gaussian and Gamma distributions to the difference image obtained from a pair of pre- and post-contrast T1-weighted images. The difference between those probabilities is used to modulate the propagation of the GDM, so that the zero level set is more likely to expand within the area of the tumor and contract in the background. The main disadvantages with this method are twofold. One is that the probability maps need to be derived using ad-hoc methods, making the application to other problems non-trivial. The other is that the usefulness of the resultant speed term is critically dependent on the accuracy of the probability estimation.

Various ad-hoc techniques have also been used in different deformable models. For example, problem-dependent search methods have been used in a preprocessing stage [18]. A robust generic approach to relaxing initialization and parameter dependencies, however, has not yet been available.

Despite its limitations, the deformable models approach provides a powerful means to use a priori knowledge for image segmentation. Therefore, many other approaches have attempted to incorporate some mechanisms of deformable models. For example, it has been found that the smoothness-imposing property of PDMs can be used to improve segmentation results using a watershed transform [19], although the watershed framework can be restrictive as to the kind of constraints or knowledge that can be used and the manner that it is used. As another example, in the classic algorithm proposed by Tek and Kimia [20], regions start to grow under image forces from a number of seeds, and boundaries emerge as the growth and merging of regions are inhibited by the presence of edges.

The approach to be presented in this chapter is within the framework of deformable models. Its purpose is to reduce sensitivity to initialization and to parameters, in order to achieve more robust automatic segmentation.

We start by examining a geometric image partitioning algorithm, namely, Skeleton by Influence Zones (SKIZ, alternatively known as a generalized Voronoi diagram).

3. SKELETON BY INFLUENCE ZONES BASED ON TOPOGRAPHIC DISTANCE

Given k sets of connected voxels in a domain D

$$\{S_i \subset D : i \in I\}$$

as the partition seeds (commonly referred to as markers in the terminology of Mathematical Morphology), where $I = \{1, 2, \ldots, k\}$, and a measure $\delta(x, S_i)$ that defines the distance in D between $x \in D$ and $\boldsymbol{S}_i$, a SKIZ is defined as [21, 22]

$$D \setminus \bigcup_i Z_i,$$

where Z_i is the influence zone of $\boldsymbol{S}_i$, defined as

$$Z_i = \{\boldsymbol{x} \in D : \forall j \in I \setminus \{i\}[\delta(\boldsymbol{x}, \boldsymbol{S}_i) < \delta(\boldsymbol{x}, \boldsymbol{S}_j)]\}.$$

A relationship exists between the SKIZ partitioning and the watershed transform if the GTD in a gradient image is used as the basis in defining $\delta(\boldsymbol{x}, \boldsymbol{S}_i)$ [22, 21, 19]. Suppose that a gradient magnitude image f, obtained after smoothing the original image $Z(\boldsymbol{x})$ via anisotropic diffusion or using an infinite impulsional response (IIR) filter that approximates a convolution with the derivative of the Gaussian kernel, is a C^2 real function.

The GTD $d(\boldsymbol{x}, \boldsymbol{S}_i)$ between $\boldsymbol{x} \in D$ and $\boldsymbol{S}_i$ is defined as

$$d(\boldsymbol{x}, \boldsymbol{S}_i) = \inf_{\boldsymbol{y} \in \boldsymbol{S}_i} d'(\boldsymbol{x}, \boldsymbol{y}), \tag{1}$$

where

$$d'(\boldsymbol{x}, \boldsymbol{y}) = \inf_{\gamma \in \{\Gamma(\boldsymbol{x}, \boldsymbol{y})\}} \int_\gamma |\nabla f(s)| ds, \tag{2}$$

where $\Gamma(\boldsymbol{x}, \boldsymbol{y})$ is the set of all paths from $\boldsymbol{x}$ to $\boldsymbol{y}$. Suppose that

$$\{\boldsymbol{S}_i : i \in I\}$$

are equal to the set of all the local minima of f. A SKIZ partition with respect to the distance definition $\forall i \in I$,

$$\delta(\boldsymbol{x}, \boldsymbol{S}_i) = f(\boldsymbol{s}_i^c) + d(\boldsymbol{x}, \boldsymbol{S}_i),$$

where $\boldsymbol{s}_i^c \in \boldsymbol{S}_i$, is equivalent to the classical watershed transform of f [21, 22] using $\{\boldsymbol{S}_i : i \in I\}$.

This has been used in the metric-based definition of the watershed transform [22, 23], and is the basis of partial differential equation (PDE) models of the watershed [19, 24], which have been exploited to incorporate smoothness into watershed segmentation. As the watershed theory has established that a watershed coincides with the greatest gradient magnitude on a GTD-based geodesic path between two markers, so does the corresponding part of the SKIZ [19, 22].

It is critical to note that in 2D or 3D, spatially isolated disjoint gradients cannot be on a GTD-based geodesic path; therefore, the gradients need to have spatially consistently large magnitudes in order to be part of the SKIZ.

It follows from the above that in order to take advantage of the significant edges corresponding to the classical definition of watersheds, the markers from which the GTDs are calculated to derive the SKIZ must be at local minima in the image and there must not be other minima. Homotopy modification via a swamping transform [22, 25, 26] is usually necessary to achieve this. This transform produces an "uphill" landscape such that when moving away from a marker the gray levels either rise up or keep constant until the SKIZ is reached (i.e., the border of the marker's influence zone). Thus, local minima that are marked will be kept but those that do not contain markers will be filled.

Defining

$$f_1(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \bigcup\limits_i \boldsymbol{S}_i \\ f(\boldsymbol{x}), & \text{otherwise} \end{cases},$$

and

$$b(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \bigcup\limits_i \boldsymbol{S}_i \\ M_b & \text{otherwise} \end{cases},$$

where the constant $M_b = \sup\{f(\boldsymbol{x}) : \boldsymbol{x} \in D\}$, the swamping transform, which results in a new image $f_2(\boldsymbol{x})$, is accomplished by a reconstruction-by-erosion of $f_1(\boldsymbol{x})$ from $b(\boldsymbol{x})$ [27]:

$$f_2(\boldsymbol{x}) = R_{f_1(\boldsymbol{x})}[b(\boldsymbol{x})].$$

Traditionally the swamping transform is accomplished by the iterative geodesic erosion (until stability) of b with f_1 as the reference [28, 29], or

$$\varepsilon^{\infty}_{f_1(\boldsymbol{x})}[b(\boldsymbol{x})].$$

This can be computed in the following way [27, 30]:

$$b_k = \varepsilon(b_{k-1}) \vee f_1, \tag{3}$$

where ε represents an erosion (of size 1), $k = 1, 2, 3, ..., k \to \infty$, and

$$b_0 = b \vee f_1.$$

Thus implemented, the iterations will finally lead to

$$f_2(\boldsymbol{x}) = b_\infty.$$

4. COMPUTING THE GTD TRANSFORM AND SWAMPING TRANSFORM

The computation of the GTD tranform and the swamping transform are key elements to the proposed approach. Unfortunately, they can also be the most computationally costly components. It is therefore critical to investigate efficient, yet accurate, algorithms for those transforms.

4.1. Wave Front Propagation and GTD transform

As the GTD $d(\boldsymbol{x}, \boldsymbol{S}_i)$ is a path integral of the gradient magnitude in the relief image, it is equivalent to the spatial distance weighted by the elevation rate per spatial unit.

On the other hand, examination of Eq. (2) reveals that $d(\boldsymbol{x}, \boldsymbol{S}_i)$ can be viewed as the infimal integral of time spent on a path between $\boldsymbol{x}$ and $\boldsymbol{S}_i$ at a speed that is the inverse of $|\nabla f(\boldsymbol{x})|$ (or its modified version $|\nabla f_2(\boldsymbol{x})|$). This minimum time is the time taken for a propagating wavefront traveling at that speed to arrive at $\boldsymbol{S}_i$ from $\boldsymbol{x}$ [27, 31].

It has long been established that, provided that the direction of the propagation does not change, the arrival time, $T(\boldsymbol{x}, \boldsymbol{S}_i)$, of a wavefront starting from $\boldsymbol{S}_i$ is governed by the following eikonal equation:

$$\begin{aligned} |\nabla T(\boldsymbol{x}, \boldsymbol{S}_i)| &= \tfrac{1}{q(\boldsymbol{x})}, \quad \boldsymbol{x} \notin \boldsymbol{S}_i, \\ T(\boldsymbol{x}, \boldsymbol{S}_i) &= 0, \qquad \boldsymbol{x} \in \boldsymbol{S}_i, \end{aligned} \tag{4}$$

where $q(\boldsymbol{x})$ is the speed (assumed to be time-invariant) of the propagation. In order to have $d(\boldsymbol{x},\boldsymbol{S}_i) = T(\boldsymbol{x},\boldsymbol{S}_i)$, $q(\boldsymbol{x})$ should be defined as

$$q(\boldsymbol{x}) = \frac{1}{|\nabla f_2(\boldsymbol{x})|}. \tag{5}$$

Based on the Hyperbolic Conservation Laws, Sethian et al. developed efficient, entropy-satisfying numerical methods to solve the eikonal PDE when the speed $q(\boldsymbol{x}) > 0$ for all $\boldsymbol{x} \in D$ [32–34], known as the Fast Marching Method (FMM). This has since become a standard numerical algorithm for efficiently tracking a propagating front by progressively solving for the field of arrival times, or the propagation map.

Compared with alternative algorithms [22, 23, 35], the advantages of FMMs include that they result in isotropic distance propagation [31], and lead to an accuracy that is not limited by discretization of the image [33, 34]. In fact, an FMM can yield a solution that is close to the ideal [31]. A high level of efficiency, with a computational time of $O(N \log N)$ when using a binary heap data structure and only performing calculation within a narrow band is an additional advantage.

An issue related to the GTD transform is that it is possible that not all of an object's boundary corresponds to the strongest gradients on a geodesic path

between the target marker and the background marker. For a higher-dimensional space such as 3D, there exists a stronger possibility that this may happen if the object of segmentation does not always contrast well with neighboring organs. In such a case, not all of the object's boundary coincides with SKIZ. In order to rectify this, it may be desirable to selectively use a range of $f_2(\boldsymbol{x})$, using a mapping such as

$$\bar{f}_2(\boldsymbol{x}) = \begin{cases} L_o, & f_2(\boldsymbol{x}) \le L_i \\ L_o + \frac{f_2(\boldsymbol{x}) - L_i}{H_i - L_i}(H_o - L_o), & L_i < f_2(\boldsymbol{x}) < H_i \\ H_o, & f_2(\boldsymbol{x}) \ge H_i \end{cases}, \tag{6}$$

where L_i, H_i, L_o, and H_o are constant parameters, with the range between L_i and H_i designating the desired band for $f_2(\boldsymbol{x})$.

Alternatively, the speed function for the FMM can be directly manipulated using a combination of a band selection and a linear approximation to ensure a significant speed reduction at the object's boundary. For example,

$$\bar{q}(\boldsymbol{x}) = \begin{cases} H_q, & |\nabla f_2(\boldsymbol{x})| \le L_f \\ H_q - \frac{|\nabla f_2(\boldsymbol{x})| - L_f}{H_f - L_f}(H_q - L_q), & L_f < |\nabla f_2| < H_f \\ L_q, & |\nabla f_2(\boldsymbol{x})| \ge H_f \end{cases} \tag{7}$$

may be used, where L_f, H_f, L_q, and H_q are also constant parameters. If this formulation is used, H_f needs to be adjusted to suit different tasks and different sources of data, while the other three can remain application independent. The parameter H_q should be large enough to ensure a near-zero GTD on any topographically flat path of a plausible length between any two points in the image. Conversely, H_q should be small. In our experiments, we have consistently used 0.001 for H_q and have chosen $L_f = 0$.

4.2. Swamping Transform

Notwithstanding the different degrees of efficiency, a distance transform can be accomplished by choosing from among a number of different algorithms the one best suited to the given circumstances. The same cannot be said of the swamping transform introduced in Section 3. To date, the only algorithm used for a swamping transform has been the iterative geodesic erosion outlined in Section 3. One of its major drawbacks is that it is computationally expensive and would add a considerable burden to the approach being presented. Moreover, certain types of structures in the image can impact the speed of execution so much that no guarantees can be given as to the processing time required in the general case [36].

Recently, an efficient method for reconstruction-by-dilation was proposed by Robinson and Whelan [36]. Reconstruction-by-dilation is opposite to the operation required here. Two algorithms were proposed. Based on the observation that

gray levels in images produced from reconstruction-by-dilation monotonically decrease with the distance from the nonzero pixels in the marker image, processing starts from the list of pixels of the highest gray level in the marker image and proceeds toward the lowest end. Although for either algorithm there are additional conditions that must be satisfied, those conditions are not so restrictive as to substantially limit their applicability. For example, the general-case algorithm merely requires that the marker image be pixel-wise less than or equal to the mask image.

An adapted version of Robinson and Whelan's method is used in our system for the novel purpose of computing the swamping transform. It performs reconstruction-by-erosion by starting from the voxels of the lowest gray level in the marker image and moving toward higher voxel values. The conditions required of the images need to change accordingly (it is easy to observe that b_0 and f_1 satisfy these conditions). The main advantage of this approach is its efficiency, as no iteration is needed and no voxel is processed twice.

5. IMAGE PARTITIONING BASED ON GTD TRANSFORMS

5.1. Dual Marking Method

First, we discuss the scenario that markers are provided to identify both the target and the background. They consist of one placed interior to the boundary of the target organ and another more external to the target. More than one background marker is usually not necessary but can result in more robustness where the image is complex, as is demonstrated later. In the text below, we assume that there is only one object marker, but that possibly more than one background marker is used.

Without loss of generality, suppose that $\boldsymbol{S}_1$ is placed within the target of segmentation, and

$$\{\boldsymbol{S}_j : j \in \{2, 3, \ldots, K\}\}$$

are placed outside in the background. A maximum difference image M is computed as follows:

$$M(\boldsymbol{x}) = \max_{j \in \{2,\ldots,k\}} \{\delta(\boldsymbol{x}, \boldsymbol{S}_1) - \delta(\boldsymbol{x}, \boldsymbol{S}_j)\}. \tag{8}$$

This equation is designed to ensure that the background marker whose GTD to the object marker is the least overrides any other markers. Eq. (8) can be implemented very efficiently via the FMM. As

$$f_2(\boldsymbol{s}_i^c) = 0, \forall i \in \{2, \ldots, k\},$$

it can be rewritten as

$$M(\boldsymbol{x}) = d(\boldsymbol{x}, \boldsymbol{S}_1) - \min_{j \in \{2,\ldots,k\}} \{d(\boldsymbol{x}, \boldsymbol{S}_j)\}, \tag{9}$$

in which the second term on the right-hand side lends itself naturally to multiple-front wave propagation via the FMM, where only one round of propagation is necessary.

A binary image can be obtained by thresholding M:

$$B_0(\boldsymbol{x}) = \begin{cases} 1, & M(\boldsymbol{x}) \leq 0, \\ 0, & \text{otherwise} \end{cases}. \tag{10}$$

Due to degradation or deficiencies in the boundary gradients that are often present, and a lack of model constraints (e.g., smoothness, shape constraints) to overcome these deficiencies, B_0 is unlikely to be an optimal segmentation of the target in most situations. However, it can provide valuable information for deformable models. Different possibilities exist in how this information can be exploited by a deformable model. A transform of this information into forces or speeds would allow full integration into a deformable model framework. Advantages of doing so include that isolated boundary gradients near SKIZ, while not playing a role in defining B_0, can be additionally taken into account.

In order to achieve this, we compute a distance map D_0 on image B_0, i.e.,

$$D_0(\boldsymbol{x}) = Æ(B_0),$$

where AE is a Euclidean distance transform. A complementary set of computation follows, namely,

$$B_1(\boldsymbol{x}) = \overline{B_0(\boldsymbol{x})},$$
$$D_1(\boldsymbol{x}) = Æ(B_1).$$

5.2. Single Marking Alternative

Under some circumstances it may be more desirable if a single marker is sufficient for the segmentation task, as this would mean significantly simplified identification requirement. This may be a crucial advantage in applications that need fully automatic initialization. If any internal gradients present within the target are known to be isolated (e.g., those due to imaging noise), or are significantly weaker compared with those at the object boundary, the difference between the GTDs on either side of the boundary should be large enough for a separating threshold to be easily found. In such a case, a binary image B_0 can be obtained by

$$B_0(\boldsymbol{x}) = \begin{cases} 1, & \delta_0(\boldsymbol{x}) \leq \eta, \\ 0, & \text{otherwise}, \end{cases} \tag{11}$$

where η is an application-dependent constant. D_0, B_1, and D_1 can be calculated similarly as with the methods presented above. However, a disadvantage is that this alternative approach introduces an additional parameter, η, which needs to be

adjusted to suit each specific segmentation task. In addition, it should not be used where the gradient composition interior to the target may be significant and cannot be estimated a priori.

6. INTEGRATION INTO DEFORMABLE MODELS

6.1. Parametric Active Surface Model

For a PDM, we define the following force field, assuming that the central differencing discretization scheme is applied to the spatial gradients $|\nabla D_0|$ and $|\nabla D_1|$:

$$\mathbf{F} = \begin{cases} -c\nabla D_0, & D_0 \geq 2 \textit{ and } |\nabla D_0| \geq 1, \\ -c\nabla D_1, & D_1 \geq 2 \textit{ and } |\nabla D_1| \geq 1, \\ \nabla f(\mathbf{x}), & \text{otherwise}, \end{cases} \tag{12}$$

where $f(\mathbf{x})$ is the same gradient magnitude image defined previously, and c is a constant parameter. While $|\nabla D_0|$ and $|\nabla D_1|$ are mostly constant, the places where they vary deserve attention. For instance, $|\nabla D_0(\mathbf{x})|$ may be zero near a narrow and elongated part of the object; conversely, $|\nabla D_1(\mathbf{x})|$ may be zero near a narrow concavity of the object. In such places, ∇f is used as a replacement. The conditionals in this equation deserve some further discussion. First, notice that

$$\{\mathbf{x} : D_0(\mathbf{x}) \geq 2 \wedge |\nabla D_0(\mathbf{x})| \geq 1\} \cap$$

$$\{\mathbf{x} : D_1(\mathbf{x}) \geq 2 \wedge |\nabla D_1(\mathbf{x})| \geq 1\} = \phi.$$

Secondly and more importantly, if ∇D_0 and ∇D_1 were to be applied to the two adjacent surfaces where $1 \leq D_0 < 2$ and $1 \leq D_1 < 2$, they might contribute to a potential risk of oscillation for the model, as they would define a force that reverses its direction between neighboring voxels. We therefore use the gradient of f for the two surfaces, where $|\nabla f|$ is small [37], reducing the likelihood of oscillation.

A remaining issue is a potential lack of external force in the normal direction of the deformable model, which may occur near major concavities. Xu et al. [12] point out that a force based on the distance to "edgels" (i.e., edge pixels, as defined and used in [9]) does not help in such a situation. A similar problem exists when using ∇D_0 and ∇D_1, even though these are not distances to edgels. A possible remedy is the use of a pressure force to compensate for the inadequacy of the external force. However, a constant or magnitude-adaptive force (such as a gradient adaptive force) is clearly unsuitable. Fortunately, due to the use of markers, the issue can be easily addressed using a directionally adaptive pressure force. As $M(\boldsymbol{x})$ is negative inside the object but positive outside, the incorporation of the sign of M,

$$\text{sgn}[M(\boldsymbol{x})],$$

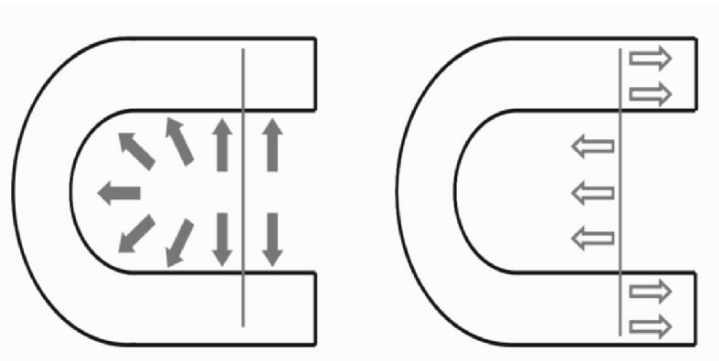


Figure 1. Left: The external force (solid arrows) may be deficient in the normal direction of the deformable model (vertical straight line) near significant concavities. Right: $\text{sgn}[M(\boldsymbol{x})]\mathbf{N}(\mathbf{x})$ (empty arrows) is a pressure force that is automatically adaptive to the need for inflation or deflation. See attached CD for color version.

into a pressure force will make the force automatically adaptive to the need for either expansion or contraction, in accordance with whether the node of the model is inside or outside the segmentation object. Thus, this force can compensate for the insufficient normal component of ∇D_0 and ∇D_1 near significant concavities, as illustrated in Figure 1.

It is worth noting, however, that it is often desirable to ignore narrow concavities, as their presence is frequently due to noise or artifacts, rather than a genuine structural feature inherent in the object. This especially tends to be true if the concavity is shallow.

F has the potential to replace the gradient image with vectors that are globally consistent, in contrast to the short ranging and inconsistent information in the gradient image.[2] A vector field such as Eq. (12) can be easily integrated into a PDM, as demonstrated in existing works with the GVF model [10, 12].

In a similar fashion to the GVF model, **F** can help drive the deformation of a PDM **v** with a surface parameterization s:

$$\frac{\partial \mathbf{v}}{\partial t} = \alpha \frac{\partial^2 \mathbf{v}}{\partial s^2} - \beta \frac{\partial^4 \mathbf{v}}{\partial s^4} + \gamma(\mathbf{F}(\mathbf{x}) \cdot \mathbf{N}(\mathbf{x}))\mathbf{N}(\mathbf{x}), \tag{13}$$

where **N** is the surface normal, and α, β, and γ are the model parameters. In our experiments, α was set to 0.

Further, as discussed above, a pressure force $\mathbf{N}(\mathbf{x})$ or $\text{sgn}[M(\boldsymbol{x})]\mathbf{N}(\mathbf{x})$ can be used in addition when it is desirable to model narrow concavities.

6.2. Geodesic Active Contour Level Set

Derived from an energy minimization framework, the GAC model is guaranteed to converge to a local minimum corresponding to a geodesic contour in a Riemannian space. It is commonly implemented using the level set method, where the movements of the implicit model consist of propagation, curvature (mean or Gaussian) motion, and advection.

There are previous proposals of using a region-based speed term as well as velocity vector fields in GAC or other level set models [10, 34, 38]. Using the previous definitions of f and M, we propose the following adapted formulation for the evolution of a level set function u:

$$\frac{\partial u}{\partial t} = \alpha h(f)|\nabla u| div\frac{\nabla u}{|\nabla u|} + \beta \text{sgn}[M(\boldsymbol{x})]|\nabla u| + \gamma \mathbf{P}\cdot\nabla u, \tag{14}$$

where $h(\cdot) : R^+ \rightarrow [0, 1)$ is a monotonically decreasing function, and α, β, and γ are the scaling parameters. $\mathbf{P}$ is an advection force and will be discussed below.

In the traditional formulation, a limitation in the application of the GAC level set model has been that in practice the model needs to be initialized completely interior or exterior to the true object boundary [10]. Further, even when this requirement is observed, there is still a risk of leakage through the relatively weak boundary gradients of the object, particularly if these are also close to the initial zero level set. This is largely due to the incorporation of unidirectional propagation. As reviewed in Section 2, the region competition approach, initially proposed in [17] for PDMs, has been adapted in attempts to overcome these limitations. For example, the authors of [11] propose the use of posterior probability maps to determine which part, if any, of the level set function is inside the object and which part is outside. This probabilistic information is then used to expand or contract the model accordingly.

Similarly, one of our major concerns is for the curvature-independent propagation of the level set to automatically, and dynamically, adapt to the need for either inward or outward propagation. Our formulation achieves this through the adaptive propagation term (the second term on the right-hand side of Eq. (14)). That is, the speed of u, $\frac{\partial u}{\partial t}$, is negative for the part that is inside the object but positive where outside.

The advection term $\mathbf{P}$ may be defined as

$$\mathbf{P} = -\mathbf{F},$$

with $\mathbf{F}$ defined in Eq.(12). The advantage of this, as in the PDM case above, is the possible incorporation of forces based on spatially disjoint gradients. As the GTD is a geodesic distance, disjoint gradients generally play no role in the distance calculation except for the spatial locations that those gradients occupy.

A simpler alternative is

$$\mathbf{P} = \nabla D_0 + \nabla D_1.$$

This alternative form is similar in effect to curvature-independent directional propagation, but is more appropriate if narrow concavities should be ignored, as discussed in Section 6.1.

Initialization of u can be performed by combining D_0 and D_1.

7. QUALITATIVE EVALUATIONS

We have conducted experiments using CT data and MR data, on segmenting tumors, the liver, the lung, the brain, and the femoral bone. In this section we present some qualitative evaluations first, which relied primarily on visual inspection. Within each group of experiments, the same set of parameters have always been applied to all the images in the group.

7.1. Tests with the PDM

Tests of segmenting the liver from a CT image of the abdomen of a pediatric patient were conducted. Two single-voxel markers were employed, one placed randomly in the liver to identify it as the target, the other outside to designate the background. When the background marker was appropriately placed (explained below), the model was able to find and segment the target despite the abundant irrelevant gradients between the initial model and the target (Figure 2), due to the globally consistent vector field providing guidance in place of image gradients. Our tests have revealed no restriction on the placement of the target marker, other than that it must be interior to the boundary profile of the target structure. On the other hand, these tests also indicate that a background marker will make a useful contribution only if it is not completely encircled by an equally strong or stronger edge than the target's boundary.

7.2. Experiments with the GDM

The first set of data used in the tests with the GDM were from the SPL and NSG Brain Tumor Segmentation Database [39], which contains 10 T1-weighted SPGR MR images of the brain with several types of tumors (meningioma, low-grade glioma, and astrocytoma). The image dimensions were $256 \times 256 \times 94$, with a voxel resolution of $0.9375 \times 0.9375 \times 1.5\ \text{mm}^3$ [39]. Segmentation of tumors was the objective of this experiment. Similar to the CT segmentation above, single-voxel markers were used, one being placed randomly inside the tumor area and the other outside.

A typical example of the segmentation, including intermediate results, is displayed in Figure 3. The visual inspection is exemplified in Figure 4.

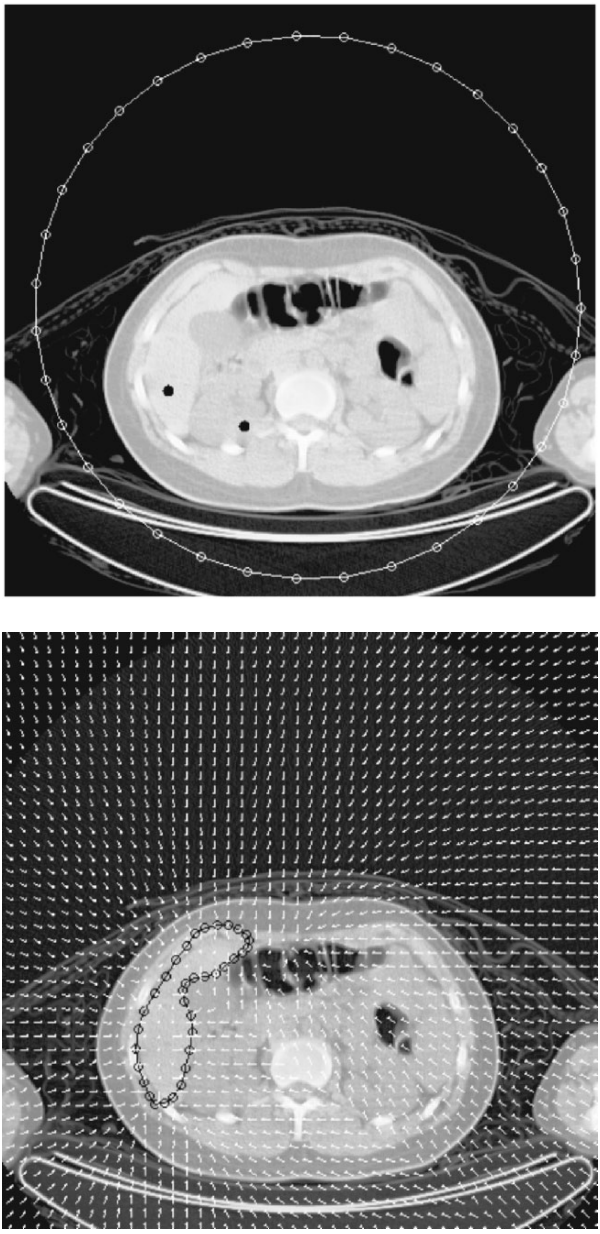


Figure 2. Experiments with a PDM on a CT image. Top: the image, the initial model (large white circle), and the two markers (black dots) that are use to respectively identify the target and non-target background. Bottom: The globally consistent external force field (white arrows) and the resultant segmentation (black contour).

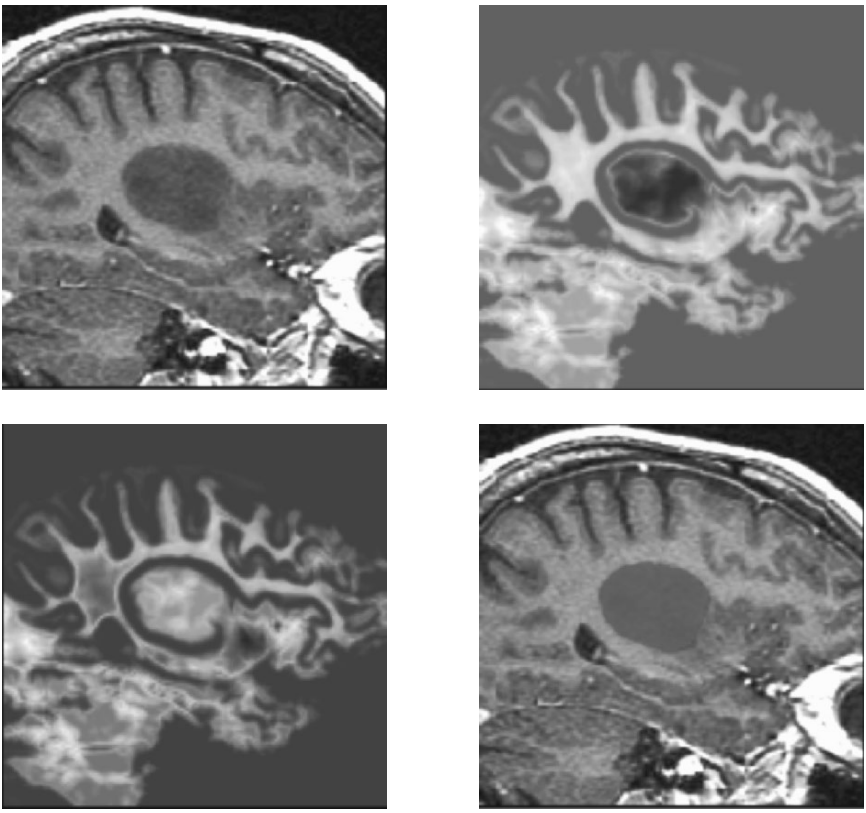


Figure 3. Brain tumor segmentation using the GAC level set model on a T1-weighted MR image. Top left: cropped original brain MR image; top right: GTD transform from the object marker; bottom left: GTD transform from the background marker; bottom right: a segmentation of the brain tumor displayed as the overlay on the original image. See attached CD for color version.

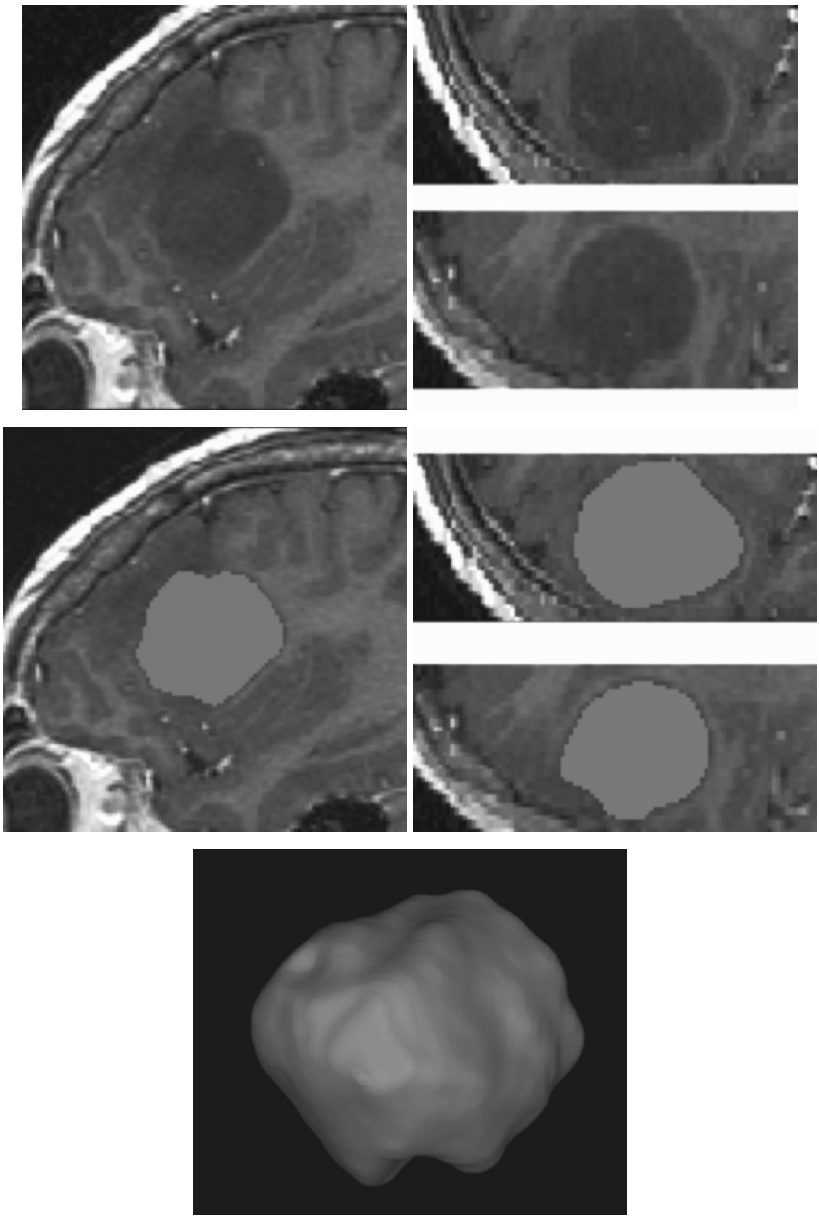


Figure 4. Example of brain tumor segmentation using the GAC level set model. Top left: axial slice view of an original T1-weighted MR image of the brain; top right: coronal (above) and sagittal (below) views of the original image; middle left: the segmentation superimposed on the axial slice view; middle right: the segmentation superimposed on coronal (above) and sagittal (below) views of the original image; bottom: a 3D view of the segmented tumor. See attached CD for color version.

Subsequently, we used other CT and MR images to further test the algorithm by segmenting the liver and lungs. An example is shown in Figure 5. In all of the tests involving tumor, liver, and lung segmentation from CT and MR images, visual inspection revealed that good results had been achieved.

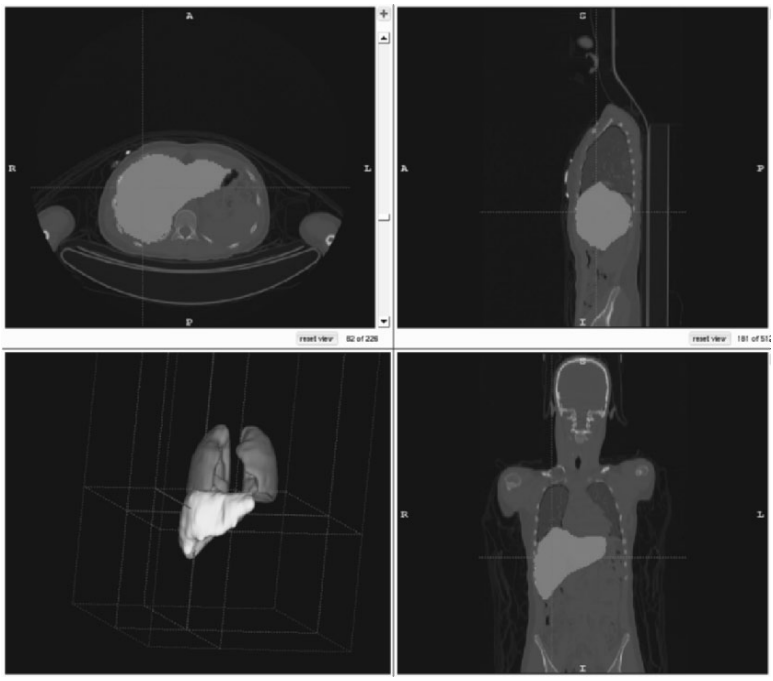


Figure 5. Example of lung and liver segmentation employing the GAC level set model on CT images, with the results displayed using ITK-SNAP software [40]. See attached CD for color version.

8. QUANTITATIVE VALIDATION

8.1. Accuracy

For quantitative assessment of the accuracy performance of the approach presented above, we used MR and CT databases to study the segmentations of the brain, the liver, and the femur. As in the qualitative tests, within each group of experiments the same set of parameters were always applied to all the images used in the group.

8.1.1. Experiments on Segmenting the Brain from MR Data

The first database is the SPL and NSG Brain Tumor Segmentation Database [41], the same as that described in Section 7.2. We investigated segmentation of the brain together with the tumors. Figure 6 demonstrates this task through segmentation of one of the images. This is a challenging task, as some of the tumors have strong gradients at their boundaries. These gradients are not only unhelpful to segmentation of the whole brain (including the tumor), but may interfere with this task. Manual segmentations by four independent experts were available on a randomly selected 2D slice in each of the ten cases in the SPL and NSG Brain Segmentation Database [41]. For this battery of tests, the "single marking" method was used. Typically, two markers[3], each comprising a single voxel, were placed within the brain to identify it as the target. That is, in order to deal with the challenge of segmenting normal tissues and tumors together despite the strong gradients surrounding the pathology, an extra marker was typically placed inside the tumor to signify that it was in fact part of the target to be segmented.

Smoothing of the original image was performed using an infinite impulsional response (IIR) filter that approximates a convolution with the derivative of the Gaussian kernel with a σ value of 1. Equation 7 was used in determining the speed of the FMM, where a value of 5.0 was used for H_f. (The values for the other parameters are as discussed in Section 4.)

For all the cases, our trials have indicated that the outcome was insensitive to placement of the markers. In fact, we failed to observe any effect of the different placements of the markers on the results.

We compared against manual segmentation of the brain (including the normal brain tissue and the tumor), which had been performed by four medical experts on one slice per image. The result of an analysis that compares our segmentation with that by each of the four experts, in terms of the sensitivity and the specificity, is shown in Table 1.

Table 1. Mean Sensitivities and Specificities and Standard Deviations with Respect to Experts' Segmentations of the Brain in 10 Cases

Expert	sensitivity	specificity
1	0.942± 0.032	0.991± 0.006
2	0.953± 0.032236	0.988± 0.006
3	0.950± 0.040	0.988± 0.006
4	0.939± 0.109	0.971± 0.017

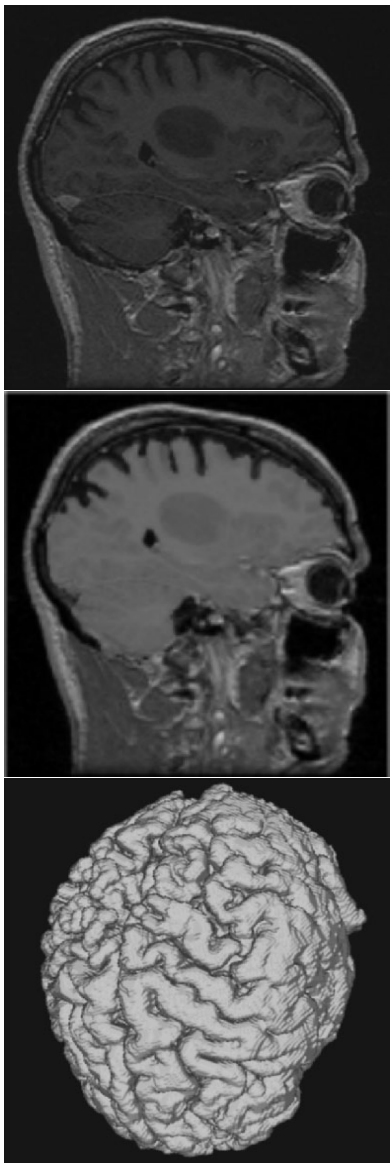


Figure 6. Example of brain segmentation using the GAC level set model. Top: an original T1-weighted MR image of the brain. Middle: a slice of the segmented brain (white overlay). Bottom: a 3D view of the segmented brain. See attached CD for color version.

8.1.2. Experiments on Segmenting the Liver from CT Data

The next database was initially acquired by another research group (Digestive Cancer Research Institute, IRCAD, France), who conducted an independent segmentation study. This database was composed of CT images of the abdomen of patients with a liver tumor. The images all have 512×512 pixels in axial slices, but with the third dimension varying. Manual delineation by independent experts was available on axial slices at the same intra-slice resolution as the original data, but only on slices that were 2 mm apart, irrespective of slice thickness in the original data.

The dual marking approach was used. The database was divided into three groups according to the apparent sources of acquisition. The six images in the first group had a uniform slice thickness of 1 mm, although their intra-slice resolutions varied from 0.5742×0.5742 mm to 0.7422×0.7422 mm. On the other hand, the second group, which contained seven images, was characterized by a slice thickness of 2 mm, with intra-slice resolutions varying from 0.6914×0.6914 mm to 0.8200×0.8200 mm.

The voxel size of the images in the third group (eight images) was the same as that in the second.

The first two groups of images were used respectively for the first two batteries of experiments. The first image in each group was used to optimize the parameter H_f based on the experimenter's visual assessment. The selected parameter was then applied to the rest of the group without change. The performance measures employed were the sensitivity (ratio of voxels correctly segmented as target to those in the ground truth segmentation) and the specificity (ratio of voxels correctly identified as background by the segmentation method to those indicated in the ground truth). The experts' manual segmentations were used as the surrogate ground truth. The mean sensitivities and specificities (both with standard deviations) achieved are shown in Table 2. As specificity can be a poor indicator of performance, we also calculated the Dice Similarity Coefficient (DSC) for these batteries.

The importance of suppressing false detection and minimizing missed tissue is not always equal. That is, in some situations it is more critical to avoid false detection, and vice versa. In optimizing the parameters, it is possible to take this into account. We used the third group of images to study the effect of doing so by conducting two batteries of experiments on the same group of data. Accordingly, the process of parameter selection was different from the previous two groups in that all the images were used in the initial visual assessment of the effect of parameter changes. After this selection process, the same set of parameters were applied to all the images used in the battery. In the Battery 3a experiments, we aimed for a specificity of approximately 0.99. The results and analysis are tabulated in Table 3. In Battery 3b, the goal was a sensitivity of 0.95, and the results are displayed in Figure 4.

Table 2. Mean Sensitivities, Specificities, and DSCs with Standard Deviations for the Liver Segmentation

Battery	Sensitivity	Specificity	DSC
1	0.950±0.044	0.990±0.007	0.914±0.033
2	0.948±0.045	0.991±0.009	0.915±0.047

Table 3. Battery 3a: Sensitivities and Specificities of Segmentation Experiments with a Target Specificity of 0.99

Case	sensitivity	specificity
1	0.908	0.993
2	0.917	0.995
3	0.916	0.990
4	0.936	0.992
5	0.943	0.994
6	0.99	0.994
7	0.910	0.994
8	0.898	0.991
Mean	0.920	0.993
Stdev	0.014	0.002

8.1.3. Experiments on Segmenting the Femoral Bone

A subset was selected from a database of T1-weighted MR of the knee, based on the criterion that the same coil type, magnetic strength, pulse repetition time, and pulse echo time had been used in acquisition of the images. The reason for selecting such a homogeneous set is to allow the same set of parameters to be used on all the images to be tested. This set of parameters came from manual optimization, with visual feedback only, using the first image.

The images were acquired using an MR acquisition protocol that consists of a single fat-suppressed 3D spoiled gradient-echo sequence in the saggital plane. The coil used was an LGEXTPA (type 3), the magnet strength was 1.5 T, the pulse repetition time (TR) 60 ms, the pulse echo time (TE) was 7 ms, and the flip angle 40°. The field of view (FOV) was 90 × 90 mm, and the acquisition matrix was 512 × 512 elements. The resultant images have a voxel size of 0.23 × 0.23 × 1.5 mm^3.

Table 4. Battery 3b: Sensitivities and Specificities of Segmentation Experiments with a Target Sensitivity of 0.95

Case	Sensitivity	Specificity
1	0.949	0.987
2	0.960	0.987
3	0.936	0.978
4	0.988	0.988
5	0.977	0.984
6	0.958	0.979
7	0.960	0.989
8	0.951	0.986
Mean	0.960	0.985
Stdev	0.016	0.004

The target of segmentation was the distal section of the femur, comprised mainly of the femoral condyles (lateral and medial). Manual segmentation by an independent expert, used as the ground truth, is available for all the slices. The resultant sensitivities and specificities are tabulated in Table 5 along with the associated statistical analysis.

Table 5. Analysis of Sensitivities and Specificities in the Femur Segmentation

Case	Sensitivity	Specificity
1	0.957	0.996
2	0.925	0.989
3	0.972	0.992
4	0.968	0.993
5	0.970	0.990
6	0.967	0.993
Mean	0.960	0.992
Stdev	0.018	0.002

8.2. Robustness

8.2.1. Experiments with the Parametric Model

The aim of this validation study was to test the robustness of the approach to placement of the initial seeds. The same CT image as described in Section 7.1 was used, the target remained the liver. The ground truth to be compared against was a manual segmentation, shown in Figure 7. Two seeds were randomly placed within the areas identified by a target mask and a background mask, respectively. These seeds were generated as follows. A foreground mask and a background mask were obtained, respectively, from the manual segmentation.

Denoting the binary image resulting from the manual segmentation as Z_b, and a circular structuring element A of 5 pixels in radius, we use

$$F = S \ominus A$$

as the foreground mask. Further defining an image Y, where

$$Y(\boldsymbol{x}) = \begin{cases} 1 & Z(\boldsymbol{x}) > t_h, Z_b(\boldsymbol{x}) = 0, \\ 0 & \text{otherwise}, \end{cases} \tag{15}$$

where t_h is a threshold, we used

$$B = Y \ominus A$$

as the background mask. A randomly generated seed $\boldsymbol{x}$ is accepted as a target marker iff $F(\boldsymbol{x}) > 0$; or as a background marker iff $B(\boldsymbol{x}) > 0$. Figure 7 shows the target and background masks used in our validation experiments on the liver.

Our tests have revealed no restriction on placement of the target marker, other than that it must be interior to the boundary profile of the target structure. On the other hand, our tests also indicate that a background marker will make a useful contribution only if it is not completely encircled by an equally strong or stronger edge than the target's boundary (such as the vertebra in this case). Examples of where "useful" background markers can be placed for segmentation of the liver in the image are indicated by the white dots in Figure 8.

For automatic marker placement, one possibility to ensure correct placement of a background marker is via intensity and neighborhood tests. Another may be to employ more restrictive thresholding, such as double thresholding.

8.2.2. Experiments with the GAC Model

A second validation study was conducted on a T1-weighted MR image from the SPL and NSG Brain Tumor Segmentation Database, with the target being a tumor. Since the maximum difference operation used in computing the image M means that a single well-placed background marker is sufficient to achieve the

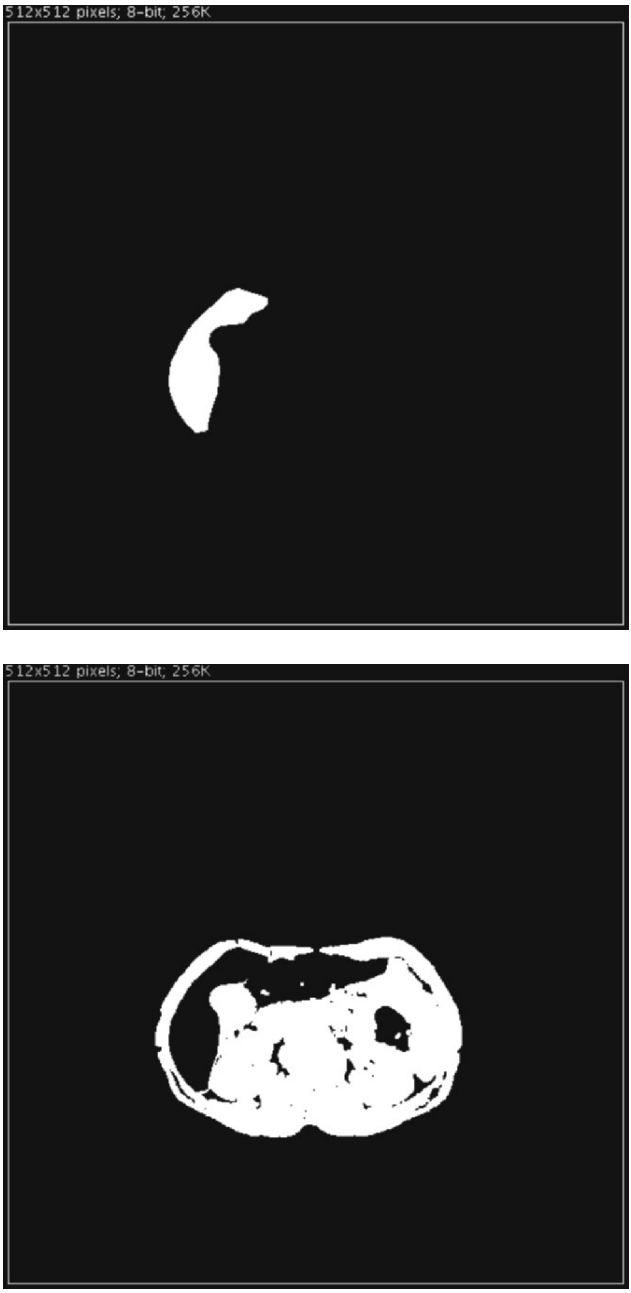


Figure 7. Top: manual segmentation used as the ground truth. Bottom: background mask in which a background identifying marker is randomly placed.

Figure 8. Examples showing where a potential background marker should be placed in order for it to make a useful contribution to segmentation of the liver. That is, it should be put in one of the places indicated by the white dots.

desired outcome, we hypothesized that the likelihood of success would increase with the number of randomly placed markers, even if no tests were performed regarding their placement. To verify this, we used the first 3D image from the SPL and NSG Brain Tumor Segmentation Database, with the target still being a tumor. In Figure 9 we show the failure rates against the number background seeds used. These seeds were accepted by a background mask after being randomly generated. The apparent anomaly when 3, 4, and 5 seeds were used can be understood considering the fact that markers were generated independently between different trials. Thus, the hypothesis was confirmed. As an extreme example, when 100 randomly generated background seeds were used, not a single failure was recorded from 7900 tests.

9. APPLICATION: DETERMINATION OF TISSUE DENSITY

It has been estimated that among children under 15 years of age who undergo a CT examination, almost 1 in a 1000 will ultimately die from a cancer attributable to their x-ray exposure [42]. CT scanning is a high-radiation-dose procedure, and its use in diagnosis is increasing. Accurate estimation of the amount of radiation

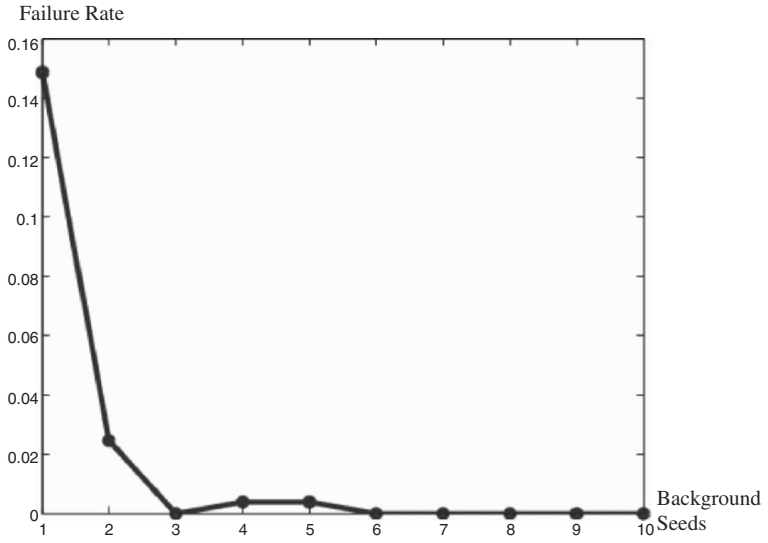


Figure 9. Failure rate as a function of the number of seeds used.

energy deposited in radiosensitive organs during a CT procedure can provide essential information for clinicians to determine an optimal radiation dose. This is particularly relevant for pediatric patients. Such estimation requires knowledge of the tissue characteristics representative of the specific age group.

Pixel values in a CT image (called CT numbers) are directly related to the linear attenuation coefficient, μ, of the tissue being imaged. It is therefore possible to investigate tissue characteristics, such as density and material composition, via CT images given some additional information. μ is the probability that an x-ray photon will interact with the material that it traverses per unit length traveled, and is measured in units of cm^{-1} [43]. μ can be expressed as

$$\mu = \rho\nu, \tag{16}$$

where ρ is the physical density of the material being imaged, and ν is called the mass attenuation coefficient. ν is dependent on the chemical composition of the medium, and can be computed (as a weighted sum) from the mass attenuation coefficients of the constituent elements. ν is essentially constant across the population and age groups, as well established in the literature. It is possible to use published elemental composition of tissues to calculate ν [44].

On the other hand, when it is possible to calibrate the CT scanner using known attenuation materials, μ can be obtained empirically via

$$\mu = \mu_{\mathrm{H_2O}} * (1 + t/K), \tag{17}$$

Table 6. Preliminary Tissue Density Analysis Based on Our Near-Automatic Segmentation Method and the Manual ROI Method

Density (g/cm3)	Manual Cortical Spine	Segmented Cortical Spine	Manual Lung	Segmented Lung	Manual Liver	Segmented Liver
Mean	0.934	1.180	0.488	0.514	1.062	1.060
Stdev	0.011	0.007	0.118	0.96	0.001	0.000

where K is a scanner- and settings-dependent variable whose value can be determined through the machine calibration, t is the CT number acquired on that machine, and $\mu_{\mathrm{H_2O}}$ is the linear attenuation coefficient of water [44]. Thus, ρ can be derived from the calculated ν and the empirically determined μ using Eq. (16).

In our preliminary study, 2D manual delineation of region of interest (ROI) was performed alongside computer-based segmentation, partly for the purpose of validating the analysis using the latter. However, for the large-scale scientific study that would be necessary to verify any density change patterns in children, the manual method would not be feasible or adequate, because of its labor intensiveness, slow speed, and often insufficient level of consistency (both inter- and intra-subject). A further limitation of the manual delineation is that it is only performed in 2D.

Access to 9 sets of CT images was obtained with ethics approval for neonates aged 2 days to 10 months. The mean pixel values of the liver, lung, and spine were determined by either (a) manual sample measurements using an ROI or (b) by using the GDM described above to segment the organs before statistical analysis was performed. In Table 6, the results using the computer and manual methods are tabulated [44]. In evaluating these results, it is worth noting that the cortical layer of the spine is extremely thin at places, which, coupled with the partial volume effect of the imaging and the fact that the computer-based segmentation was performed in a higher-dimensional space than the manual method, results in a likely inconsistency between the two methods in the extent to which the medullary bone tissue was excluded. This may partially explain the discrepancy related to the cortical spine. In general, however, there is good agreement between the results obtained using the two methods. Further experiments and statistical analysis will be carried out to quantify the extent of the agreement. Considerable variations in densities have been found in neonates, particularly in the lungs and bony regions.

Despite their preliminary nature, our results indicate a distinct possibility that the physical densities of some tissues may change considerably in neonates. From these results, a larger-scale study of tissue densities seems to be warranted.

10. DISCUSSION AND CONCLUSION

10.1. Comparison with other work

The GVF approach [13] reviewed in Section 2 bears some similarity to the proposed approach. Through a diffusion process, the forces from image gradients are extended. However, due to the indiscrimination between object and irrelevant gradients, the vector force field it produces is not globally consistent. Thus, although the initialization requirements can be somewhat relaxed through the use of GVF, oftentimes they are still too restrictive to be used in a gradient-rich image for automatic segmentation.

On the other hand, the watersnake method [19] applies snake-like motion to watersheds, resulting in a much smoother version of the latter than a classical, plain watershed transform. However, this approach suffers from the drawback of being sensitive to parameters in the same way as the classical PDM does. More importantly, the watershed framework is restrictive as to what kind of shape preference or knowledge can be used and the manner that it is used. Thus, for example, it is not possible to extend this approach to be based on a statistical deformable model.

10.2. Future Work

10.2.1. Multiple Region Concurrent Coupled Segmentation

Competitive simultaneous multiple-region segmentation using deformable models is an attractive idea that has been explored with both the PDM and the GDM [17, 45–48]. The attractiveness is several-fold in nature. The first aspect is the ability to take advantage of the a priori knowledge of the classes associated with the regions. The a priori statistical knowledge derived from registered pre- and post-contrast T1-weighted MR images, for example, has been used in [11].

Second, in this approach segmentation of one region is no longer an isolated effort, but is constrained by similar efforts nearby, allowing more information to be used for the segmentation of any given region. Additionally, this allows the segmentation to be more robust to ambiguity arising from boundary deficiencies, as globally optimized segmentation helps diminish common problems such as leakage through weak boundaries. These are natural features of the coupled segmentation paradigm, where, for example, a functional can be designed such that a curve's evolution guided by minimization of that functional involves not only the curve in question but also its siblings [48]. Similarly, through the combination of single-phase functions, the Chan-Vese multiphase level set function [47] achieves a partition of the image using global information.

A consequence of a global approach such as [47, 48] is that the segmented regions are consistent with each other (i.e., no vacuum or overlapping), in contrast to the need for careful integration if the regions are segmented independently.

Relatedly, the computational efficiency may be markedly improved for simultaneous coupled segmentation if the number of regions is significant. Particularly in medical imaging, segmentation of one organ may be more robust and more efficient when taking into account segmentation of nearby organs, as recently demonstrated by Li et al. [49] using a competitive level set for automatic computer-aided dental x-ray analysis. These are also advantages of the multiple-region coupled approach.

Initialization for multiphase level sets can be more complex than the binary case. Furthermore, the multitude of parameters can make tuning exceedingly difficult. These issues lead to an increased need for a reliable deformable model framework that is insensitive to variation in initialization and parameters. This can be a natural extension of the approach presented in this chapter. In fact, the ingredients of this approach — the SKIZ, the FMM, and the deformable model (in one of its extended version as reviewed above) — are all inherently capable of handling multi-region competition.

10.3. Conclusion

We have presented a generic approach to reducing the behavioral dependences on the initialization and parameters for deformable models. Based on topographic distance transforms from two markers, this novel approach uses an object-versus-background identification in deformable models. In this approach, GTDs are computed on the gradient magnitude image in order to locate appropriate gradients given the two identifying markers. This information is integrated into a deformable model as forces or speed terms to influence its evolution. This work takes advantage of existing theoretical research into the watershed transform, yet it is outside the watershed framework and preserves fully the advantages of deformable models. The implementation is based on efficient numerical methods. Our experiments reveal that, even when a relatively high level of accuracy is needed to be achieved, the requirement for initialization was minimal, and that the dependence on parameters was limited in that the same parameters were applicable to an entire set of images.

The approach described in this chapter has been applied to determination of tissue characteristics in early childhood using a CT database of neonates, and will be used for further pediatric dosimetry studies. We believe that by relaxing dependences on the initialization and parameters this approach will enable a degree of automation needed for segmentation-based construction of organ models from large image databases using deformable models, particularly when seeds can be placed automatically based on, for example, a priori knowledge regarding anatomy and the intensity differentiation between object and background.

11. ACKNOWLEDGMENTS

The authors are grateful to Dr. Donald McLean and Mr. Luke Barclay of Westmead Hospital, Sydney, Australia, and the University of Sydney, Australia,

for their contribution and support; and to Dr. Pierrick Bourgeat and Dr. Hans Frimmel for proofreading the manuscript and providing many invaluable suggestions.

The authors thank Drs. Simon Warfield, Michael Kaus, Ron Kikinis, Peter Black, and Ferenc Jolesz of the Department of Neurosurgery and the Surgical Planning Laboratory, Department of Radiology of the Harvard Medical School at Brigham and Womens's Hospital, Boston, Massachusetts, USA, for sharing the SPL and NSG Brain Tumor Segmentation Database.

This work benefited from the use of the Insight Segmentation and Registration Toolkit (ITK), open-source software developed as an initiative of the US National Library of Medicine, and available at www.itk.org.

The authors are grateful to Professor Luc Soler of Digestive Cancer Research Institute, IRCAD, Strasbourg, France, for sharing the database of livers used in this publication.

The authors wish to thank Andrea J.U. Mewes, Simon K. Warfield, and Johannes Pauser of the Computational Radiology Laboratory, Harvard Medical School, Department of Radiology, Brigham and Women's Hospital and Children's Hospital, Boston, Massachusetts, USA, for sharing the original MR scans presented with the femoral bone segmentation in this chapter, and interactively segmenting the scans. Permission from the Westmead Hospital, Sydney, Australia, to use the data for this publication is gratefully acknowledged.

12. NOTES

1. In this chapter the term GAC refers to both the original 2D version and the subsequently extended version into 3D (which is also called minimal surface models [4]).
2. For the inside part of the model inside the target, a consistently expanding flow from the viewpoint of the model may promote more robustness than using Eq. (12). Therefore, we also implemented an alternative that takes into account the direction of the normal vector if any portion of the model is interior to the target boundary. However, it has not been our first choice of methods because of its inseparability from the model.
3. An exception is case 9, for which a total of 15 markers were placed inside and around the tumor in order to overcome the strong gradients present in and around the tumor. Essentially, it is a case where the tumor and normal tissues are very difficult to be segmented together using the same parameters applied to the other cases. Nonetheless, for the sake of completeness we present the result for this case together with the other cases.

13. REFERENCES

1. Kass M, Witkin M, Terzopoulos D. 1987. Snakes: active contour models. *Int J Comput Vision* **1**:321–331.
2. McInerney T, Terzopoulos D. 1995. Topologically adaptable snakes. In *Proceedings of the fifth international conference on computer vision*, pp. 840–845. Washington, DC: IEEE Computer Society.
3. Malladi R, Sethian JA, Vemuri BC. 1995. Shape modeling with front propagation: a level set approach. *IEEE Trans Pattern Anal Machine Intell* **17**(2):158–175.

4. Caselles V, Kimmel R, Sapiro G. 1997. Geodesic active contours. *Int J Comput Vision* **22**(1):61–79.
5. Cootes T, Hill A, Taylor C, Haslam J. 1994. The use of active shape models for locating structures in medical images. *Image Vision Comput* **12**(6):355–366.
6. Cootes T, Taylor C. 2001. Statistical models of appearance for medical image analysis and computer vision. In *Proc SPIE Med Imaging* **4322**:236–248.
7. Amini AA, Weymouth TE, Jain RC. 1990. Using dynamic programming for solving variational problems in vision. *IEEE Trans Pattern Anal Machine Intell* **12**:855–867.
8. Nipper JC, Williams JL, Bolch WE. 2002. Creation of two tomographic voxel models of paediatric patients in the first year of life. *Phys Med Biol* **47**:3143–3164.
9. Cohen L, Cohen I. 1993. Finite-element methods for active contour models and balloons for 2D and 3D images. *IEEE Trans Pattern Anal Machine Intell* **15**(11):1131–1147.
10. Paragios N, Mellina-Gottardo O, Ramesh V. 2004. Gradient vector flow fast geometric active contours. *IEEE Trans Pattern Anal Machine Intell* **26**(3):402–417.
11. Ho S, Bullitt E, Gerig G. 2002. Level set evolution with region competition: automatic 3D segmentation of brain tumors. In R. Kasturi, D. Laurendeau, and C. Suen, editors, *Proceedings of the 16th international conference on pattern recognition*, pp. 532–535. Washington, DC: IEEE Computer Society.
12. Xu C, Prince JL. 1998. Snakes, shapes, and gradient vector flow. *IEEE Trans Image Process* **7**(3):359–369.
13. Xu C, Prince JL. 2000. Gradient vector flow deformable models. In *Handbook of Medical Imaging*, pp. 159–169. Ed I Bankman. New York: Academic Press.
14. Chen T, Metaxas D. 2002. Integration of Gibbs prior models and deformable models for 3D medical image segmentation. In *Proceedings of the 16th international conference on pattern recognition*, Vol. 1, pp. 719–722. Washington, DC: IEEE Computer Society.
15. Metaxas D, Chen T. 2004. A hybrid 3D segmentation framework. *IEEE Int Symp Biomed Imaging* **1**:13–16.
16. Parke J, Keller J. 2001. Snakes on the watershed. *IEEE Trans Pattern Anal Machine Intell* **23**(10):1201–1205.
17. Zhu SC, Yuille AL. 1996. Region competition: Unifying snakes, region growing, and bayes/MDL for multiband image segmentation. *IEEE Trans Pattern Anal Machine Intell* **18**(9):884–900.
18. Thodberg HH, Rosholm A. 2003. Application of the active shape model in a commercial medical device for bone densitometry. *Image Vision Comput* **21**:1155–1161.
19. Nguyen HT, Worring MI, van den Boomgaard R. 2003. Watersnakes: energy-driven watershed segmentation. *IEEE Trans Pattern Anal Machine Intell* **25**(3):330–342.
20. Tek H, Kimia B. 1997. Volumetric segmentation of images by three-dimensional bubbles. *Comput Vision Image Understand* **65**(2):246–258.
21. Najman L, Schmitt M. 1994. Watershed of a continuous function. *Signal Process* **38**:99–112.
22. Meyer F. 1994. Topographic distance and watershed lines. *Signal Process* **38**:113–125.
23. Roerdink JBTM, Meijster A. 2001. The watershed transform: definitions, algorithms and parallelization strategies. *Fundam Inform* **41**(1–2):187–228.
24. Maragos P, Butt MA. 1998. Advances in differential morphology: image segmentation via eikonal PDE and curve evolution and reconstruction via constrained dilation flow. In *Mathematical morphology and its applications to image and signal processing*, pp. 167–174. Ed HJAM Heijmans, JBTM Roerdink. Amsterdam: Kluwer Academic.
25. Meyer F. 2001. An overview of morphological segmentation. *Int J Pattern Recognit Artif Intell* **15**(7):1089–1118.
26. Bueno G, Mussea O, Heitza F, Armspach JP. 2001. Three-dimensional segmentation of anatomical structures in MR images on large data bases. *Magn Reson Imaging* **19**(1):73–88.
27. Meyer F, Maragos P. 1999. Multiscale morphological segmentations based on watershed, flooding, and eikonal PDE. *Scale-space theories in computer vision*, pp. 351–362. Ed M Nielsen, P Johansen, OF Olsen, J Weickert. Berlin: Springer.

28. Beucher S. 2001. Geodesic reconstruction, saddle zones and hierarchical segmentation. *Image Anal Stereol* **20**:137–141.
29. Soille P. 2003. *Morphological image analysis: principles and applications*, 2nd ed. Berlin: Springer.
30. Vincent L. 1993. Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. *IEEE Trans Image Process* **2**(2):176–201.
31. Maragos P, Butt MA, Pessoa LFC. 1998. Two frontiers in morphological image analysis: differential evolution models and hybrid morphological/linear neural networks. In *Proceedings of the international symposium on computer graphics, image processing and vision*, Vol. 11, pp. 10–17. http://cvsp.cs.ntua.gr/publications/confr/MaragosButtPesoa_DifMorfMRLNN_SIBGRAPI1998.pdf.
32. Adalsteinsson D, Sethian JA. 1995. A fast level set method for propagating interfaces. *J Comput Phys* **118**:269–277.
33. Sethian JA. 1996. A fast marching level set method for monotonically advancing fronts. *Proc Natl Acad Sci USA* **93**(4):1591–1595.
34. Sethian JA. 1999. *Level set methods and fast marching methods: evolving interfaces in geometry, fluid mechanics, computer vision, and materials science*. Cambridge: Cambridge UP.
35. Vincent L, Soille P. 1991. Watersheds in digital spaces: an efficient algorithm based on immersion simulation. *IEEE Trans Pattern Anal Machine Intell* **13**(6):583–598.
36. Robinson K, Whelan PF. 2004. Efficient morphological reconstruction: a downhill filter. *Pattern Recognit Lett* **25**(15):1759–1767.
37. Marr D, Hildreth E. 1980. Theory of edge detection. *Proc Roy Soc London* **B207**:187–217.
38. Lorigo LM, Faugeras OD, Grimson WEL, Keriven R, Kikinis R, and Westin C-F. 1999. Co-dimension 2 geodesic active contours for MRA segmentation. In *Proceedings of the international conference on information processing in medical imaging*, pp. 126–133. Washington, DC: IEEE Computer Society.
39. Warfield SK, Kaus M, Jolesz FA, Kikinis R. 2000. Adaptive, template moderated, spatially varying statistical classification. *Med Image Anal* **4**(1):43–55.
40. Yushkevich PA, Piven J, Cody H, Ho S, Gee JC, Gerig G. 2005. User-guided level set segmentation of anatomical structures with ITK-SNAP. *Insight J*. To appear.
41. Kaus M, Warfield SK, Nabavi A, Jolesz FA, Black PM, Kikinis R. 2001. Automated segmentation of MRI of brain tumors. *Radiology* **218**(2):586–591.
42. Brenner DJ, Elliston CD, Hall EJ. 2001. Estimated risks of radiation-induced fatal cancer from pediatric CT. *Am J Roentgenol* **176**:289–296.
43. Curry TS, Dowdey JE, Murry RC. 1990. *Christensen's physics of diagnostic radiology*. Philadelphia: Lea & Febiger.
44. McLean D, Barclay L, Li R, Ourselin S. 2006. Estimation of paediatric tissue characteristics from CT image analysis. In *Proceeding of the 6th international topical meeting on industrial radiation and radioisotope measurement applications*. *Lecture notes in computer science*, Vol. 3708. Ed J Blanc-Talon, W Philips, DC Popescu, P Scheunders. Berlin: Springer. To appear.
45. Zhao H, Chan T, Merriman B, Osher S. 1996. A variational level set approach to multiphase motion. *J Comput Phys* **127**(1):179–195.
46. Paragios N, Deriche R. 2000. Coupled geodesic active regions for image segmentation: a level set approach. In *Proceedings of the sixth European conference on computer vision*, Part 2. *Lecture notes in computer science*, Vol. 1843, pp. 224–240. Berlin: Springer.
47. Vese LA, Chan TF. 2002. A multiphase level set framework for image segmentation using the mumford and shah model. *Int J Comput Vision* **50**(3):271–293.
48. Yezzi AJ, Tsai A, Willsky A. 2002. A fully global approach to image segmentation via coupled curve evolution equations. *J Vis Commun Image Represent* 50(13):195–216.
49. Li S, Fevens T, Krzyzak A, Jin C, Li S. 2005. Toward automatic computer aided dental x-ray analysis using level set method. In *Proceedings of the international confernece on medical image computing and computer assisted intervention*, pp. pp. 670–678. Berlin: Springer.

10

APPLICATION OF DEFORMABLE MODELS FOR THE DETECTION OF ACUTE RENAL REJECTION

Ayman El-Baz, Aly A. Farag, and Seniha E. Yuksel

Computer Vision & Image Processing Laboratory (CVIP)
University of Louisville, Louisville, Kentucky

Mohamed E.A. El-Ghar, Tarek A. Eldiasty, and Mohamed A. Ghoneim

Urology and Nephrology Department
University of Mansoura, Mansoura, Egypt

Acute rejection is the most common reason for graft failure after kidney transplantation, and early detection is crucial to survival of function in the transplanted kidney. In this study we introduce a new framework for automatic classification of normal and acute rejection transplants from Dynamic Contrast Enhanced Magnetic Resonance Images (DCE-MRI). The proposed framework consists of three main steps. The first isolates the kidney from the surrounding anatomical structures by evolving a deformable model based on two density functions; the first function describes the distribution of the gray level inside and outside the kidney region and the second describes the prior shape of the kidney. In the second step, nonrigid registration algorithms are employed to account for the motion of the kidney due to the patient's breathing. In the third step, the perfusion curves that show transportation of the contrast agent into the tissue are obtained from the segmented cortex of the whole image sequence of the patient. In the final step, we collect four features from these curves and use Bayesian classifiers to distinguish between acute rejection and normal transplants. Applications of the proposed approach yield promising results that would, in the near future, replace the use of current technologies such as nuclear imaging and ultrasonography, which are not specific enough to determine the type of kidney dysfunction.

Address all correspondence to: Dr. Aly A. Farag, Professor of Electrical and Computer Engineering, University of Louisville, CVIP Lab, Room 412, Lutz Hall, 2301 South 3rd Street, Louisville, KY 40208, USA. Phone: (502) 852-7510, Fax: (502) 852-1580. aafara01@louisville.edu.

1. INTRODUCTION

In the United States, more than 12000 renal transplantations are performed each year [1], but the transplanted kidneys face a number of surgical and medical complications that cause a decrease in their functionality. Although such a decrease in functionality can be reversed by proper treatment strategies and drug therapies [2], the currently used techniques are not specific enough to diagnose the possible diseases. For example, *measurement of creatinine levels* can be affected by diet and medications [3]. *Clearances of inulin and DTPA* require multiple blood and urine tests, and they provide information on both kidneys together, but not unilateral information [4]. On the other hand, imaging tests are favorable since they provide information on each kidney separately. However, the most frequently used imaging technique, *scintigraphy* (also called nuclear imaging), preferred for its good functional information, does not provide good spatial resolution. Without good spatial resolution, precise anatomical detail cannot be obtained, so the diseases that affect the different parts of the kidney (such as the cortex and medulla) cannot be diagnosed accurately [5]. Moreover, scintigraphy exposes the patients to a small dose of radioactivity [6]. Another traditional imaging modality, *Computed Tomography* (CT), uses nephrotoxic contrast agents and exposes patients to radiation despite its superior functional and anatomical information [7]. *Ultrasonography* has been found to show normal findings despite severe renal dysfunction [8], and several studies on color Doppler sonography and power Doppler sonography (e.g., [9–11]) have not been able to yield significant information to evaluate renal function. Therefore, despite all its high costs and morbidity rates, *biopsy* is remains the gold standard for diagnosis after renal transplantation. Unfortunately, the downside of biopsy is that patients are subjected to such risks as bleeding and infection; moreover, the relatively small needle biopsies may lead to over- or underestimation of the extent of inflammation in the entire graft [12]. Hence, a noninvasive and repeatable technique would not only be useful but is needed to ensure survival of transplanted kidneys. For this reason, a fairly new imaging technique, Dynamic Contrast Enhanced Resonance Imaging (DCE-MRI), has gained considerable attention due to its ability to yield superior anatomical and functional information. With DCE-MRI it has been possible to distinguish the different structures of the kidney (such as the cortex and medulla) in a noninvasive way, and, combined with function information, image analysis with DCE-MRI can help in detecting diseases that affect the different parts of renal transplants. Therefore, the CVIP Lab at the University of Louisville and the Urology and Nephrology Center at the University of Mansoura have begun an ongoing collaboration to detect acute rejection from normal functioning of transplants. This study focuses on the DCE-MRI findings of this collaboration.

Acute rejection is the attack of the recipient's immune system on a foreign kidney. Therefore, prompt anti-rejection therapy can suppress the immune system and save the transplanted kidney [13]. Moreover, acute rejection is the most important cause of renal dysfunction [14] among the diagnostic possibilities following renal transplantation (e.g., acute rejection, acute tubular necrosis, cyclosporin toxicity, and obstruction [15]). Therefore, early and noninvasive detection of acute rejection is crucial.

Within the framework of our collaborative research, patients' urine and blood are tested daily at 9:00 a.m. for detection of acute rejection after transplantation. Following these tests, the abdomen is imaged using Color Doppler Ultrasonography during the first 2 weeks. If circumstances do not require it previously, DCE-MRI is performed at the end of the 2-week nursing period. If in addition to the other tests, DCE-MRI findings are normal, then no further action is taken. However, if the findings are abnormal, a biopsy is performed to determine the type of rejection or the type of kidney disease. Images of patients with histopathological diagnosis of acute rejection episodes and normal transplant images are sent to the University of Louisville to test and train the analysis software. In this study, we focused on Dynamic Contrast Enhanced Resonance Imaging (DCE-MRI) findings.

In Dynamic Contrast Enhanced Resonance Imaging (DCE-MRI), the contrast agent gadolinium diethylene triamine pentaacetic acid (Gd-DTPA) is injected into the bloodstream. During its passage from the blood to the kidneys and on to the urinary system, the kidneys are imaged rapidly and repeatedly. As the Gd-DTPA perfuses into the kidney, it causes a change in tissue relaxation time, which creates a contrast change (intensity increase/decrease) in the images. If the intensity is averaged over the kidney tissue (either cortex or medulla) and plotted against time, a signal curve (change in average intensity vs. time) is obtained. Such intensity curves obtained from the kidney are called MR renograms [5]. The pattern of an MR renogram is an indicator of the kidney's degree of functionality, and it can be used in determining the type of the rejection or dysfunction. Moreover, since DCE-MRI provides good anatomical information, separate renograms can be obtained for the cortical and medullary structures of the kidney, which will help in distinguishing the diseases that affect different parts of the kidney. Hence, a typical protocol using dynamic MRI involves the following steps: (1) collecting a sequence of images from the kidney region as the contrast agent perfuses through the kidney; (2) following the contrast difference in the kidney over time (image intensity information from the cortex varies with time as the contrast agent perfuses through the kidney); and (3) establishing a correspondence between change of contrast in the image sequence and kidney status. Such a protocol requires some intermediate image processing steps — namely, segmentation, registration, and classification — explained in later sections of this chapter.

2. RELATED WORK IN RENAL IMAGE ANALYSIS USING DCE-MRI

Starting with examination of rats, studies to acquire functional, dynamic, and anatomic information on the kidneys date back to the early 1980s, covering a variety of objectives from simply understanding the normal behavior of a kidney to detecting kidney diseases. The potential of DCE-MRI to help understand kidney function has been investigated in various studies (see [7, 16–28], to cite a few). This imaging modality has been applied to various kidney diseases in several other studies, such as for detection of renal ischemic lesions [29], for evaluation of renal failure [30], for acute tubular necrosis [31], for assessment of chronic medical nephropathies [32], to differentiate obstructive from nonobstructive hydronephrosis [33, 34], to evaluate rejection [35], to differentiate acute tubular necrosis from transplant rejection in patients with delayed kidney function [36], to differentiate acute tubular necrosis from acute transplant rejection [14], for functional and morphological evaluation of live kidney donors [37], to study renal function in nephrolithiasis and hydronephrosis [26], to observe the effect of hypertension on kidney function [38], and for evaluation of normal and transplanted rat kidneys [12].

These studies are significant, as they have ascertained DCE-MRI to be a very promising technique not only for assessing renal blood flow and consequently help understand kidney function, but also to evaluate several clinical disorders. However, most of these studies were performed by radiologists who selected a region of interest (ROI) (a small window) from the kidney and followed signal change within this region of interest. Unfortunately, such approaches not only require manual interaction of the operators, but also ROI selection biases the final decision and brings up the same issue of over- or underestimating the problem in the entire graft, just as with biopsy. Moreover, manual window selection, and generating a function curve from this window over a time-sequence of images, assumes that the kidneys (renal contours) remain exactly the same from scan to scan. However, renal contours may not always exactly match due to patient movement or breathing effects; therefore, image registration schemes should be applied first before ROI selection. Also, to automate the algorithm and to cancel ROI dependency, segmentation algorithms that can separate the kidney from the surrounding structures, and that can further separate the kidney into its cortex and medulla compartments are needed. In the remainder of this section, we discuss the previous computerized studies that made use of image processing techniques.

To the best of our knowledge, the first computerized renal image analysis scheme was developed by Gerig et al. [25] in 1992. In this study, the prior contour information for each study was obtained by manually drawing the contour from one image. For the rest of the images, image discontinuities (edges) were detected, and the model curve was matched to these edges using a Hough transform. To extend the algorithm to arbitrary rotation and translation, the model contour was

rotated and translated to get the best correlation, giving the registration parameters of the image. The rotation in this scheme was limited to ∓ 4 degrees in 1-degree steps. For the cases of when the patient inadvertently moved or breathed, detection of kidney contour was severely impeded; therefore, a 50% vote rule was defined, that is, at least 50% of the kidney needed to be detected for registration to work. A similar procedure with some extensions was used by Yim et al. in [22].

The handicaps of this image analysis scheme can be listed as follows: (i) the need for manual selection of a contour for each study; (ii) the problems that can be faced in case of a large movement by the patient, and (iii) the great computational expense of the Hough transform (the algorithm was implemented in parallel in [25], and one patient took about an hour to evaluate). Moreover, this registration method is highly dependent on the edge detection filter. Although the strength of edge detection was increased by using opposed-phase gradient echo imaging that puts a dark line between the water-based kidney and the perirenal fatty tissue, still, the algorithm worked better in smaller areas since increasing the field of view (FOV) parameter caused the algorithm to match more partial contours. Following this same procedure, healthy volunteers and hydronephrosis patients were compared in [26], and DCE-MRI was shown to be a reliable method for kidney analysis.

Noting the lack of special protocols and the consequent problems with edge detection in the registration process, the second image analysis study came from Giele et al. [28] in 2001, where three movement correction methods were compared based on image matching, phase difference movement detection (PDMD), and cross-correlation. In all these methods, a mask is generated from the best image manually, and its similarity to a given image is calculated. Consequently, the (i, j) values that give the highest similarity become the translation parameters for the given image. Among these methods, the PDMD method demonstrated the best performance, but only with an 68% accuracy compared to the results of a radiologist. More importantly, in all three registration algorithms only translational motion was handled and rotational motion was not mentioned, the existence of which has been discussed in a number of studies (see [26, 39, 40]).

For segmentation of the kidneys, Priester et al. [41] subtracted the average of pre-contrast images from the average of early-enhancement images, and thresholded the subtraction image to obtain a black-and-white mask kidney image. Following this step, objects smaller than a certain size were removed and the remaining kidney object was closed with erosion operations and manual interactions to obtain a kidney contour.

This approach was furthered by Giele et al. [5] by applying an erosion filter to the mask image to obtain a contour via a second subtraction stage. The possible gaps in this contour were closed by a hull function to get the boundary of the kidney, then via repeated erosions applied to this contour, several rings were obtained, which formed the basics of the segmentation of the cortex from the medulla structures. Of course, in such a segmentation the medulla structures were intermixed

with the cortex structures, so a correlation study had to be applied to better classify the cortical and medullary pixels.

Also in 2001, Boykov et al. [42] presented the use of graph cuts using Markov models. In this algorithm, each voxel is described as a vector of intensity values over time, and initially, several seed points are put on the objects and on the background to give user-defined constraints as well as an expert sample of intensity curves. These expert samples of intensity curves are used to compute a two-dimensional histogram that would be used further as a data penalty function in minimizing the energy function in the Markov model. Although the results looked promising, this algorithm was tested only on one kidney volume, and manual interaction was still required.

Following these studies, computerized image analysis schemes for the registration and segmentation of kidneys were introduced by Sun et al. [12, 21, 23, 24, 40, 43] in a series of studies performed on rats and human subjects. The study on humans ([21, 40]) made use of a multistep registration approach. Initially, the edges were aligned using an image gradient-based similarity measure considering only translational motion. Once roughly aligned, a high-contrast image was subtracted from a pre-contrast image to obtain a kidney contour, which was then propagated over the other frames searching for the rigid registration parameters (rotation and translation). For segmentation of the cortex and medulla, the level sets approach of Chan et al. [44] was used.

In most of these previous efforts, healthy transplants were used in the image analysis, so the edge detection algorithms were applicable. However, in the case of acute rejection patients, the uptake of contrast agent was decreased, so edge detection algorithms generally failed in giving connected contours. Therefore, in our approach, we avoid edge detection schemes; instead, we combine the use of gray-level and prior shape information.

3. RELATED WORK IN SHAPE-BASED SEGMENTATION

Both active contours and level sets tend to fail in the case of noise, poor image resolution, diffused boundaries, or occluded shapes if they do not take advantage of the a priori models. Four of the popular segmentation algorithms that make use of only gray-level information are shown in Figure 1 implemented with ITK Version 2.0; namely, thresholding level sets [45], fast marching level sets [46], and geodesic active contours [47]. However, especially in the area of medical imaging, organs have well-constrained forms within a family of shapes [48]. Thus, additional constraints based on the shape of the objects have been greatly needed aside from the gray-level information of these objects.

Therefore, to allow shape-driven segmentation, Leventon et al. [49] used a shape prior whose variance was obtained thorough Principal Component Analysis (PCA), and used this shape prior to evolving the level sets to the maximum a

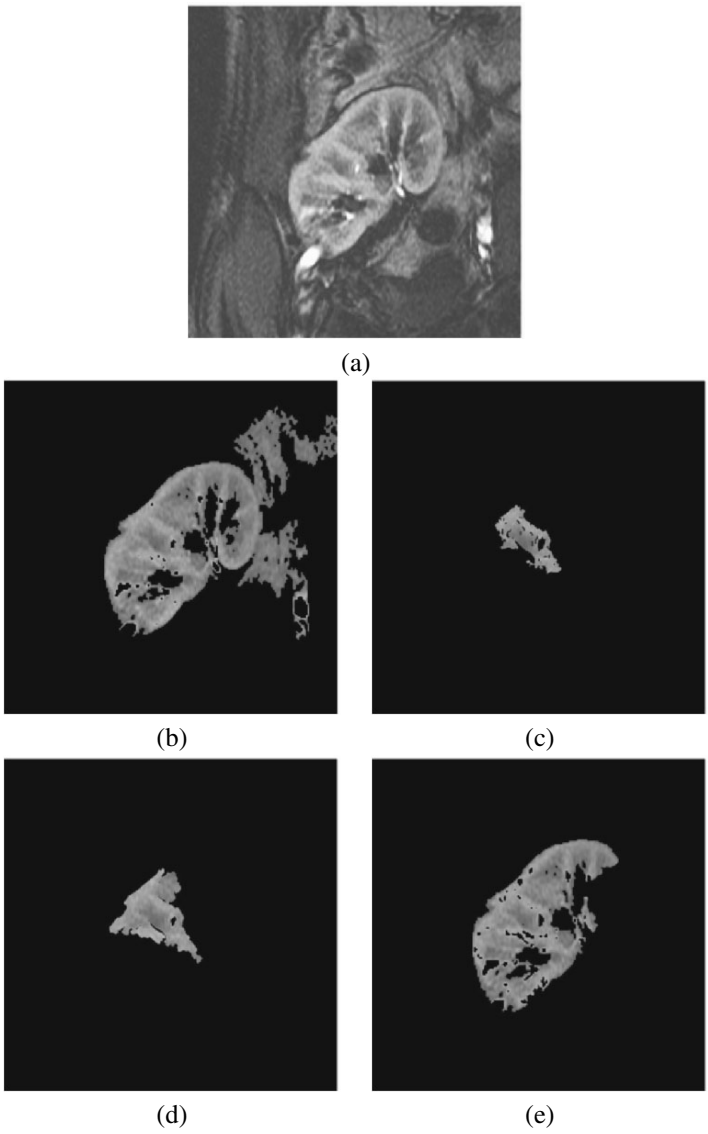


Figure 1. (a) Kidney to be segmented from the surrounding organs. The results of some popular segmentation algorithms that depend only on gray-level and gradient information, (b) connected thresholding, (c) fast marching level sets, (d) geodesic active contours, and (e) thresholding level sets segmentation.

posteriori shape. Chen et al. [50] calculated an average of a registered training set to be a shape prior, and defined an energy functional that basically minimizes a Euclidean distance between a given point and its shape prior. In [51] a representation of the segmenting curve was generated based on the pose and shape parameters of a training set, which were optimized using a region-based energy functional. In [48, 52] a shape prior and its variance obtained from training data were used to define a Gaussian distribution, which was then used in the external energy component of a level sets framework. In Litvin et al.'s study [53], a shape boundary prior was formed from the features of the boundary, and this boundary was used within a level set framework.

Recently, Tsai et al. [54] used a deterministic model to represent the desired shape as a linear combination of weighted signed 2D distance maps and estimated these weights by minimizing a mutual information-based cost function. In addition, Yang et al. [55] described the shapes with a multidimensional Gaussian probability model of these weights.

Different from previous studies, in our segmentation approach the contour points move by comparing the energy at the neighboring points; but with different energy terms that will be explained in Section 5. With this new energy component, the contour moves with both the current gray-level and prior shape information; thus, it does not get stuck in edge points, and handles intricate shapes. Also different from previous studies, the prior shape information is a density estimation of the signed distance map inside and outside the object of interest, not the average image itself. This algorithm overcomes the inability of deformable models to stop in a state of high noise or in the case of missing edges.

During the remainder of the present study, we will discuss the data acquisition protocol in Section 4 and give an overview of the components of the framework. In Section 5 we will introduce our segmentation approach, starting with the general idea of the method. We will then go into the details and explain shape model construction from a database in Section 5.1. In Section 5.2 we will offer an introduction to our density estimation scheme; in Sections 5.3, 5.4, and 5.5 we will explain the details of implementation of our density estimation framework. Section 5.3 introduces a modified EM algorithm, and Section 5.4 introduces a sequential EM algorithm to closely initialize the parameters of the EM estimation. Section 5.5 explains classification of the Gaussian components and the stopping conditions. Given the shape model and the density estimations for the shape model and the gray level of an image to segment, in Section 5.6 we explain how the deformable model evolves in an image. After the whole sequence of a patient is segmented, we introduce a nonrigid registration algorithm in Section 6 that deforms the kidney object on isocontours. Once all the images are aligned, the cortex of one early-enhancement image is extracted in Section 7, and then used as a mask to calculate the average cortex intensity signal in the rest of the sequence. The results of classification based on these signals are presented in Section 8, and

the study is concluded in Section 9 with speculation about future work and further discussion.

4. METHODS AND DATA ACQUISITION

During development of this study, we observed that good selection of a DCE-MRI imaging protocol is as important as the image analysis — if not more important. The key point in the protocol is to take the images as quickly as possible while trying to conserve the quality. A compromise on image quality results in too much noise and partial volume effects; on the other hand, a compromise on speed results in fewer data points, which prevents us from being able to classify signals. Therefore, with collaborative efforts, the protocol was modified a number of times to acquire standard and better-quality imaging. The protocol described below has been found to be optimal with the current MRI hardware (Signa Horizon GE 1.5T scanner).

In our protocol, gradient-echo T1 imaging is employed by a Signa Horizon GE 1.5T scanner (Signa Horizon LX Echo speed; General Electric Medical Systems, Milwaukee, WI, USA) with the use of a phased-array Torso surface coil; the contrast agent (Gadolinium DTPA) is introduced via a wide-bore veno-catheter, placed at the antecubital vein, at a rate of 3–4 ml/sec, with a dose of 0.2 ml/kg·BW. Images are taken at 5 mm thickness with no interslice gap, the repetition time (TR) is 34 msec, the TE minimum, the field of view (FOV) is 42×42 cm, and with a 600×600 matrix. For each patient, 150 temporal sequences of coronal scans are taken at 4-second intervals. A sample of what a DCE-MRI looks like with this protocol is shown in Figure 2.

With this protocol, we propose the following framework [56] for image analysis, as shown in Figure 3:

1. Segmentation of the kidneys from the abdomen images.
2. Nonrigid registration for motion compensation.
3. Renogram generation.
4. Feature extraction.
5. Classification of acute rejection patients and normal transplants.

5. KIDNEY SEGMENTATION

Accurate segmentation of the kidney from DCE-MRI is a challenge, since the gray-level distribution of the kidney and surrounding organs is not highly distinguishable; we thus add additional constraints based on the shape of the objects

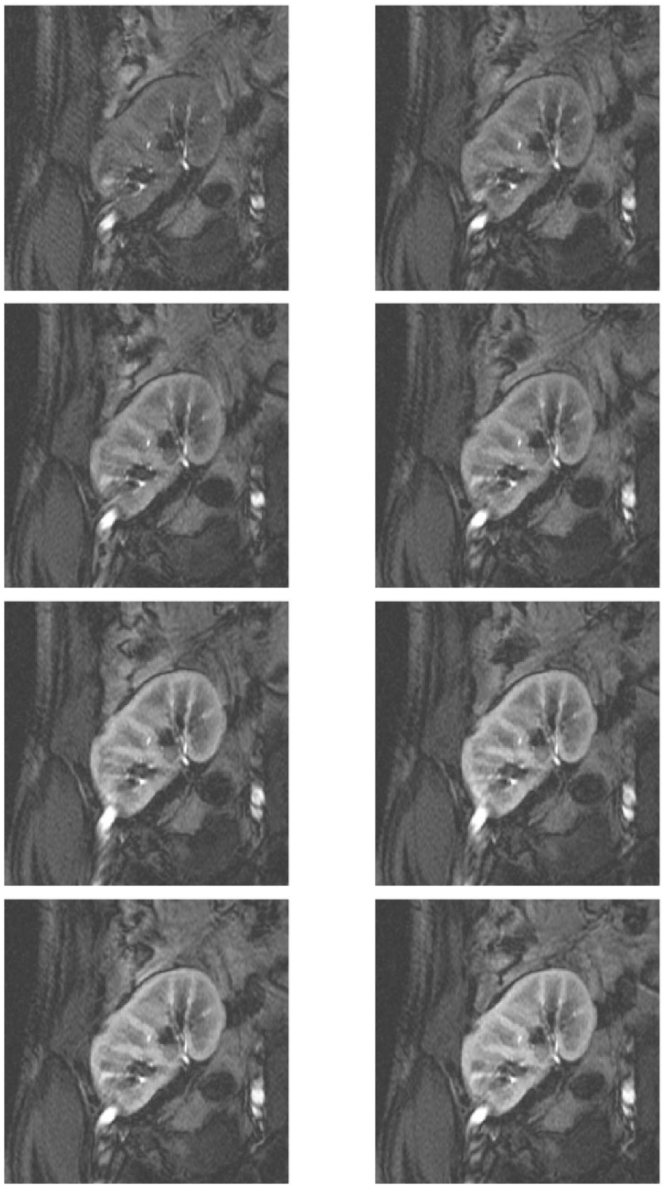


Figure 2. Example of a DCE-MRI series. For each patient, 150 images are taken from one cross-section with 4-second intervals. Eight of one subject (numbers 1, 4, 5, 6, 10, 15, 21, 27) are shown here to give an idea of the protocol.

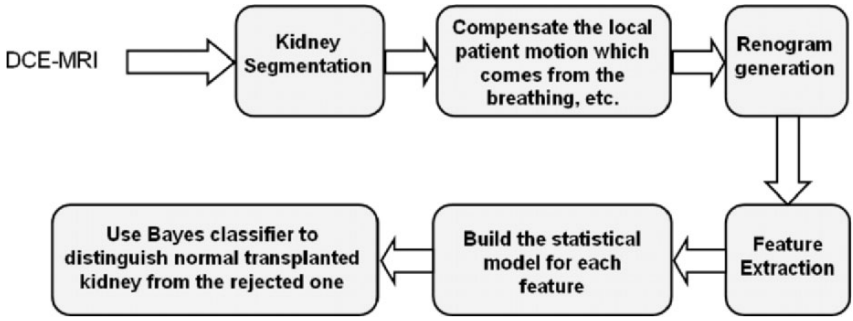


Figure 3. Block diagram of the proposed image analysis to create a CAD system for renal transplantation. See attached CD for color version.

to control evolution of the deformable models in the segmentation process. So the proposed deformable model takes into account not only the gray-level distribution but also a shape model of the kidney that depends on a sign distance map.

A conventional parametric deformable 2D model, or snake, is a curve $\Phi = (\phi(\tau) = (u(\tau), v(\tau); \tau \in T)$ in planar Cartesian coordinates (u, v), where τ is the continuous or discrete index of a contour point, and T is the index range. The deformable model moves through the spatial image domain to minimize the total energy:

$$E = E_{\text{int}} + E_{\text{ext}} = \int_{\tau \in T} \xi_{\text{int}}\big(\phi(\tau)\big) + \xi_{\text{ext}}\big(\phi(\tau)\big) d\tau, \tag{1}$$

where $\xi_{\text{int}}(\phi(\tau))$ and $\xi_{\text{ext}}(\phi(\tau))$ denote the internal and external forces, respectively, that control the pointwise model movements. The total energy is the sum of two terms: the internal energy keeping the deformable model as a single unit and the external one attracting the model to the region boundary. The internal force is typically defined as $\xi_{\text{int}}(\phi(\tau) = \varsigma|\phi'(\tau)|^2 + \kappa|\phi''(\tau)|^2$, where weights ς and κ control the curve's tension and rigidity, respectively, and $\phi'(\tau)$ and $\phi''(\tau)$ are the first and second derivatives of $\phi(\tau)$ with respect to τ.

Typical external forces designed in [57] to lead an active contour toward step edges in a grayscale image $\mathbf{Y}$ are:

$$\begin{aligned} \xi_{ext}\big(\phi(\tau)\big) \quad = \quad & -|\nabla \mathbf{Y}(\phi(\tau))|^2 \ \text{ or} \\ & -\big|\nabla\big[G(\phi(\tau)) * \mathbf{Y}(\phi(\tau))\big]\big|^2 \end{aligned}, \tag{2}$$

where $G(\ldots)$ is a 2D Gaussian kernel and ∇ denotes the gradient operator. But both these and other traditional external forces (e.g., based on lines, edges, or the GVF) fail to make the contour closely approach an intricate boundary with concavities.

Moreover, due to high computational complexity, the deformable models with most of such external energies are slow compared to other segmentation techniques.

As a solution to these problems, we modify the external energy component of this energy formulation, and we formulate an energy function using the density estimations of two distributions: the signed distance map from shape models and the gray-level distribution [58]. The external energy component of our deformable models is formulated as:

$$\xi_{ext}\big(\phi(\tau)\big) = \begin{cases} -p_g(q|k)p_s(d|k) & \text{if} \quad k = k^* \\ p_g(q|k)p_s(d|k) & \text{if} \quad k \neq k^* \end{cases}.$$

In this formulation, k is the region label with $k = 1, 2$, with $k = 1$ denoting the kidney class, $k = 2$ denoting the other tissue, q is the gray level, and d is the signed distance, where $p_s(d|k)$ is the density that describes the signed distance map inside and outside the object, and $p_g(q|k)$ is the density estimation of the gray level. With this energy function, the stochastic external force for each control point $\phi(\tau)$ of the current deformable model evolves in a region k^*. The detailed steps of shape model construction are explained in Section 5.1, and gray-level density estimation is explained in Section 5.2.

5.1. Shape Model Construction

The signed distance map density, $p_s(d|k)$, in the above-mentioned external energy is calculated using a shape model obtained from the images in the data set. The steps to construct this shape model are as follows:

1. Manually segment the objects of interest from the database as shown in Figure 4a,b.
2. Align the images in the training data sets using 2D affine registration (rotation, translation, and scaling) using Mutual Information (MI) [59] as a similarity measure as shown in Figure 4c.
3. Convert aligned images obtained from the previous step to binary images as shown in Figure 4d.
4. Calculate the 2D edge V that describes the boundary of the object for all the manually segmented M number of images obtained in Step 3.
5. From the M number of shapes calculated in Step 3, calculate the signed distance map inside and outside each object. For $m = 1, \ldots, M$, the signed distance map can be calculated as follows:

$$d_m(i,j) = \begin{cases} 0 & (i,j) \in V_i \\ S((i,j), V_m) & (i,j) \in RV_m \\ -S((i,j), V_m) & \text{Otherwise} \end{cases}, \tag{3}$$

where RV is the region lying inside the kidney shape and $S((i,j),V_m)$ is the minimum Euclidean distance between image location (i,j) and curve V_m. The results of this step are shown in Figure 4e,f.

6. Compute the empirical density of the aligned signed distance maps (Figure 4e) as shown in Figure 4g.

7. Calculate the average signed distance map of the kidney at location (i,j) as:

$$d(i,j) = \frac{1}{M}\sum_{m=1}^{M} d_m(i,j). \tag{4}$$

Then threshold the sign distance map at zero level to obtain an average shape.

8. Average the signed distance maps to obtain an average density estimation of the shape.

Figure 5a shows the average signed distance map for a kidney object. We get the average shape of the kidney by thresholding the signed distance map shown in Figure 5a at zero level, the result of which is shown in Figure 5b. In the same way, we calculate the average empirical densities of the empirical densities shown in Figure 4g, with the result shown in Figure 6. With this approach, all the shape variability is gathered into one density function. Compared to the other approaches such as that of [60], we do not need to conduct a principal component analysis, which is difficult for a big database that contains big images (size 600×600).

Figure 7 evaluates the quality of MI-based affine alignment, with the region maps images being pixelwise averages of all the training maps images, $m = 1,\ldots,M$, before and after mutual alignment of training set M. Similar shapes are significantly overlapped after the alignment, that is, we decrease the variability between shapes.

5.2. Density Estimation

In this section we introduce a new statistical model that approximates an empirical probability density function of scalar data with a linear combination of discrete Gaussians (LCDG) with positive and negative components. Due to both positive and negative components, the LCDG approximates interclass transitions more accurately than a conventional mixture of only positive Gaussians. To estimate the parameters of LCDG, we modify an Expectation-Maximization (EM) algorithm to deal with the LCDG with positive and negative components and also propose a novel EM-based sequential technique to get a close initial LCDG approximation with which the modified EM algorithm should start.

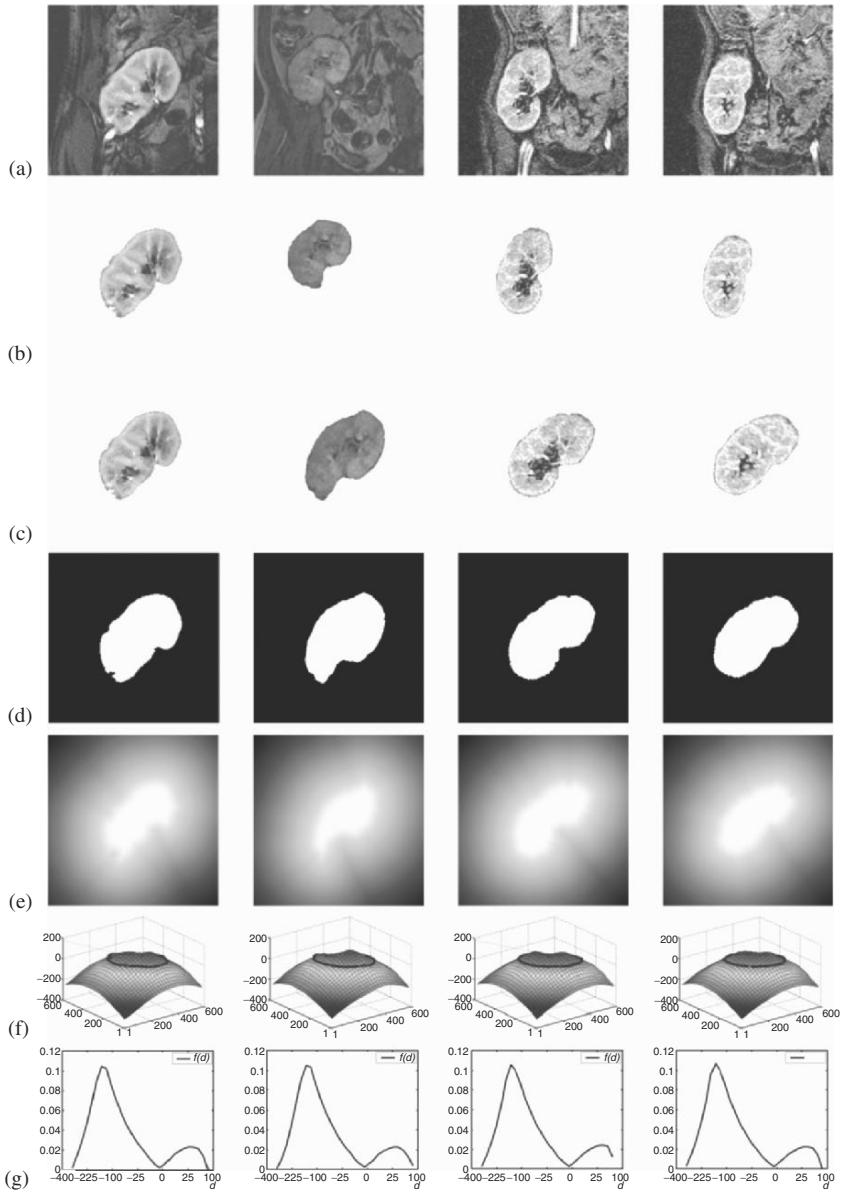


Figure 4. Steps of shape reconstruction. Samples of the database (a), manual segmentation results (b), affine mutual information registration (c), binarization (d), signed distance maps (e), level sets functions (f), and the empirical densities of signed distance maps (g). See attached CD for color version.

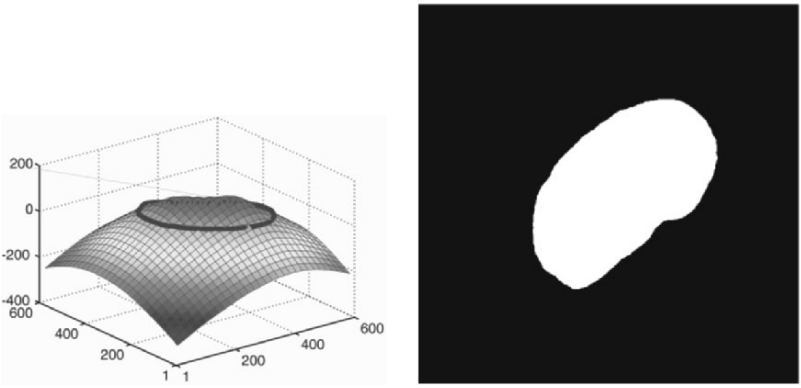


Figure 5. Average signed distance map for the kidney object (a), and the average shape of the kidney after thresholding the average signed distance map from the zero level (b). See attached CD for color version.

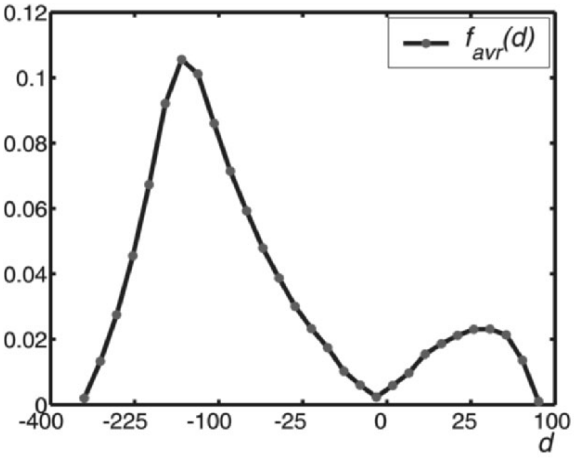
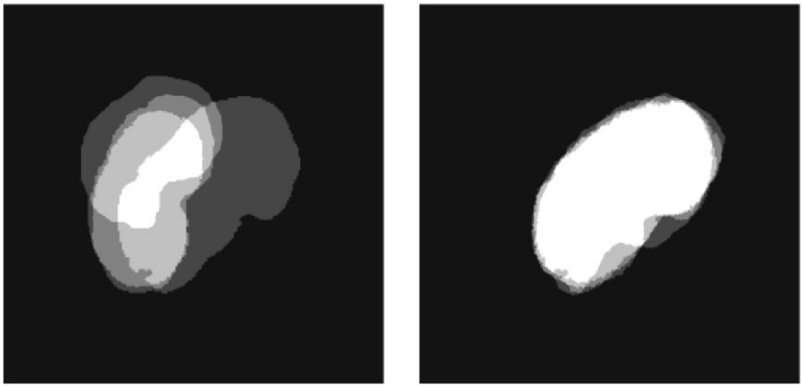


Figure 6. Average empirical signed distance map density representing the shape. It is calculated as the average empirical density of empirical densities shown in Figure 4g. The positive distances indicate the kidney area, while the negative distances indicate the background.

In the following we describe this model to estimate the marginal density for the gray-level distribution $p_g(q)$ in each region. The same approach is used to estimate the density of the signed distances $p_s(d)$ for the object and background.



(a) (b)

Figure 7. Comparison of the shape overlaps in the training data sets before (a) and after (b) alignment.

Let $\mathbf{Q} = \{0, \ldots, Q\}$ denote sets of gray levels q in the given image. Here, $Q + 1$ is the number of gray levels in the given image. Let $\mathbf{S} = \{(i, j) : 1 \leq i \leq I, 1 \leq j \leq J\}$ be a finite arithmetic grid supporting gray level images $\mathbf{Y} : \mathbf{S} \rightarrow \mathbf{Q}$. The discrete Gaussian (DG) is defined as the discrete probability distribution $\Psi_\theta = (\psi(q|\theta) : q \in \mathbf{Q})$ on $\mathbf{Q}$ such that $\psi(q|\theta) = \Phi_\theta(q + 0.5) - \Phi_\theta(q - 0.5)$ for $q = 1, \ldots, Q - 2$, $\psi(0|\theta) = \Phi_\theta(0.5)$, $\psi(Q - 1|\theta) = 1 - \Phi_\theta(Q - 1.5)$ where $\Phi_\theta(q)$ is the cumulative Gaussian (normal) probability function with a shorthand notation $\theta = (\mu, \sigma^2)$ for its mean, μ, and variance, σ^2.

In contrast to a conventional mixture of Gaussians and/or other simple distributions, one per region, we closely approximate the empirical gray-level distribution for the given image with an LCDG having C_p positive and C_n negative components:

$$p_{g:\mathbf{w},\boldsymbol{\Theta}}(q) = \sum_{r=1}^{C_\mathrm{p}} w_{\mathrm{p},r}\psi(q|\theta_{\mathrm{p},r}) - \sum_{l=1}^{C_\mathrm{n}} w_{\mathrm{n},l}\psi(q|\theta_{\mathrm{n},l}), \tag{5}$$

under the obvious restriction on the weights $\mathbf{w} = [w_{p,\cdot}, w_{n,\cdot}]$:

$$\sum_{r=1}^{C_\mathrm{p}} w_{\mathrm{p},r} - \sum_{l=1}^{C_\mathrm{n}} w_{\mathrm{n},l} = 1. \tag{6}$$

To estimate the parameters for the model given in Eq. (6), we modify the conventional EM algorithm to take into account both positive and negative discrete Gaussian components. The details of the algorithm are described below.

5.3. Modified EM Algorithm for LCDG

Let $\mathbf{F} = [f(q) : q \in \mathbf{Q}]$ be an empirical relative frequency distribution of gray levels q collected over the whole image $\mathbf{Y}$. Assuming independent signals in the pixels $(i, j) \in \mathbf{S}$, the empirical distribution represents an unknown probability density $\psi\psi(q)$ of gray values such that $\int_{-\infty}^{\infty} \psi\psi(q)dq \equiv \sum_{q=0}^{Q} f(q) = 1$. We assume that $\mathbf{F}$ is approximated by an LCDG $\mathbf{P}_{C:\mathbf{w},\mathbf{\Theta}} = [p_C(q) : q \in \mathbf{Q}]$ with C_p positive and C_n negative components $\psi(q|\theta)$:

$$p_{\mathbf{w},\mathbf{\Theta}}(q) = \sum_{r=1}^{C_\mathrm{p}} w_{\mathrm{p},r}\psi(q|\theta_{\mathrm{p},r}) - \sum_{l=1}^{C_\mathrm{n}} w_{\mathrm{n},l}\psi(q|\theta_{\mathrm{n},l}). \tag{7}$$

In line with Eq. (7), the positive weights $\mathbf{w}$ are restricted to unity as shown in Eq. (6):

We also assume here that the numbers C_p and C_n of the components of each type are known after the initialization in Section 5.4 and do not change during the EM process. The initialization also provides the starting parameter values $\mathbf{w}^{[0]}$ and $\mathbf{\Theta}^{[0]}$.

The probability densities form a proper subset of the set of the LCDG due to the additional restriction $p_{\mathbf{w},\mathbf{\Theta}}(q) \geq 0$, which holds automatically for probability mixtures with no negative components only. As mentioned earlier, this special feature is ignored because our goal is to closely approximate the empirical data only within the limited range $[0, Q]$. The approximating function of Eq. (7) is assumed strictly positive only in the points $q = 0, 1, \ldots, Q$.

The log-likelihood of the empirical data under the assumed independent signals is as follows:

$$\begin{aligned} L(\mathbf{w}, \mathbf{\Theta}) &= \frac{1}{|\mathbf{S}|} \log \left(\prod_{(i,j)\in\mathbf{S}} f(Y_{i,j}) \right) \\ &= \frac{1}{|\mathbf{S}|} \log \left(\prod_{q\in\mathbf{Q}} (p_{\mathbf{w},\mathbf{\Theta}}(q))^{|\mathbf{S}|f(q)} \right) \\ &= \sum_{q\in\mathbf{Q}} f(q) \log p_{\mathbf{w},\mathbf{\Theta}}(q). \end{aligned} \tag{8}$$

The LCDG that provides a local maximum of the log-likelihood in Eq. (8) can be found using the iterative block relaxation process extending the conventional scheme in [61] that was proposed initially in [62].

Let

$$p^{[m]}_{\mathbf{w},\mathbf{\Theta}}(q) = \sum_{r=1}^{C_\mathrm{p}} w^{[m]}_{\mathrm{p},r}\psi(q|\theta^{[m]}_{\mathrm{p},r}) - \sum_{l=1}^{C_\mathrm{n}} w^{[m]}_{\mathrm{n},l}\psi(q|\theta^{[m]}_{\mathrm{n},l})$$

be the LCDG at step, or iteration, m. Relative contributions of each data item $q = 0, \ldots, Q$ into each positive and negative Gaussian at the step m are specified by the respective conditional weights:

$$\pi_{\rm p}^{[m]}(r|q) = \frac{w_{{\rm p},r}^{[m]} \psi(q|\theta_{{\rm p},r}^{[m]})}{p_{\mathbf{w},\mathbf{\Theta}}^{[m]}(q)}; \;\; \pi_{\rm n}^{[m]}(l|q) = \frac{w_{{\rm n},l}^{[m]} \psi(q|\theta_{{\rm n},l}^{[m]})}{p_{\mathbf{w},\mathbf{\Theta}}^{[m]}(q)}, \tag{9}$$

such that the following condition holds:

$$\sum_{r=1}^{C_{\rm p}} \pi_{\rm p}^{[m]}(r|q) - \sum_{l=1}^{C_{\rm n}} \pi_{\rm n}^{[m]}(l|q) = 1; \; q = 0, \ldots, Q. \tag{10}$$

Multiplying the right-hand side of Eq. (8) by the left-hand side of Eq. (10), which is valid since this latter has unit value, the log-likelihood of Eq. (8) can be rewritten in the equivalent form:

$$\begin{aligned} L(\mathbf{w}^{[m]}, \mathbf{\Theta}^{[m]}) \;\; &= \;\; \sum_{q=0}^{Q} f(q) \left[\sum_{r=1}^{C_{\rm p}} \pi_{\rm p}^{[m]}(r|q) \log p_{\mathbf{w},\mathbf{\Theta}}^{[m]}(q) \right] \\ &- \;\; \sum_{q=0}^{Q} f(q) \left[\sum_{l=1}^{C_{\rm n}} \pi_{\rm n}^{[m]}(l|q) \log p_{\mathbf{w},\mathbf{\Theta}}^{[m]}(q) \right]. \end{aligned} \tag{11}$$

The next equivalent form more convenient for specifying the block relaxation process is obtained after replacing $\log p_{\mathbf{w},\mathbf{\Theta}}^{[m]}(q)$ in the first and second brackets with the equal terms: $\log w_{{\rm p},r}^{[m]} + \log \psi(q|\theta_{{\rm p},r}^{[m]}) - \log \pi_{\rm p}^{[m]}(r|q)$ and $\log w_{{\rm n},l}^{[m]} + \log \psi(q|\theta_{{\rm n},l}^{[m]}) - \log \pi_{\rm n}^{[m]}(l|q)$, respectively, which follow directly from Eq. (9).

The block relaxation converges to a local maximum of the likelihood function in Eq. (11) by repeating iteratively the following two steps of conditional maximization (comprising the E-step and the M-step, respectively [61]):

1. Find the conditional weights of Eq. (9) by maximizing the log-likelihood $L(\mathbf{w}, \mathbf{\Theta})$ under the fixed parameters $\mathbf{w}^{[m]}$, $\mathbf{\Theta}^{[m]}$ from the previous iteration m, and

2. Find the parameters $\mathbf{w}^{[m+1]}$, $\mathbf{\Theta}^{[m+1]}$ by maximizing $L(\mathbf{w}, \mathbf{\Theta})$ under the fixed conditional weights of Eq. (9)

until the changes of the log-likelihood and all the model parameters become small.

The first step performing the conditional Lagrange maximization of the log-likelihood of Eq. (11) under the $Q + 1$ restrictions of Eq. (10) results just in the conditional weights $\pi_{\rm p}^{[m+1]}(r|q)$ and $\pi_{\rm n}^{[m+1]}(l|q)$ of Eq. (9) for all $r = 1, \ldots, C_{\rm p}$; $l = 1, \ldots, C_{\rm n}$ and $q = 0, \ldots, Q$.

The second step performs conditional Lagrange maximization of the log-likelihood of Eq. (11) under the restriction of Eq. (6). It results in the conditional mathematical expectations of the model parameters involving the fixed conditional weights of Eq. (9) as conditional probabilities. The expected weights $\mathbf{w}^{[m+1]}$ at the second step conform to:

$$\begin{aligned} w_{\mathrm{p},r}^{[m+1]} &= \sum_{q\in\mathbf{Q}} f(q)\pi_{\mathrm{p}}^{[m+1]}(r|q) \\ w_{\mathrm{n},l}^{[m+1]} &= \sum_{q\in\mathbf{Q}} f(q)\pi_{\mathrm{n}}^{[m+1]}(l|q) \end{aligned} .$$

The expected parameters $\mathbf{\Theta}^{[m+1]}$ of each Gaussian have conventional forms that follow from the unconditional maximization of the log-likelihood of Eq. (11):

$$\begin{aligned} \mu_{\mathrm{c},r}^{[m+1]} &= \tfrac{1}{w_{\mathrm{c},r}^{[m+1]}} \sum_{q\in\mathbf{Q}} q f(q)\pi_{\mathrm{c}}^{[m+1]}(r|q) \\ (\sigma_{\mathrm{c},r}^{[m+1]})^2 &= \tfrac{1}{w_{\mathrm{c},r}^{[m+1]}} \sum_{q\in\mathbf{Q}} \left(q-\mu_{\mathrm{c},i}^{[m+1]}\right)^2 f(q)\pi_{\mathrm{c}}^{[m+1]}(r|q). \end{aligned}$$

where c stands for p or n, respectively.

This modified EM algorithm is valid while the weights of Eq. (9) are strictly positive, and the initial LCDG approximation should comply with this limitation. The iterations have to be terminated when the log-likelihood of Eq. (11) begins to decrease. Generally, if the initialization is incorrect, this algorithm may diverge from the very beginning. Thus, the initial LCDG has to closely approximate the empirical distribution.

5.4. Sequential EM-Based Initialization

We assume that the number of dominant modes is equal to the given number of classes. To simplify the notation, let the empirical distribution have only two separate dominant modes representing the object (kidney) and the background, respectively, since we have only two classes for this specific problem. The algorithm we present below is easily extended to the general case of $K > 2$ dominant modes. We assume that each dominant mode is roughly approximated with a single Gaussian and the deviations of the empirical density from the two-component dominant Gaussian mixture are described by other components of the LCDG in Eq. (5). Therefore, the model has the two dominant positive weights, say, $w_{\mathrm{p},1}$ and $w_{\mathrm{p},2}$ such that $w_{\mathrm{p},1} + w_{\mathrm{p},2} = 1$, and a number of "subordinate" weights of smaller absolute values such that $\sum_{r=3}^{C_{\mathrm{p}}} w_{\mathrm{p},r} - \sum_{l=1}^{C_{\mathrm{n}}} w_{\mathrm{n},l} = 0$.

The following sequential algorithm allows for estimating both the weights and parameters of the individual Gaussians in the previous LCDG model, including the number of the non-dominant components:

1. Approximate a given empirical distribution $\mathbf{F}$ of gray levels in the image $\mathbf{Y}$ with a dominant mixture $\mathbf{P}_2$ of two Gaussians using the conventional EM algorithm.

2. Find the deviations $\mathbf{\Delta} = [\Delta(q) = f(q) - p_2(q) : \ q \in \mathbf{Q}]$ between $\mathbf{F}$ and P_2 and split them into the positive and negative parts such that $\delta(q) = \delta_{\rm p}(q) - \delta_{\rm n}(q)$:

$$\begin{aligned} \mathbf{\Delta}_{\rm p} &= [\delta_{\rm p}(q) = \max\{\delta(q), 0\} : q \in \mathbf{Q}\}], \\ \mathbf{\Delta}_{\rm n} &= [\delta_{\rm n}(q) = \max\{-\delta(q), 0\} : q \in \mathbf{Q}\}]. \end{aligned} \tag{12}$$

3. Compute the scaling factor for the deviations: $Scale = \int\limits_{-\infty}^{\infty} \delta_{\rm p}(q)dq \equiv \int\limits_{-\infty}^{\infty} \delta_{\rm n}(q)dq$.

4. If the factor $Scale$ is less than an accuracy threshold, terminate and return the model $\mathbf{P}_C = \mathbf{P}_2$.

5. Otherwise, consider the scaled-up absolute deviations $\frac{1}{Scale}\mathbf{\Delta}_{\rm p}$ and $\frac{1}{Scale}\mathbf{\Delta}_{\rm n}$ as two new "empirical densities" and use iteratively the conventional EM algorithm to find sizes $C_{\rm p}$ and $C_{\rm n}$ of the Gaussian mixtures, $\mathbf{P}_{\rm p}$ and $\mathbf{P}_{\rm n}$, respectively, approximating the best scaled-up deviations.

   (a) The size of each mixture corresponds to the minimum of the integral absolute error between the scaled-up absolute deviation $\mathbf{\Delta}_{\rm p}$ (or $\mathbf{\Delta}_{\rm n}$) and its model $\mathbf{P}_{\rm p}$ (or $\mathbf{P}_{\rm n}$). The number of components is increasing sequentially by a unit step while the error is decreasing.

   (b) Due to multiple local maxima, such a search may be repeated several times with different initial parameter values in order to select the best approximation.

6. Scale down the subordinate models $\mathbf{P}_{\rm p}$ and $\mathbf{P}_{\rm n}$ (i.e., scale down the weights of their components) and add the scaled model $\mathbf{P}_{\rm p}$ to and subtract the scaled model $P_{\rm n}$ from the dominant model $\mathbf{P}_2$ in order to form the desired model $\mathbf{P}_C$ of the size $C = 2 + C_{\rm p} + C_{\rm n}$.

Since the EM algorithm converges to a local maximum of the likelihood function, it may be repeated several times with different initial parameter values for choosing the model that yields the best approximation. In principle, this process can be repeated iteratively in order to approximate more and more closely the residual absolute deviations between $\mathbf{F}$ and $\mathbf{P}_C$. But because each Gaussian in the latter model impacts all the values $p(q)$, the iterations should be terminated when the approximation quality begins to decrease.

We use the Levy distance [63], $\rho(\mathbf{F}, \mathbf{P})$, between the estimated model $\mathbf{P}$ and the empirical distribution $\mathbf{F}$ to evaluate the approximation quality. The distance is defined as the minimum positive value α such that the two-sided inequalities $p(q-\alpha)-\alpha \leq f(q) \leq p(q+\alpha)+\alpha$ hold for all $q \in \mathbf{Q}$:

$$\rho(\mathbf{F}, \mathbf{P}) = \min_{\alpha>0}\{\alpha : p(q-\alpha)-\alpha \leq f(q) \leq p(q+\alpha)+\alpha \ \ \forall q \in \mathbf{Q}\}. \quad (13)$$

It has been proven [63] that model $\mathbf{P}$ weakly converges to $\mathbf{F}$ when $\rho(\mathbf{F}, \mathbf{P}) \to 0$. Our experiments show that the modified EM algorithm typically decreases an initially large Levy distance between the empirical distribution and its estimated model to a relatively small value.

5.5. Classification of the Model Components

The final mixed LCDG model P has to be split into K LCDG submodels, one per class, by associating each subordinate component with a particular dominant term in such a way as to minimize the expected misclassification rate. To illustrate the association principle, let us consider the bimodal case with the two dominant Gaussians having the mean values μ_1 and μ_2; $0 < \mu_1 < \mu_2 < Q$. Let all the subordinate components be ordered by their mean values, as well. Then those with mean values smaller than μ_1 and greater than μ_2 relate to the first and second class, respectively. The components having the mean values in the range $[\mu_1, \mu_2]$ are associated with the classes by simple thresholding such that the means below the threshold, t, belong to the components associated with the first class. The desired threshold minimizes the classification error $e(t)$:

$$e(t) = \int_{-\infty}^{t} p(q|2)dq + \int_{t}^{\infty} p(q|1)dq. \quad (14)$$

To make our approach of density estimation clear, we will discuss step-by-step density estimation for the mixed empirical density of the average signed distance map of the kidney object. Figure 8a shows the initial step of the bimodal empirical distribution of signed distance map in Figure 6. The dominant modes represent the brighter kidney area and its background, respectively. Figure 8b shows the deviation and absolute deviation between the two dominant modes and the empirical density. Figure 8c–f shows the final DG mixtures; the initial mixed LCDG model consists of the 2 dominant, 3 additive, and 4 subtractive DGs, that is, $C_{\mathrm{p}} = 5$ and $C_{\mathrm{n}} = 4$. Figure 8f shows that our modified EM algorithm converges after the 16 first iterations of the refinement before the process terminates and increases the log-likelihood from -6.18 for the initial LCDG to -5.10.

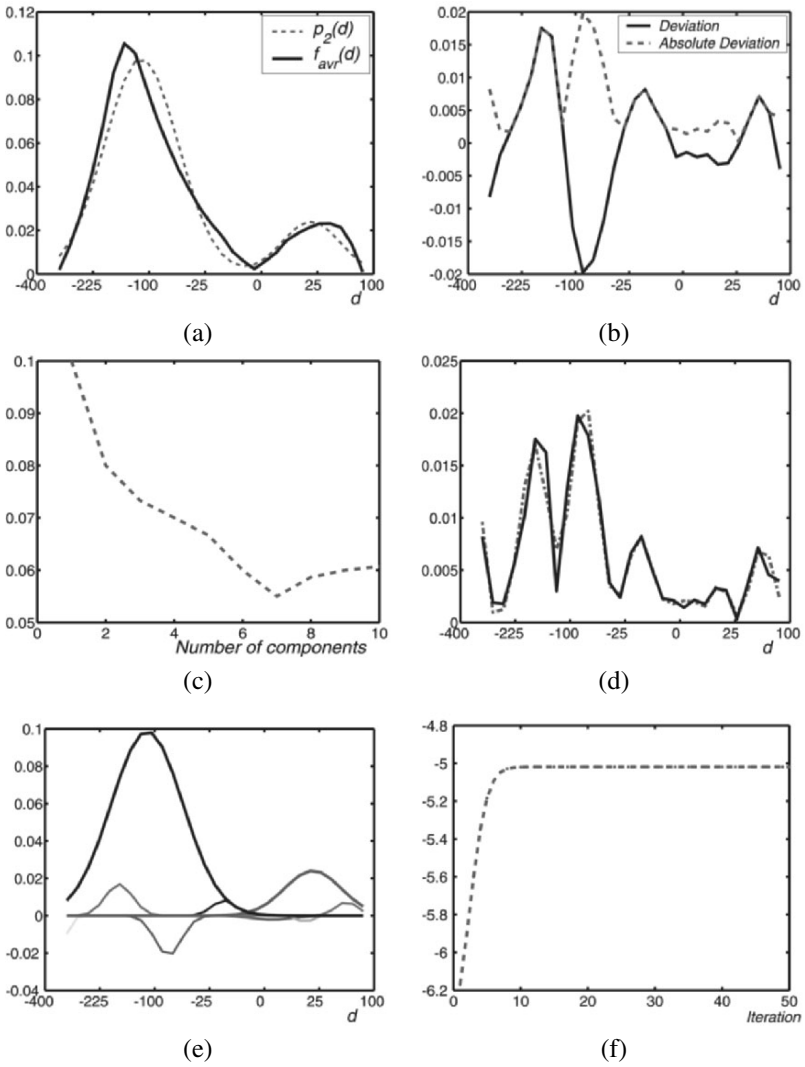
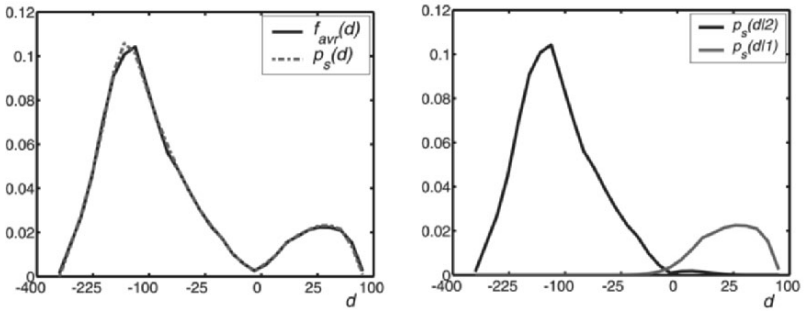


Figure 8. Estimated two dominant modes that present the kidney area and its background (a); the scaled-up absolute deviation of the approximation and its LCDG model (b); approximation error for the scaled absolute deviation as a function of number of subordinate Gaussians (c); density estimation of the scaled-up absolute deviation (d); final LCDG model (e); and log-likelihood changes at the EM iterations (f).



(a) (b)

Figure 9. Final two-class LCDG approximation of the mixed density (a); and final LCDG models for each class (b).

Figure 9 presents the final estimated density using the LCDG model for both mixed density and the marginal density for each class. In Figure 9a the Levy distance between the empirical density and the estimated density is 0.008, which indicates a good match between the empirical distribution and the estimated one.

It is important to note that this shape density is calculated only once; then it is used as it is during kidney segmentation of all other given abdomen images.

5.6. Stepwise Deformable Model Algorithm and Segmentation Results

For any given image, the proposed algorithm of segmenting the region k^* is as follows:

1. Register the given image to one of the aligned images in the database using 2D affine registration based on using mutual information as a similarity measure. This step makes the algorithm invariant to scaling, rotation, or translation.

2. Use the modified EM algorithm to estimate the parameters of the LCDG density model for signed distance map $p_s(d|k)$ inside and outside the object of interest from the average shape which was calculated a priori.

3. Calculate the normalized histogram for the given image.

4. Use the modified EM algorithm to estimate the parameters of the LCDG density model for each class $p_g(q|k)$, k being the class number $k = 1...K$.

5. Initialize the control points $\phi(\tau)$ for the deformable model, and for each control point $\phi(\tau)$ on the current deformable model, calculate sign dis-

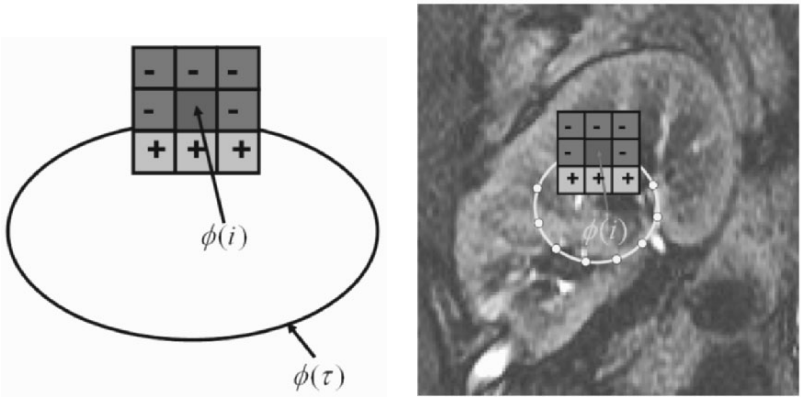


Figure 10. Greedy propagation of the deformable model (a). The deformable model in (a) is initialized in the given kidney to be segmented (b). See attached CD for color version.

tances indicating exterior $(-)$ or interior $(+)$ positions of each of the eight nearest neighbors w.r.t. the contour as shown in Figure 10.

6. Check the label k for each control point:

   (a) If the point is assigned to the region $k = k^*$, then

      i. Estimate the region labels for its neighbors using a Bayesian classifier such that they have the $(-)$ distance.

      ii. If some of these sites are also assigned to the class k^*, then move the control point to the neighboring position ensuring the minimum total energy (i.e., expand the contour).

      iii. Otherwise, do not move this point (the steady state).

   (b) If the point is assigned to the region $k \neq k^*$, then

      i. Estimate the region labels for its neighbors using a Bayesian classifier such that they have the $(+)$ distance.

      ii. Move the control point to the neighboring position ensuring the minimum total energy (i.e., contract the contour).

7. If the iteration adds new control points, use the bicubic interpolation of the whole contour and then smooth all its control points with a lowpass filter.

8. Repeat steps 6 and 7 until no positional changes in the control points occur.

The first step of our approach is to estimate the density from the given image for both the kidney object and its background. Unlike the shape density estimation,

the gray-level density estimation step will be repeated for every image we need to segment; so that our approach is adaptable for all data.

Figure 11 shows the initial approximation of the bimodal empirical distribution of $Q = 256$ gray levels over a typical DCE-MRI (Dynamic Contrast-Enhanced Magnetic Resonance Imaging) slice of human abdomen. The dominant modes represent the brighter kidney area and its darker background, respectively. After the additive and subtractive parts of the absolute deviation are approximated with the DG mixtures, the initial mixed LCDG model consists of the 2 dominant, 4 additive, and 4 subtractive DGs, that is, $C_{\mathrm{p}} = 6$ and $C_{\mathrm{n}} = 4$. The LCDG models of each class are obtained with $t = 78$, ensuring the best class separation.

Figure 12 presents the final LCDG model obtained by refining the above initial one using the modified EM algorithm introduced in Section 5.3. The first 37 iterations of the algorithm increase the log-likelihood of Eq. (11) from -6.90 to -4.49. Also, the Levy distance between the empirical density and the estimated density is 0.007, which indicates a good match between the empirical distribution and the estimated one.

The second step of the proposed approach is the alignment step. Figure 13a demonstrates one of our aligned databases. Figure 13b shows one kidney image that we need to segment. Figure 13c shows the result of the registration of (a) and (b) using MI. Figure 14 shows the final steps of the proposed approach. The resulting segmentation in Figure 14c has an error of 0.23% with respect to radiologist segmentation (ground truth). Figure 15 shows another example of kidney segmentation using the proposed approach. To highlight the accuracy of the proposed approach, we compared the proposed approach with the more conventional geometric deformable model presented in [60], where the level set-based shape prior gives an error of 4.9% (Figure 15g). Similar results for four other kidney images in Figure 16 suggest that the latter approach fails to detect sizeable fractions of the goal objects.

In the above examples, the abdominal images were from different patients to show the applicability of our approach, and also to demonstrate the difficulty of the problem. From here on, we will be using the images of only one patient to illustrate the steps of our approach. The segmentation results of one patient are given in Figure 17, a part of whose sequence was given in Figure 2.

6. MODEL FOR THE LOCAL DEFORMATION

In DCE-MRI sequences, the registration problem arises because of patient movement and breathing. To overcome this problem, we proposed a new approach to handle kidney motion. The proposed approach is based on deforming the segmented kidney over closed equispaced contours (i.e., isocontours) to closely match the prototype. We did not use a free-form deformation based on a B-spline on a square lattice because that requires additional smoothing constraints that lead

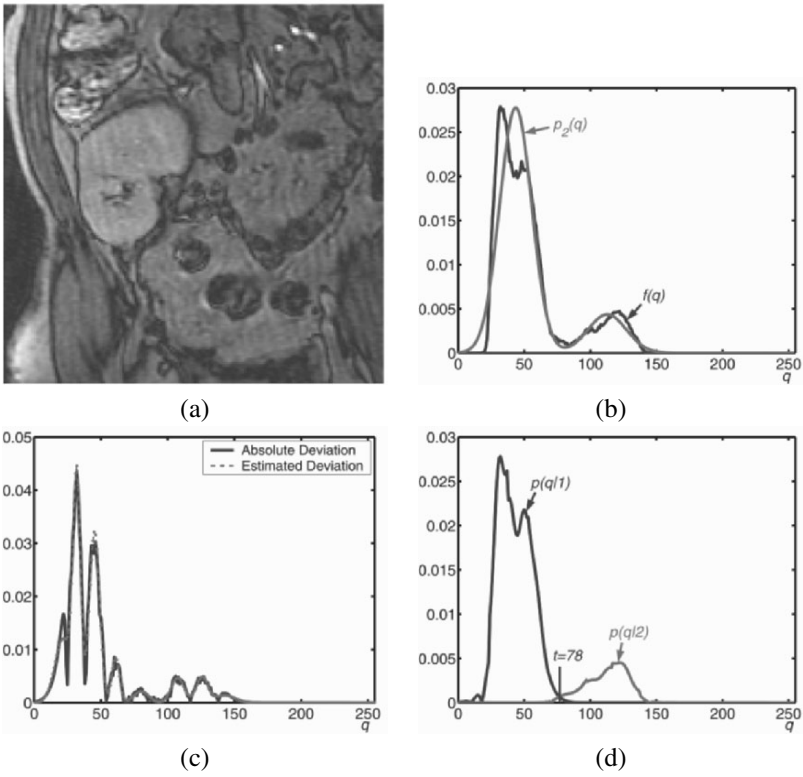


Figure 11. Initial LCDG model of the bimodal empirical gray-level distribution: the DMRI slice (a), its empirical gray-level distribution approximated with the dominant mixture of the DGs (b), the scaled-up absolute deviation of the approximation and its LCDG model (c), and the LCDG model of each class (d) for the best separating threshold $t = 78$. See attached CD for color version.

to very time-consuming computations. Instead we used evolution of the isocontours guided with an exponential speed function in the directions that minimize the distances between corresponding pixel pairs on the isocontours of both the objects. The normalized cross-correlation is used as an image similarity measure insensitive to intensity changes (e.g., due to tissue motion in medical imagery and the contrast agent).

The first step of the proposed approach is to use the fast marching level set methods [46] to generate the distance map inside the kidney regions, as shown in Figure 18a,b. The second step is to use this distance map to generate equally spaced separated contours (isocontours) as shown in Figure 18c,d. Note that the number of isocontours depends on the accuracy and speed that the user needs to achieve.

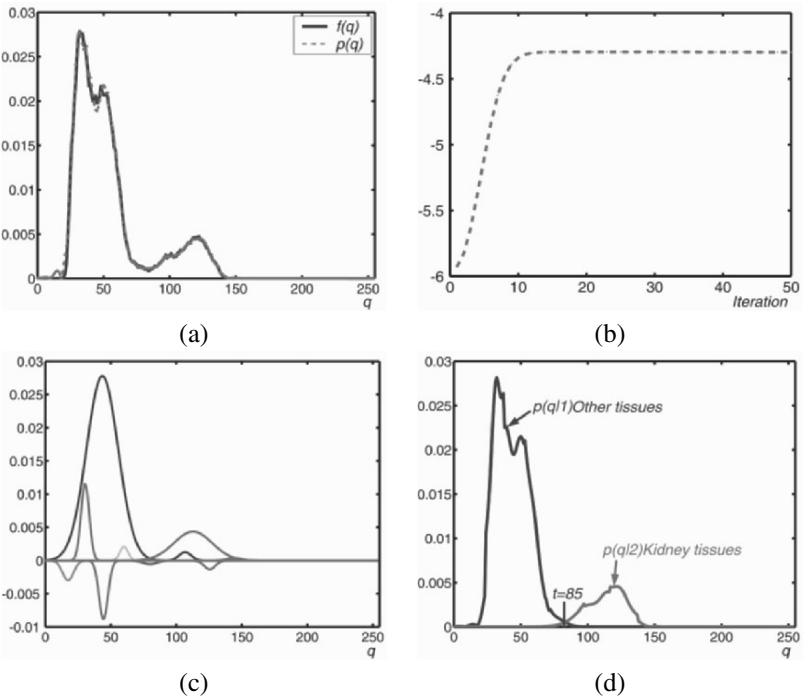


Figure 12. Final 2-class LCDG model (a), log-likelihood changes at the EM iterations (b), ten components of the final LCDG (c), the final LCDG model of each class for the best separating threshold $t = 85$ (d). See attached CD for color version.

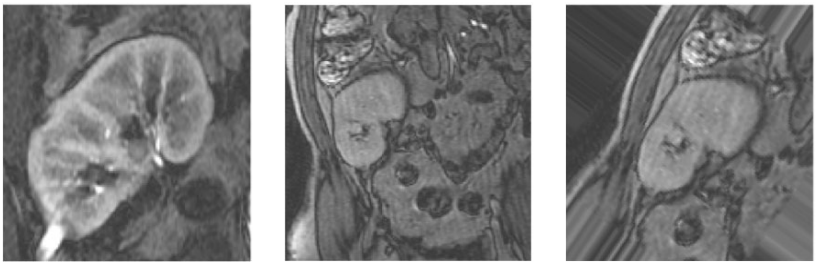


Figure 13. Kidney from the aligned database (a); a kidney to segment (b); alignment of (a) and (b) using affine mutual information registration (c).

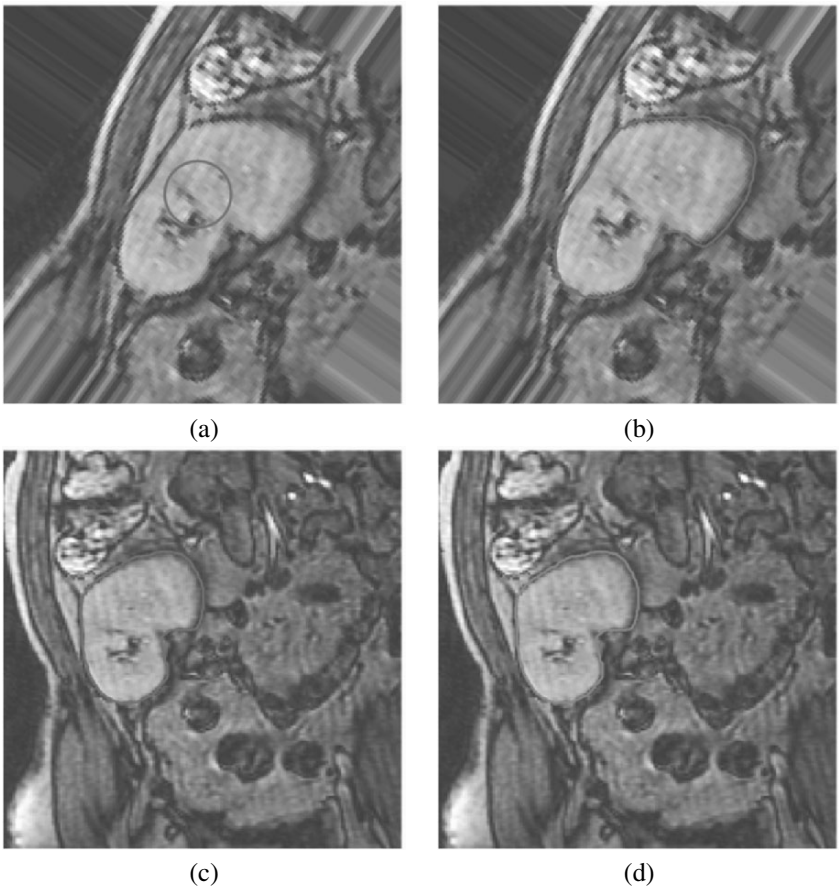


Figure 14. Initialization of deformable model (a); final segmentation of the aligned image (b); final segmentation (c) after multiplying the aligned image by inverse transformation (error = 0.23%); and radiologist segmentation (d). See attached CD for color version.

The third step of the proposed approach is to use normalized cross-correlation to find the correspondence between isocontours. Since we start with aligned images, we limit our searching space to a small window (e.g., 10×10) to improve the speed of the proposed approach. The final step is evolution of the isocontours. Here our goal is to deform the isocontours in the first (target) image to match the isocontours in the second (source) image. Before we discuss the details of the evolution algorithm, let us define the following terminology:

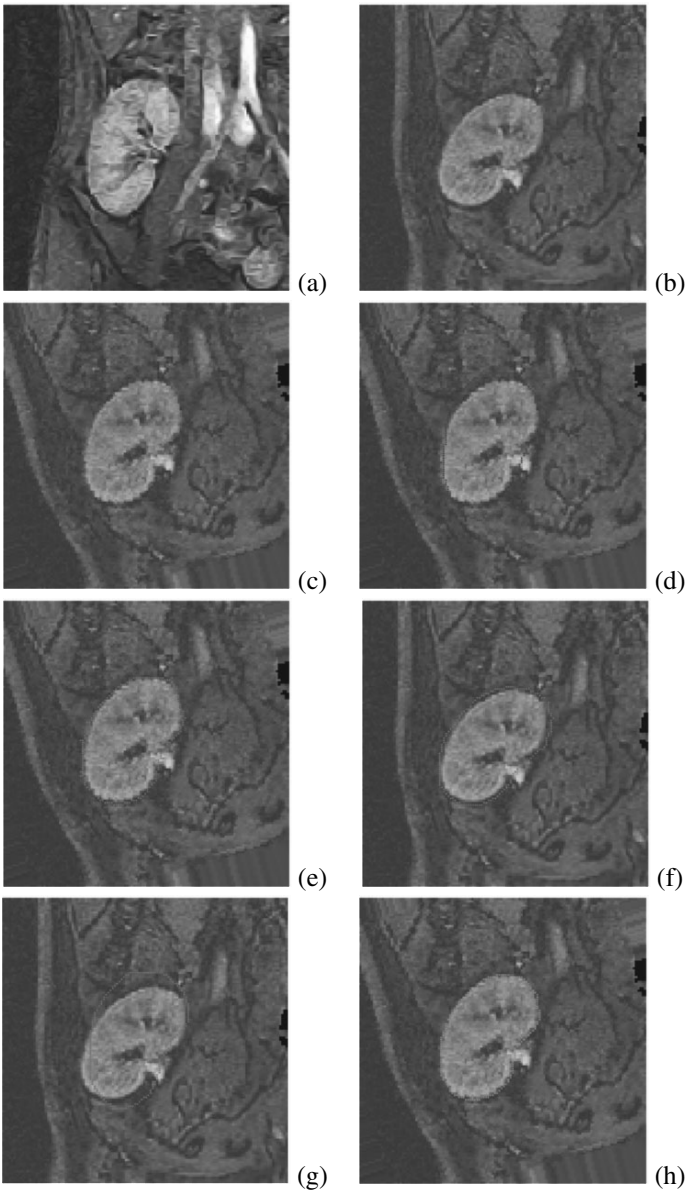


Figure 15. Chosen training kidney prototype (a); an image to be segmented (b); its alignment to the prototype (c); the contour of the training prototype superimposed on the aligned test image (d); the segmentation result (e); the same result (f) after its inverse affine transform to the initial image (b) (total error 0.63% in comparison to ground truth (h); the final boundary and the ground truth are in red and green, respectively), and the segmentation (g) with the algorithm in [60] (the total error 4.9% in comparison to ground truth (h)). See attached CD for color version.

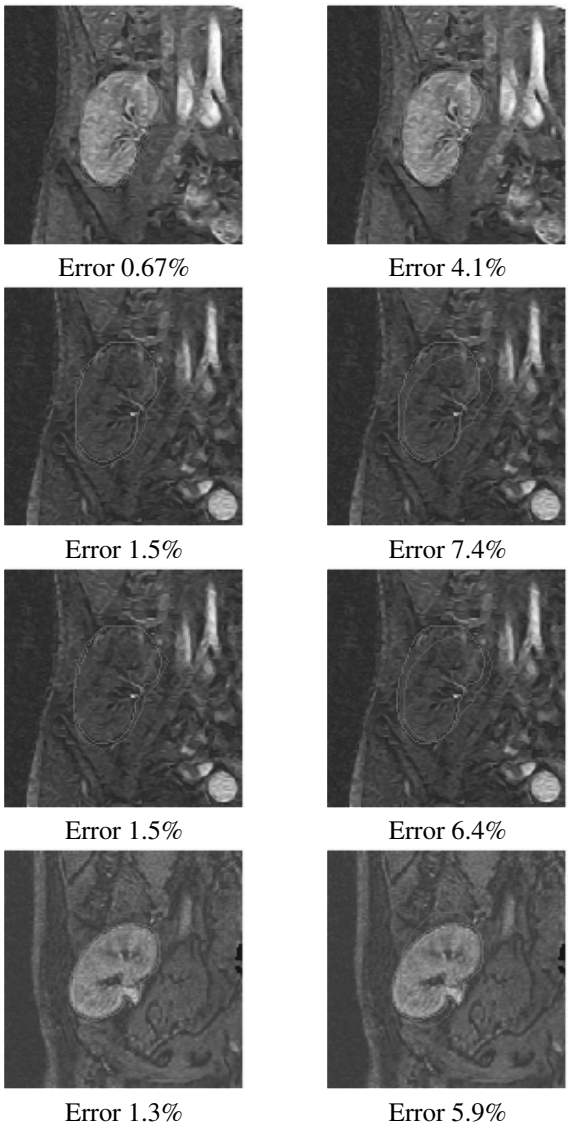


Figure 16. Segmentation of four other kidney DCE-MR images: the left column — our final boundaries and the ground truth (in red and green, respectively); the right column — the segmentation with the algorithm in [60] vs. the ground truth (in red and green, respectively). See attached CD for color version.

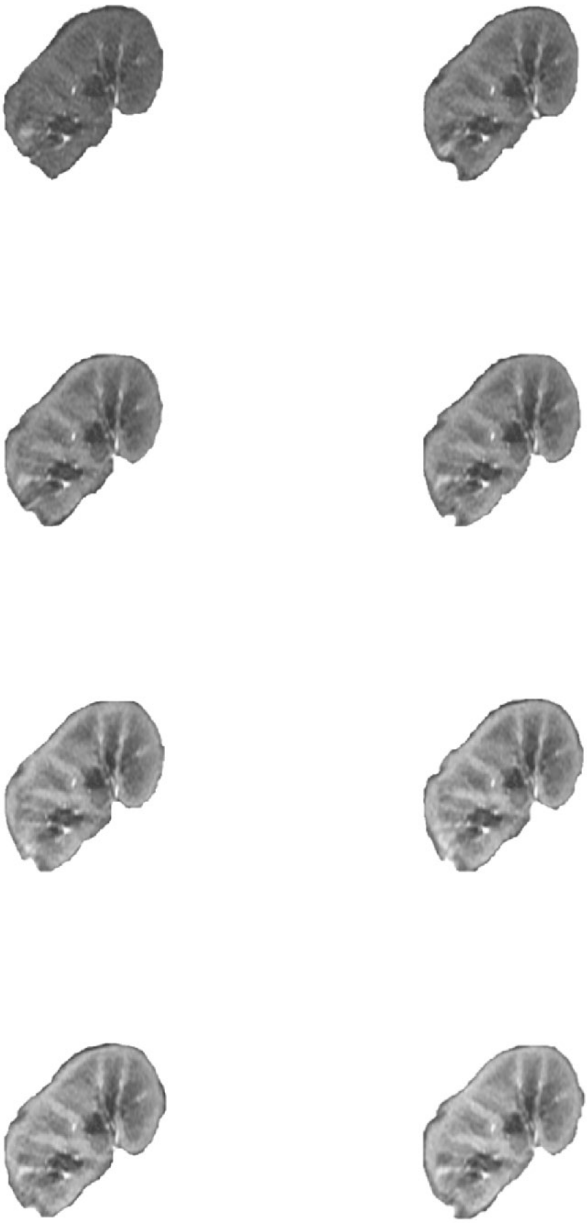


Figure 17. Segmentation results of a DCE-MRI series, a part of which was shown in Figure 2.

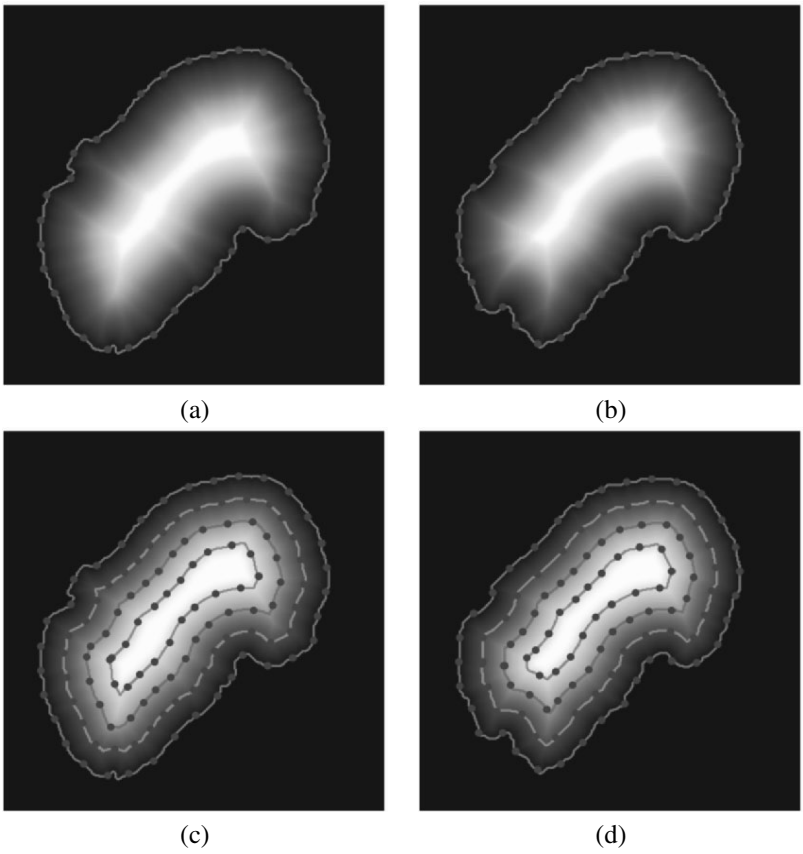
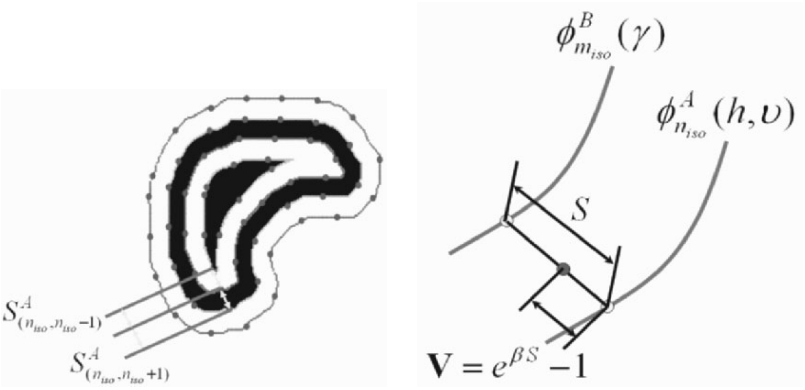


Figure 18. Distance map of two kidneys (a, b) and the samples of isocontours (c, d). See attached CD for color version.

- $\phi^A_{n_{\text{iso}}}(h, \nu)$ are the isocontours in the target image, where $h = 1, \ldots, \mathcal{H}$ is the index of the control points in the given contour, $n_{\text{iso}} = 1, \ldots N_{\text{iso}}$ is the index of the isocontours, and ν is the iteration step.
- $\phi^B_{m_{\text{iso}}}(\gamma)$ are the isocontours in the source image, where $\gamma = 1, \ldots, \Gamma$ is the index of control points in the given contour, and $m_{\text{iso}} = 1, \ldots, M_{\text{iso}}$ is the index of the isocontours.
- S is the Euclidean distance between corresponding points located on both isocontours of both images.
- $S^A_{n_{\text{iso}}, n_{\text{iso}}-1}$ is the Euclidian distance between $\phi^A_{n_{\text{iso}}}(l, \nu)$ and $\phi^A_{n-1}(l, \nu)$.
- $\mathbf{V}$ is the propagation speed function.



(a) (b)

Figure 19. Model constraints (a), and model evolution (b). See attached CD for color version.

The most important step in model propagation is selection of propagation speed function $\mathbf{V}$. Selection of the speed function must satisfy the following conditions:

1. $\mathbf{V} = 0$ if $S = 0$.
2. $\mathbf{V} \leq \min(S, S^A_{n_{\text{iso}},n_{\text{iso}}-1}, S^A_{n_{\text{iso}},n_{\text{iso}}+1})$ if $S > 0$; is the smoothness constraint, which prevents the current point from passing the closest neighbor contour, as shown in Figure 19.

A speed function of the following form satisfies the above conditions:

$$\mathbf{V} = e^{\beta S} - 1, \tag{15}$$

where β is the propagation constant with the upper bound

$$\beta \preceq \frac{\ln\left(\min(S, S^A_{n_{\text{iso}},n_{\text{iso}}-1}, S^A_{n_{\text{iso}},n_{\text{iso}}+1}) + 1\right)}{S}.$$

Based on the speed function shown in Eq. (15) we can deform the isocontours using the following equation, as shown in Figure 19b:

$$\phi^A(h, \nu+1) = \tfrac{\mathbf{V}}{S}\phi^B_{m_{\text{iso}}}(\gamma) + \tfrac{S-\mathbf{V}}{S}\phi^A_{n_{\text{iso}}}(h,\nu) \tag{16}$$

for $h = 1, \ldots, \mathcal{H}$, $m_{\text{iso}} = 1, \ldots, M_{\text{iso}}$, $n_{\text{iso}} = 1, \ldots, N_{\text{iso}}$.

To show the quality of the proposed approach, we fused the two kidney images by means of the checkerboard visualization in Figure 20. It is clear from Figure 20b that the connectivity between the two images at the edges and inside the kidney region are improved after applying the proposed deformation model.

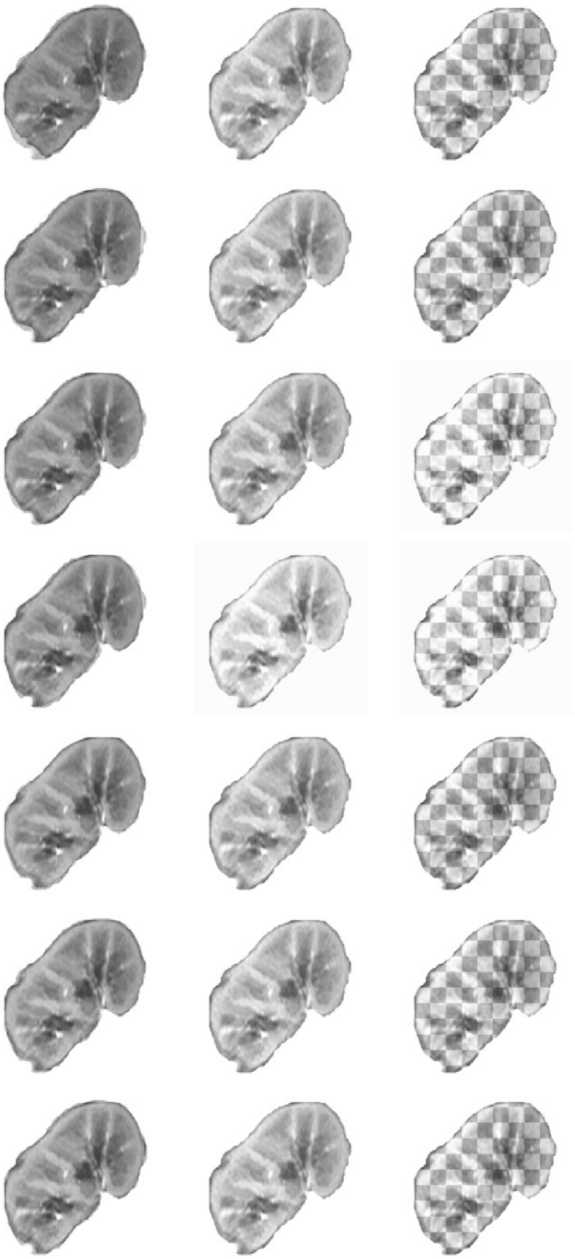


Figure 20. Example of a DCE-MRI series after nonrigid registration.

7. CORTEX SEGMENTATION

Strake et al. [64] have shown that the most important signal characteristics come from the cortex of the kidney during acute rejection. Therefore, the final step of our approach is to segment the cortex from the segmented kidney. To achieve this task, we use the same approach but based now only on intensity. At this step, since all the kidneys are aligned together, we select seed points from the medulla regions, and evolve our deformable model based only on intensity. After we extract the medullary regions, the rest is cortex, which is used as a mask and propagated over the whole sequence to plot the average cortex intensity. In Figure 21 we show the cortex segmentation results. In Figure 21a we manually initialize several deformable models inside the medullar regions, and we allow our deformable model evolve in these regions with gray-level information as shown in Figure 21b. The cortex mask is applied to the rest of the sequence as shown in Figure 21c–h.

8. RESULTS

The ultimate goal of the proposed algorithms is to successfully construct a renogram (mean signals) from DCE-MRI sequences, showing the behavior of the kidney as the contrast agent perfuses through the transplanted organ. In acute rejection patients, the DCE-MRI images show a delayed perfusion pattern and reduced cortical enhancement. We tested the above algorithms on thirty patients, four of which are shown in Figure 22. The normal patient shows the expected abrupt increase to higher signal intensities and the dip with a small slope. The acute rejection patients show a delay in reaching peak signal intensities. From these observations we have been able to conclude that the relative peak signal intensity, the time to peak signal intensity, the slope between the peak and the first minimum, and the slope between the peak and the signal measured from the last image in the sequence are the four major features in the renograms of the segmented kidney undergoing classification.

To distinguish between normal and acute rejection, we used a Bayesian supervised classifier, learning statistical characteristics from a training set for normal and acute rejection. The density estimation required in the Bayes classifier is performed for each feature by using a linear combination of Gaussians (LCDG) with positive and negative components, and their parameters are estimated using a modified EM algorithm that appeared in [65]. In our approach we used 50% of the data for training and the other 50% for testing. For testing data the Bayes classifier succeeded in classifying 13 of 15 correctly (86.67%). For the training data the Bayes classifier classified all of them correctly, so the overall accuracy of the proposed approach is 93.3%. All these values were calculated with respect to biopsy results.

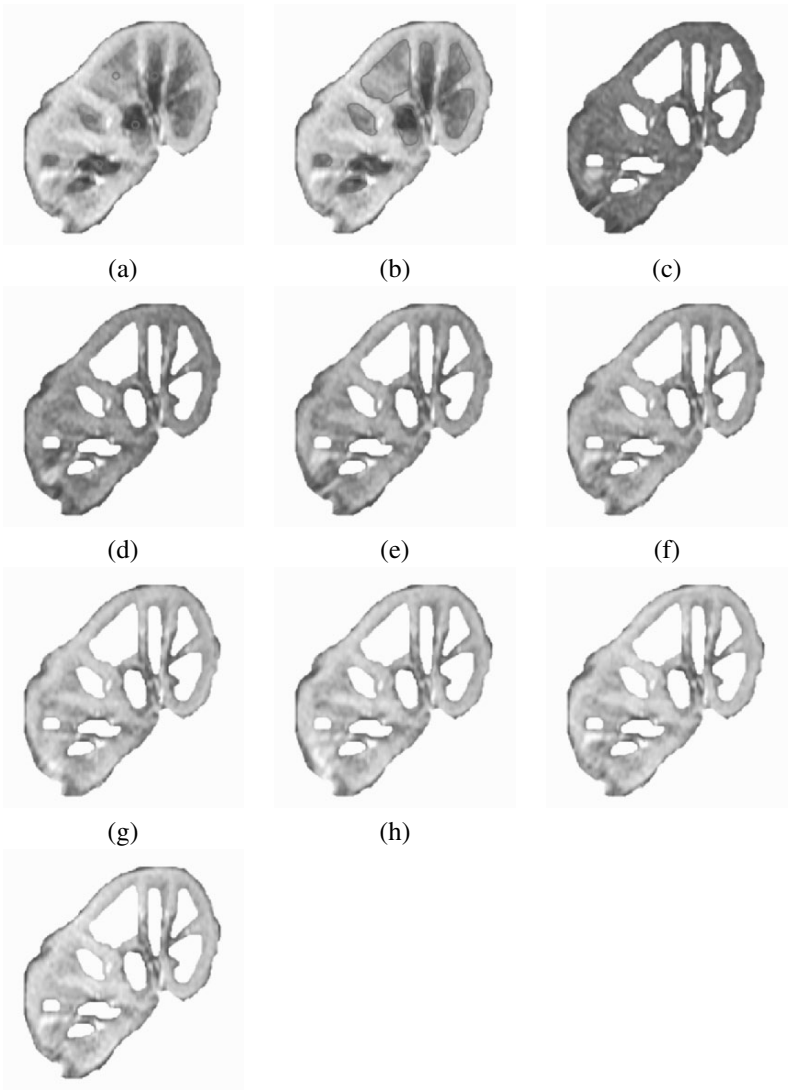


Figure 21. Segmentation of the cortex from kidney images. Several medullary seeds are initialized (a), and the deformable model grows from these seed points (b). After the medulla is extracted from the kidney, the cortex is propagated over the whole sequence of images, as shown in (c)-(h). See attached CD for color version.

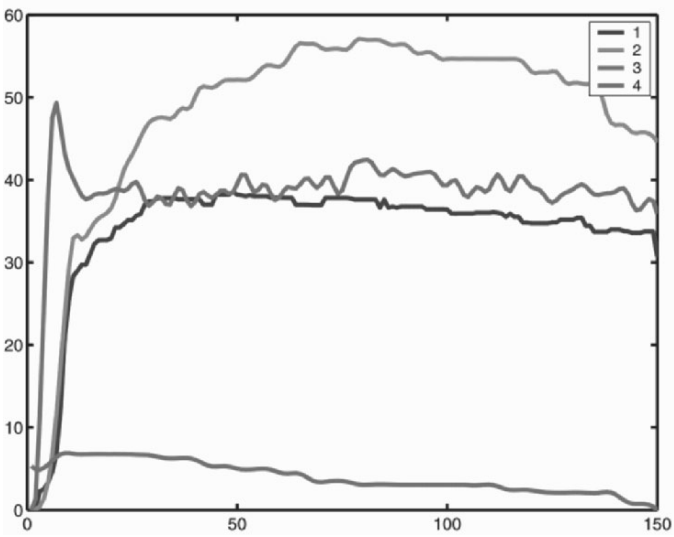


Figure 22. Cortex intensity vs. scan number from 4 subjects. There are 4 seconds between each scan. Subjects 1 and 2 have acute rejection, subject 3 is normal, and subject 4 is chronic glomerulopathy proved by biopsy. In these cortical renograms the normal patient shows the expected abrupt increase in intensity along with a fast decrease, followed by a constant valley and a slow decrease. On the other hand, these abrupt patterns are not seen in acute rejection patients; there is no definite peak, and the time to reach peak intensity is delayed. Subject 4 shows that DCE-MRI is also powerful in distinguishing other diseases. See attached CD for color version.

9. CONCLUSION

In this chapter we presented a framework for the detection of acute renal rejection from Dynamic Contrast Enhanced Magnetic Resonance Images that includes segmentation of kidneys from abdomen images, nonrigid registration, and Bayes classification. For segmentation of kidneys from the abdomen images, we introduced a new deformable model that evolves with both the gray-level information of a given abdomen image, and the shape information obtained from a database of manually segmented kidneys. The energy function of this deformable model is a combination of (i) the gray-level density and (ii) the prior shape information as a 1D density function. For these density estimations we introduced a modified EM algorithm that closely approximates the densities. Following segmentation, we introduced a nonrigid registration algorithm that deforms the kidney object on isocontours instead of a square lattice, which provides more degrees of freedom to obtain accurate deformation. After nonrigid registration, the kidney is segmented into cortex and medulla, and the average gray-level value of the cortex for the whole sequence of a patient is plotted. The features extracted from these signal

plots (renograms) undergo Bayesian classification to understand if the transplanted kidney is undergoing acute rejection or if it is functioning normally.

Our future work will include testing more patients, and testing the robustness of our approach. In addition, we will try to classify other possible diseases that occur after transplantation, and even try to understand the severity of rejection. With such developments we believe that analysis of DCE-MRI imagery has a high potential for replacing the current nuclear imaging tests or invasive biopsy techniques.

10. REFERENCES

1. *2000 annual report of the U.S. scientific registry of transplant recipients and the organ procurement and transplantation network: transplant data 1990–1999*. 2001. Richmond, VA: United Network for Organ Sharing, Richmond, VA.
2. Sharma RK, Gupta RK, Poptani H, Pandey CM, Gujral RB, Bhandari M. 1995. The magnetic resonance renogram in renal transplant evaluation using dynamic contrast-enhanced MR imaging. *Radiology* **59**:1405–1409.
3. Kasiske BL, Keane WF. 1996. Laboratory assessment of renal disease: clearance, urinalysis, and renal biopsy. In *The kidney*, 5th ed., pp. 1137–1173. Ed BM Brenner, FC Rector. Philadelphia: Saunders.
4. Bennett HF, Li D. 1997. MR imaging of renal function. *Magn Reson Imaging Clin North Am* **5**(1):107–126.
5. Giele ELW. 2002. *Computer methods for semi-automatic MR renogram determination*. PhD dissertation. Department of Electrical Engineering, University of Technology, Eindhoven.
6. Taylor A, Nally JV. 1995. Clinical applications of renal scintigraphy. *Am J Roentgenol* **164**:31–41.
7. Katzberg RW, Buonocore MH, Ivanovic M, Pellot-Barakat C, Ryan RM, Whang K, Brock JM, Jones CD. 2001. Functional, dynamic and anatomic MR urography: feasibility and preliminary findings. *Acad Radiol* **8**:1083–1099.
8. Tublin ME, Bude RO, Platt JF. 2003. The resistive index in renal Doppler sonography: where do we stand? *Am J Roentgenol* **180**(4):885–892.
9. Huang J, Chow L, Sommer FG, Li KCP. 2001. Power Doppler imaging and resistance index measurement in the evaluation of acute renal transplant rejection. *J Clin Ultrasound* **29**:483–490.
10. Turetschek K, Nasel C, Wunderbaldinger P, Diem K, Hittmair K, Mostbeck GH. 1996. Power Doppler versus color Doppler imaging in renal allograft evaluation. *J Ultrasound Med* **15**(7):517–522.
11. Trillaud H, Merville P, Tran Le Linh P, Palussiere J, Potaux L, Grenier N. 1998. Color Doppler sonography in early renal transplantation follow-up: resistive index measurements versus power Doppler sonography. *Am J Roentgenol* **171**(6):1611–16115.
12. Yang D, Ye Q, Williams M, Sun Y, Hu TCC, Williams DS, Moura JMF, Ho C. 2001. USPIO enhanced dynamic MRI: evaluation of normal and transplanted rat kidneys. *Magn Reson Med* **46**:1152–1163.
13. Chan L. 1999. Transplant rejection and its treatment. In *Atlas of diseases of the kidney*, Vol. 5, chap. 9. Series Ed RW Schrier. Philadelphia: Current Medicine Inc.
14. Szolar DH, Preidler K, Ebner F, Kammerhuber F, Horn S, Ratschek M, Ranner G, Petritsch P, Horina JH. 1997. Functional magnetic resonance imaging of the human renal allografts during the post-transplant period: preliminary observations. *Magn Reson Imaging* **15**(7):727–735.
15. Lorraine KS, Racusen C. 1999. Acute tubular necrosis in an allograft. *Atlas of diseases of the kidney*, Vol. 1, chap. 10. Series Ed RW Schrier. Philadelphia: Current Medicine Inc.

16. Krestin GP, Friedmann G, Steinbrich W. 1988. Gd-DTPA enhanced fast dynamic MRI of the kidneys and adrenals. *Diagn Imaging Int* **4**:40–44.
17. Krestin GP, Friedmann G, Steinbrich W. 1988. Quantitative evaluation of renal function with rapid dynamic gadolinium-DTPA enhanced MRI. In *Proceedings of the international society for magnetic resonance in medicine*, Book of Abstracts. Los Angeles: MRSTS.
18. Frank JA, Choyke PL, Girton M. 1989. Gadolinium-DTPA enhanced dynamic MR imaging in the evaluation of cisplatinum nephrotoxicity. *J Comput Assist Tomogr* **13**:448–459.
19. Knesplova L, Krestin GP. 1998. Magnetic resonance in the assessment of renal function. *Eur Radiol* **8**:201–211.
20. Choyke PL, Frank JA, Girton ME, Inscoe SW, Carvlin MJ, Black JL, Austin HA, Dwyer AJ. 1989. Dynamic Gd–DTPA-enhanced MR imaging of the kidney: experimental results. *Radiology* **170**:713–720.
21. Sun Y, Jolly M, Moura JMF. 2004. Integrated registration of dynamic renal perfusion MR images. In *Proceedings of the IEEE international conference on image processing*, pp. 1923–1926. Washington, DC: IEEE
22. Yim PJ, Marcos HB, McAuliffe M, McGarry D, Heaton I, Choyke PL. 2001. Registration of time-series contrast enhanced magnetic resonance images for renography. In *Proceedings of the 14th IEEE symposium on computer-based medical systems*, pp. 516–520. Washington, DC: IEEE.
23. Sun Y, Moura JMF, Ho C. 2004. Subpixel registration in renal perfusion MR image sequence. In *Proceedings of an IEEE international symposium on biomedical imaging*, pp. 700–703, Washington, DC: IEEE.
24. Sun Y, Moura JMF, Yang D, Ye Q, Ho C. 2002. Kidney segmentation in MRI sequences using temporal dynamics. In *Proceedings of an IEEE international symposium on biomedical imaging*, pp. 98–101. Washington, DC: IEEE.
25. Gerig G, Kikinis R, Kuoni W, van Schulthess GK, Kubler O. 1992. Semiautomated ROI analysis in dynamic MRI studies, part I: image analysis tools for automatic correction of organ displacements. *IEEE Trans Image Process* **11**:(2):221–232.
26. von Schulthess GK, Kuoni W, Gerig G, Duewell S, Krestin G. 1991. Semiautomated ROI analysis in dynamic MRI studies, part II: application to renal function examination, first experiences. *J Comput Assist Tomogr* **2**:733–741.
27. Liang Z, Lauterbur PC. 1994. An efficient method for dynamic magnetic resonance imaging. *IEEE Trans Med Imaging* **13**(4):677–686.
28. Giele ELW, de Priester JA, Blom JA, den Boer JA, van Engelshoven JMA, Hasman A, Geerlings M. 2001. Movement correction of the kidney in dynamic MRI scans using FFT phase difference movement detection. *J Magn Reson Imaging* **14**(6):741–749.
29. Vosshenrich R, Kallerhoff M, Grone HJ, Fischer U, Funke M, Kopka L, Siebert G, Ringert RH, Grabbe E. 1996. Detection of renal ischemic lesions using Gd-DTPA enhanced turbo flash MRI: experimental and clinical results. *J Comput Assist Tomogr* **20**(2):236–243.
30. Munechika H, Sullivan DC, Hedlund LW, Beam CA, Sostman HD, Herfkens RJ, Pelc NJ. 1991. Evaluation of acute renal failure with magnetic resonance imaging using gradient-echo and Gd-DTPA. *Invest Radiol* **26**(1):22–27.
31. Carvlin MJ, Arger PH, Kundel HL, Axel L, Dougherty L, Kassab EA, Moore B. 1987. Acute tubular necrosis: use of gadolinium-DTPA and fast MR imaging to evaluate renal function in the rabbit. *J Comput Assist Tomogr* **11**(3):488–95.
32. Dalla-Palma L, Panzetta G, Pozzi-Mucelli RS, Galli G, Cova M, Meduri S. 2000. Dynamic magnetic resonance imaging in the assessment of chronic medical nephropathies with impaired renal function. *Eur Radiol* **10**(2):280–286.
33. Kikinis R, von Schulthess GK, Jager P, Durr R, Bino M, Kuoni W, Kubler O. 1987. Normal and hydronephrotic kidney: evaluation of renal function with contrast-enhanced MR imaging. *Radiology* **165**(3):837–842.
34. Semelka RC, Hricak H, Tomei E, Floth A, Stoller M. 1990. Obstructive nephropathy: evaluation with dynamic Gd-DTPA-enhanced MR imaging. *Radiology* **175**:797–803.

35. Beckmann N, Joergensen J, Bruttel K, Rudin M, Schuurman HJ. 1996. Magnetic resonance imaging for the evaluation of rejection of a kidney allograft in the rat. *Transpl Int* 9(3):175–83.
36. Preidler KW, Szolar D, Schreyer H, Ebner F, Kern R, Holzer H, Horina JH. 1996. Differentiation of delayed kidney graft function with gadolinium-DTPA-enhanced magnetic resonance imaging and Doppler ultrasound. *Invest Radiol* **31**(6):364–371.
37. El-Diasty T, Mansour O, Farouk A. 2003. Diuretic contrast enhanced mru versus ivu for depiction of non-dilated urinary tract. *Abd Imaging* 28:135–145.
38. Laurent D, Poirier K, Wasvary J, Rudin M. 2002. Effect of essential hypertension on kidney function as measured in rat by dynamic MRI. *Magn Reson Med* **47**(1):127–131.
39. Krestin GP. 1994. Magnetic resonance imaging of the kidneys: current status. *Magn Reson Q* **10**:2–21.
40. Sun Y. 2004. *Registration and segmentation in perfusion MRI: kidneys and hearts*. PhD dissertation, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburg.
41. de Priester JA, Kessels AG, Giele EL, den Boer JA, Christiaans MHL, Hasman A, van Engelshoven JMA. 2001. MR renography by semiautomated image analysis: performance in renal transplant recipients. *J Magn Reson Imaging* **14**(2):134–140.
42. Boykov Y, Lee VS, Rusinek H, Bansal R. 2001. Segmentation of dynamic N–D data sets via graph cuts using Markov models. In *Proceedings of the 4th international conference on medical image computing and computer-assisted intervention (MICCAI)*. Lecture Notes in Computer Science, Vol. 2208, pp. 1058–1066. Utrecht: Springer.
43. Sun Y, Yang D, Ye Q, Williams M, Moura JMF, Boada F, Liang Z, Ho C. 2003. Improving spatiotemporal resolution of USPIO-enhanced dynamic imaging of rat kidneys. *Magn Reson Imaging* **21**:593–598.
44. Chan TF, Vese LA. 2001. Active contours without edges. *IEEE Trans Image Process* **10**(2):266–277.
45. Ibanez L, Schroeder W, Ng L, Cates J, and the Insight Software Consortium. 2005. *The ITK software guide*. Clifton Park, NY: Kitware Inc.
46. Sethian JA. 1996. *Level set methods and fast marching methods*. Cambridge: Cambridge UP.
47. Caselles V, Kimmel R, Sapiro G. 1997. Geodesic active contours. *Int J Comput Vision* **22**(1):61–79.
48. Rousson M, Paragios N. 2002. Shape priors for level set representations. In *Proceedings of the 7th European conference on computer vision, part II (ECCV'02)*. Lecture Notes in Computer Science, Vol. 2751, pp. 78–92. Berlin: Springer.
49. Leventon M, Grimson WL, Faugeras O. 2000. Statistical shape influence in geodesic active contours. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1316–1324. Washington, DC: IEEE Computer Society.
50. Chen Y, Thiruvenkadam S, Tagare H, Huang F, Wilson D. 2001. On the incorporation of shape priors into geometric active contours. In *IEEE workshop on variational and level set methods*, pp. 145–152, Washington, DC: IEEE.
51. Tsai A, Yezzi AJ, Wells WM, Tempany C, Tucker D, Fan A, Eric W, Grimson L, Willsky AS. 2001. Model-based curve evolution technique for image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 463–468, Washington, DC: IEEE Computer Society.
52. Paragios N. 2003. A level set approach for shape-driven segmentation and tracking of the left ventricle. *IEEE Trans Med Imaging* **22**:773–776.
53. Litvin A, Karl WC. 2003. Level set-based segmentation using data driven shape prior on feature histograms. In *IEEE workshop on statistical signal processing*, pp. 166–169. Washington, DC: IEEE.
54. Tsai A, Wells W, Warfield SK, Willsky AS. 2004. Level set methods in an em framework for shape classification and estimation. In *Proceedings of the international conference on medical image computing and computer-assisted intervention (MICCAI)*. Lecture Notes in Computer Science, Vol. 2211, pp. 1–9. Utrecht: Springer.

55. Yang J, Duncan J. 2004. 3d image segmentation of deformable objects with joint shape-intensity prior models using level sets. *Med Image Anal* **8**:285–294.
56. Yuksel SE, El-Baz A, Shi H, Farag AA, El-Ghar MEA, Eldiasty TA, Ghoneim MA. 2005. Automatic detection of renal rejection after kidney transplantation. In *Proceedings of the conference on computer assisted radiology and surgery (CARS)*, pp. 773–778. Berlin: Springer.
57. Witkin A, Kass M, Terzopoulos D. 1987. Snakes: Active contour models. *Int J Comput Vision* **1**:321–331.
58. El-Baz A, Yuksel SE, Shi H, Farag AA, El-Ghar MA, Eldiasty T, Ghoneim MA. 2005. 2d and 3d shape-based segmentation using deformable models. In *Proceedings of the international conference on medical image computing and computer-assisted intervention (MICCAI)*. Lecture Notes in Computer Science, Vol. 2212,pp. 821–829. Utrecht: Springer.
59. Viola P, Wells WM. 1995. Alignment by maximization of mutual information. In *Proceedings of the 5th international conference on computer vision*, pp. 16–23. Washington, DC: IEEE Computer Society.
60. Tsai A, Yezzi A, Wells W, Tempany C, Tucker D, Fan A, Grimson E, Willsky A. 2003. A shape-based approach to curve evolution for segmentation of medical imagery. *IEEE Trans Med Imaging* **22**(2):137–154.
61. Webb A. 2002. *Statistical pattern recognition*. 2nd. ed. Chichester: J. Wiley & Sons.
62. Schlesinger MI. 1968. A connection between supervised and unsupervised learning in pattern recognition. *Kibernetika* **2**:81–88.
63. Lamperti JW. 1996. *Probability*. New York: J. Wiley & Sons.
64. te Strake L, Kool LJS, Paul LC, Tegzess AM, Weening JJ, Hermans J, Doornbos J, Bluemm RG, Bloem JL. 1988. Magnetic resonance imaging of renal transplants: its value in the differentiation of acute rejection and cyclosporin A nephrotoxicity. *Clin Radiol* **39**(3):220–228.
65. Farag, AA, El-Baz A, Gimel'farb G. 2004. Density estimation using modified expectation-maximization for a linear combination of gaussians. In *Proceding of the IEEE international conference on image processing*, Vol. 3, pp. 1871–1874. Washington, DC: IEEE Computer Society.

11

PHYSICALLY AND STATISTICALLY BASED DEFORMABLE MODELS FOR MEDICAL IMAGE ANALYSIS

Ghassan Hamarneh and Chris McIntosh

School of Computing Science, Simon Fraser University
Burnaby, British Columbia, Canada

Medical imaging continues to permeate the practice of medicine, but automated yet accurate segmentation and labeling of anatomical structures continues to be a major obstacle to computerized medical image analysis. Deformable models, with their roots in estimation theory, optimization, and physics-based dynamical systems, represent a powerful approach to the general problem of medical image segmentation. This chapter presents an introduction to deformable models, beginning with the classical Active Contour Models (ACMs), or snakes, and focusing on explicit, physics-based methods. Snakes are useful for segmenting amorphous shapes when little or no prior knowledge about shape and motion is available. Many extensions of snakes incorporate such additional knowledge. An example presented in this chapter is the use of optical flow forces to incorporate knowledge of shape dynamics and guide the snake deformations to track the leading edge of an injected contrast agent in an echocardiographic image sequence. Active Shape Models (ASMs), or smart snakes, is a powerful method for incorporating statistical models of shape variability in the segmentation process. ASMs and ACMs offer different advantages, and, as such, a method combining both is presented. Statistical knowledge about shape dynamics is useful for segmenting and tracking objects with distinctive motion patterns (such as a beating heart). An extension of the ASM to model knowledge of spatiotemporal constraints is presented.

1. ENERGY-MINIMIZING DEFORMABLE MODELS

1.1. Introduction

The classical deformable model, the Active Contour Model or snakes [1, 2], gained wide acceptance as a segmentation tool due to its robustness to image

Address all correspondence to: Dr. Ghassan Hamarneh, School of Computing Science, Simon Fraser University, 8888 University Drive, Burnaby, BC, V5A 1S6, Canada. Phone: +1.604.291.3007, Fax: +1.604.291.3045, hamarneh@cs.sfu.ca.

noise and boundary gaps. Ideas related to snakes date back to the early 1970s [3, 4]. In short, an ACM is an energy-minimizing contour with smoothness constraints, deformed according to image data. They integrate boundary elements into a single, inherently connected, smooth, mathematically well-defined structure, which can be implemented on the continuum achieving sub-pixel accuracy. ACMs were originally designed to be semiautomatic tools supporting intuitive interaction mechanisms for guiding the segmentation process. In active contour models, a contour is initialized on the image and left to deform in a way that, first, moves it toward features of interest in the image and, second, maintains a certain degree of smoothness in the contour. Consequently, an energy term is associated with the contour and is designed to be inversely proportional to both the contour's smoothness and its fit to the image data, in order to segment the desired image features. Deformation of the contour in the image will change its energy; thus, one can imagine an energy surface on top of which the contour moves (in a way that resembles the slithering of a snake, and hence the name) while seeking valleys of low energy [5].

1.2. Classical Snakes

A 2D snake in a continuous spatial domain is represented as a 2D parametric contour $\mathbf{v}(s) = (x(s), y(s))$, where $s \in [0, 1]$. In order to fit the snake model to the image data we associate energy terms with the snake and aim to deform the snake in a way that minimizes its total energy. The energy of the snake, ξ, depends on both the shape of the contour and the image data $I(x, y)$ reflected via the internal and external energy terms, $\alpha(\mathbf{v}(s))$ and $\beta(\mathbf{v}(s))$, respectively. The total snake energy is written as

$$\xi(\mathbf{v}(s)) = \alpha(\mathbf{v}(s)) + \beta(\mathbf{v}(s)). \tag{1}$$

The internal energy term is given by

$$\alpha(\mathbf{v}(s)) = \int_0^1 w_1(s) \left| \frac{\partial \mathbf{v}(s)}{\partial s} \right|^2 + w_2(s) \left| \frac{\partial^2 \mathbf{v}(s)}{\partial s^2} \right|^2 ds, \tag{2}$$

whereas the external energy term is given as

$$\beta(\mathbf{v}(s)) = \int_0^1 w_3(s) P(\mathbf{v}(s)) ds. \tag{3}$$

Weighting functions w_1 and w_2 control the tension and flexibility of the contour, respectively, and w_3 controls the influence of the image data. w_i can depend on s but are typically set to different constants. For the contour to be attracted to

image features, function $P(\mathbf{v}(s))$ is designed such that it has minima where the features have maxima. For example, for the contour to be attracted to high-intensity changes (high gradients), we can choose

$$P(\mathbf{v}(s)) = P(x(s), y(s)) = -\|\nabla[G_\sigma * I(x,y)]\|, \tag{4}$$

where $G_\sigma * I$ denotes the image convolved with a smoothing (e.g., Gaussian) filter with a parameter σ controlling the extent of the smoothing (e.g., variance of Gaussian).

The contour $\mathbf{v}$ that minimizes energy ξ must, according to the calculus of variations [6], satisfy the vector-valued partial differential (Euler-Lagrange) equation:

$$-\frac{\partial}{\partial s}\left(w_1\frac{\partial \mathbf{v}}{\partial s}\right) + \frac{\partial^2}{\partial s^2}\left(w_2\frac{\partial^2 \mathbf{v}}{\partial s^2}\right) + w_3\nabla P(\mathbf{v}(s)) = \mathbf{0}. \tag{5}$$

1.3. Dynamic Deformable Models

In order to attack the problem of tracking non-rigid time-varying objects, deformable models were extended to dynamic deformable models. These describe the shape changes in a single model that evolves through time to reach a state of equilibrium where internal forces representing constraints on shape smoothness balance the external image forces and the contour comes to rest [7]. In this case the time-varying (dynamic) contour is written as $\mathbf{v}(s,t) = (x(s,t), y(s,t))$, where $s \in [0,1]$ and the corresponding constraint equation becomes

$$\mu(s)\frac{\partial^2 \mathbf{v}}{\partial t^2} + \gamma(s)\frac{\partial \mathbf{v}}{\partial t} - \frac{\partial}{\partial s}\left(w_1\frac{\partial \mathbf{v}}{\partial s}\right) + \frac{\partial^2}{\partial s^2}\left(w_2\frac{\partial^2 \mathbf{v}}{\partial s^2}\right) + w_3\nabla P(\mathbf{v}(s,t)) = \mathbf{0}, \tag{6}$$

where $\mu(s)$ and $\gamma(s)$ are the mass and damping densities, respectively.

1.4. Snakes Drawbacks

Snakes do not, however, solve the entire problem of finding contours in images. This is mainly because they lack high-level control and depend on other mechanisms, such as interaction with the user or some higher-level image understanding process, and information from image data adjacent in time or space [8]. Snakes are sensitive to initialization and parameter selection (e.g., w_i in (2) and (3), or σ in (4)) and therefore rely on human guidance to specify approximate shape and starting position for the snake somewhere near the desired contour as well as suitable values of the snake's parameters (see Figures 1 and 2). Without such guidance snakes can leak or latch onto erroneous edges. In addition to user interaction, other approaches have been proven useful to deal with the snakes' limited capture range problem and difficulties in progressing into boundary concavities. These include the incorporation of inflation forces [9], multi-resolution

search [10], an initial step to optimize for rigid parameters, distance transform-based potential functions, and Gradient Vector Flow Fields (see Section 1.6.5). In the next chapter, deformable organisms are introduced as an attempt to provide a simple model of the human expert's cognitive abilities in order to guide the model's deformations.

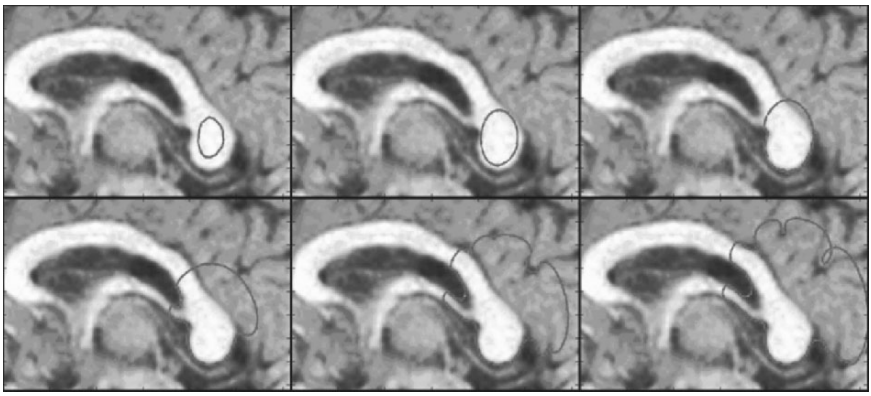


Figure 1. Sample frames (progressing left to right, top to bottom) showing incorrect progress of a deformable model (snake) for segmenting the corpus callosum (CC) in a midsagittal brain magnetic resonance image (MRI), due to the wrong choice of parameters. See attached CD for color version.

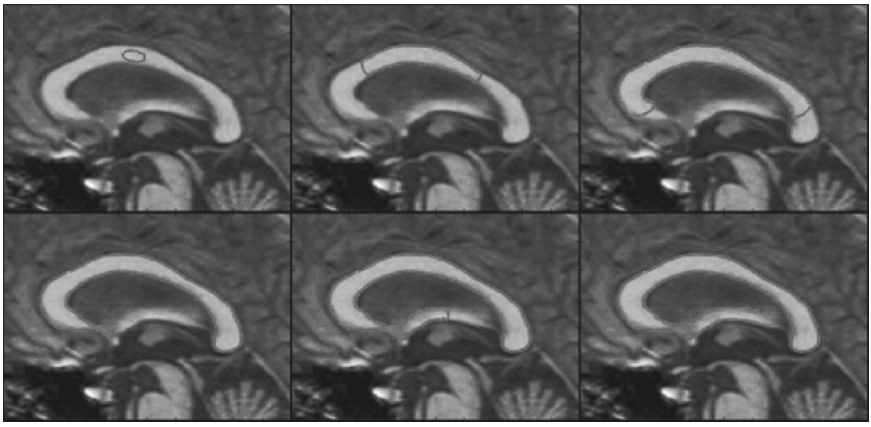


Figure 2. Sample frames (progressing left to right, top to bottom) showing incorrect progress of a snake segmenting the CC in an MRI image. Leaking of the snake occurs because of the weak edge strength (lower left) and incorrect parameter setting. See attached CD for color version.

1.5. Discretization and Numerical Simulation

For a polygonal snake contour the discrete version of (6) can be written as

$$\mu_i \ddot{\mathbf{v}}_i(t) + \gamma_i \dot{\mathbf{v}}_i(t) - w_1 \mathbf{v}_i''(t) + w_2 \mathbf{v}_i'''' (t) + w_3 \nabla P(\mathbf{v}_i(t)) = \mathbf{0}, \quad (7)$$

where $\{\mathbf{v}_i(t) = (x_i(t), y_i(t))\}_{i=1,2,\ldots,N}$ are the nodes of the snake polygon, t is used as the discrete time variable, and i is the snake node index. $\dot{\mathbf{v}}$ and $\ddot{\mathbf{v}}$ are the first and second derivates of $\mathbf{v}$ with respect to t. $\mathbf{v}''$ and $\mathbf{v}''''$ are the second and fourth derivates of $\mathbf{v}$ with respect to i. Setting the mass density to zero[1] ($\mu_i = \mu = 0$) and the damping density to a constant ($\gamma_i = \gamma$), we rewrite Eq. (7) for simulating the deformations of the discrete snake as

$$\gamma \dot{\mathbf{v}}_i - w_1 \mathbf{F}_i^{\text{tensile}}(t) + w_2 \mathbf{F}_i^{\text{flexural}}(t) = w_3 \mathbf{F}_i^{\text{external}}(t). \quad (8)$$

$\mathbf{F}_i^{\text{tensile}}(t)$ is a tensile force (resisting stretching) acting on node i at time t and is given by

$$\mathbf{F}_i^{\text{tensile}}(t) = 2\mathbf{v}_i(t) - \mathbf{v}_{i-1}(t) - \mathbf{v}_{i+1}(t). \quad (9)$$

$\mathbf{F}_i^{\text{flexural}}(t)$ is a flexural force (resisting bending) and is given by

$$\mathbf{F}_i^{\text{flexural}}(t) = 2\mathbf{F}_i^{\text{tensile}}(t) - \mathbf{F}_{i-1}^{\text{tensile}}(t) - \mathbf{F}_{i+1}^{\text{tensile}}(t). \quad (10)$$

$\mathbf{F}_i^{\text{external}}(t)$ is an external (image-derived) force. It is derived in a way that causes the snake node to move toward regions of higher intensity gradient in the image and is given by

$$\mathbf{F}_i^{\text{external}}(t) = \nabla P(x_i(t), y_i(t)), \quad (11)$$

where $P(x_i(t), y_i(t))$ is given in (4).

The equation used for updating the position of any snake node i can be obtained from (8) by using a finite-difference derivative approximation $\dot{\mathbf{v}}_i = (\mathbf{v}_i(t + \Delta t) - \mathbf{v}_i(t))/\Delta t$, where Δt is a finite time step, yielding

$$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) - \frac{\Delta t}{\gamma}(w_1 \mathbf{F}_i^{\text{tensile}}(t) - w_2 \mathbf{F}_i^{\text{flexural}}(t) + w_3 \mathbf{F}_i^{\text{external}}(t)). \quad (12)$$

1.6. Extensions

1.6.1. Inflation Force

In addition to the above forces, an inflation force, $\mathbf{F}_i^{\text{inflation}}(t)$, can be utilized to allow for initializing the snake farther away from the target boundary.

[1] In static shape recovery problems not involving time-varying data, the mass density is often set to zero, resulting in simplified equations of motion and a snake that comes to rest as soon as the internal forces balance the external forces [11].

$\mathbf{F}_i^{\text{inflation}}(t)$ is given by

$$\mathbf{F}_i^{\text{inflation}}(t) = F\left(I_s\left(x_i, y_i\right)\right)\mathbf{n}_i(t), \tag{13}$$

where I_s is a smoothed version of I, $\mathbf{n}_i(t)$ is the unit vector in the direction normal to the contour at node i and the binary function

$$F\left(I\left(x,y\right)\right) = \begin{cases} +1 & if\ I\left(x,y\right) \geq T \\ -1 & \text{otherwise.} \end{cases} \tag{14}$$

links the inflation force to the image data, where T is an image intensity threshold. More elaborate rules can be used (e.g., using region-based image intensity statistics [12]). Consequently, Eq. (8) becomes

$$\gamma\dot{\mathbf{v}}_i - w_1\mathbf{F}_i^{\text{tensile}}(t) + w_2\mathbf{F}_i^{\text{flexural}}(t) = w_3\mathbf{F}_i^{\text{external}}(t) + q\mathbf{F}_i^{\text{inflation}}(t), \tag{15}$$

where q is a scalar weighting the inflation force. Subsequently, (12) becomes

$$\begin{aligned} \mathbf{v}_i(t+\Delta t) &= \mathbf{v}_i(t) - \frac{\Delta t}{\gamma}(w_1\mathbf{F}_i^{\text{tensile}}(t) - w_2\mathbf{F}_i^{\text{flexural}}(t) \\ &\quad + w_3\mathbf{F}_i^{\text{external}}(t) + q\mathbf{F}_i^{\text{inflation}}(t)). \end{aligned} \tag{16}$$

1.6.2. Adaptive Inflation

In order to dampen the inflation force when the snake nodes reach the target boundary, a node-specific inflation weight q_i can be used, where i is the snake node number. Having only a single value for the inflation force for all the nodes is insufficient, since this causes the nodes that reach the target boundary earlier than others to pass over it and cause the snake to leak. If a node reaches the target boundary the inflation direction is reversed (inflation becomes deflation and vice versa), and if a certain number of inflation reversals occur within a limited number of past iterations, then the inflation force is dampened (q_i is replaced by αq_i, $\alpha < 1$) for that particular node only. This updated Eq. (16) becomes

$$\begin{aligned} \mathbf{v}_i(t+\Delta t) &= \mathbf{v}_i(t) - \frac{\Delta t}{\gamma}(w_1\mathbf{F}_i^{\text{tensile}}(t) - w_2\mathbf{F}_i^{\text{flexural}}(t) \\ &\quad + w_3\mathbf{F}_i^{\text{external}}(t) + q_i\mathbf{F}_i^{\text{inflation}}(t)). \end{aligned} \tag{17}$$

1.6.3. Adaptive Subdivision Scheme

Different subdivision schemes can be applied, where the number of snake nodes is adapted based on the distance between nodes and the curvature along the snake. A polygonal snake can start with only a few nodes and then increase the number of nodes as it deforms to accommodate for a longer, concavity seeking snake. Figure 3 presents one possible implementation, while Figure 4 shows a snake segmentation example that utilizes adaptive inflation and subdivision.

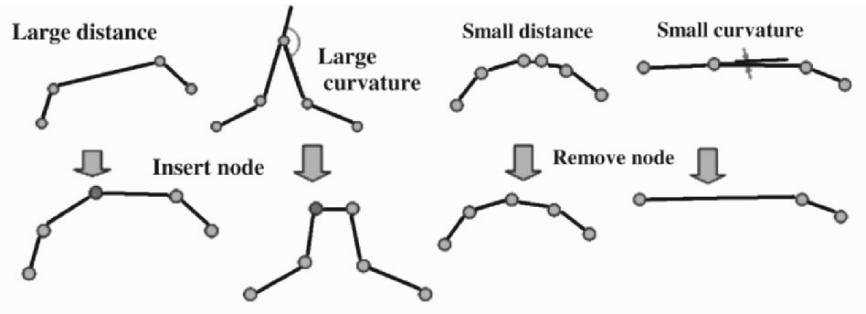


Figure 3. Adaptive resampling of a polygonal snake. (Left) Adding a node if the distance between consecutive nodes is large or curvature is high. (Right) Removing a node if the distance between nodes is small or the curvature is small (appropriate distance and curvature thresholds must be chosen). See attached CD for color version.

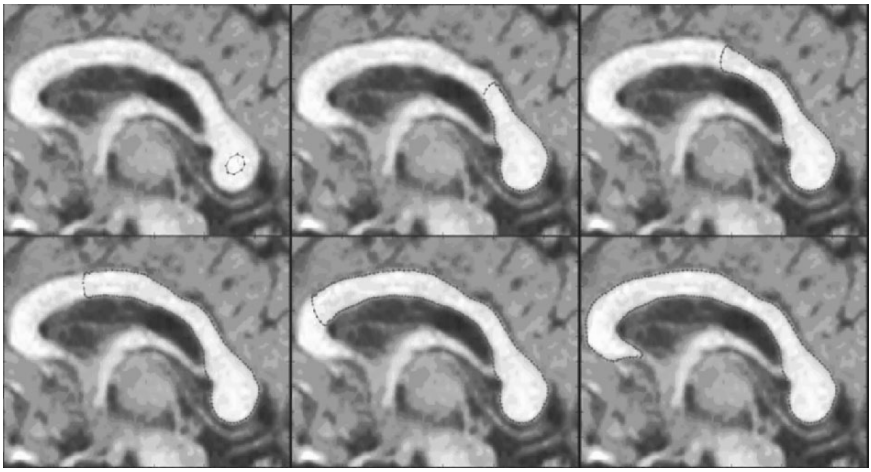


Figure 4. Sample frames (progressing left to right, top to bottom) showing progress of correct snake segmentation of the corpus callosum. The segmentation utilizes adaptive inflation and subdivision and an appropriate choice of weights. See attached CD for color version.

1.6.4. User Interaction

User interaction is typically employed to assist the snake in latching onto the desired structure, though its implementation often varies. A user can click a point on the image through which they believe the snake must pass, and the snake node closest to the mouse-click position is found and its position reset to the location of the user-supplied point. A force vector field can also be designed in the vicinity of

the user point that attracts the snake to that location. Alternatively, a force resulting from a virtual spring connecting the mouse-click position with the closest snake contour node can be applied. See examples of user-assisted snake segmentation using the first method in Figures 5 and 6.

1.6.5. Gradient Vector Flow Snakes

Traditional snakes require initialization near their targets and consequently are unable to push into concavities. Gradient vector flow (GVF) snakes use a new external force that enables initialization in areas with little gradient information, thus enabling far coarser snake initialization and segmentation of concave objects [13]. The GVF field, denoted by $\mathbf{F}^{gvf}(x,y) = (u(x,y), v(x,y))$, is obtained by minimizing the energy functional E^{gvf} with respect to $\mathbf{F}^{gvf}$:

$$E^{gvf} = \iint (\mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 |\mathbf{F}^{gvf} - \nabla f|^2)\, dx\, dy. \qquad (18)$$

This minimization diffuses the image gradient such that it smoothly declines in lower-valued regions and directly approximates the gradient in high-magnitude areas, reflected by the first and second terms of (18), respectively (Figure 7). The relative weighting of these two terms is governed by the scalar μ. $f(x,y) = -E_{\text{ext}}(x,y)$, where $E_{\text{ext}}(x,y)$ is an image edge map (for example, obtained using a Sobel or Canny filter). u_x, u_y, v_x, and v_y are the derivates of the components of $\mathbf{F}^{gvf}$, $u(x,y)$, and $v(x,y)$, with respect to x and y. $\mathbf{F}^{gvf}$ is then used in place of the external forces, $\mathbf{F}^{\text{external}}$, in Eq. (12), to obtain

$$\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) - \frac{\Delta t}{\gamma}(w_1 \mathbf{F}_i^{\text{tensile}}(t) - w_2 \mathbf{F}_i^{\text{flexural}}(t) + w_3 \mathbf{F}_i^{gvf}(v_i(t))). \qquad (19)$$

1.6.6. Color Images

2D color images map a two-dimensional ($n = 2$) spatial space (x, y) into a three-dimensional ($m = 3$) color space with coordinates (u, v, w) (e.g., R, G, B). In order to apply snakes to segmenting structures in color images, without first converting the color image to a single band, external forces must be redefined to make use of the color gradient. Let the channel derivative matrix $\boldsymbol{D}$ be defined as

$$\boldsymbol{D}(x,y) = \begin{bmatrix} \partial u/\partial x & \partial u/\partial y \\ \partial v/\partial x & \partial v/\partial y \\ \partial w/\partial x & \partial w/\partial y \end{bmatrix}. \qquad (20)$$

Then, the gradient magnitude and the gradient direction are taken as the square root of the largest eigenvalue, $\sqrt{\lambda_{\max}}$, of the matrix $\boldsymbol{D}^T\boldsymbol{D}$ and its corresponding

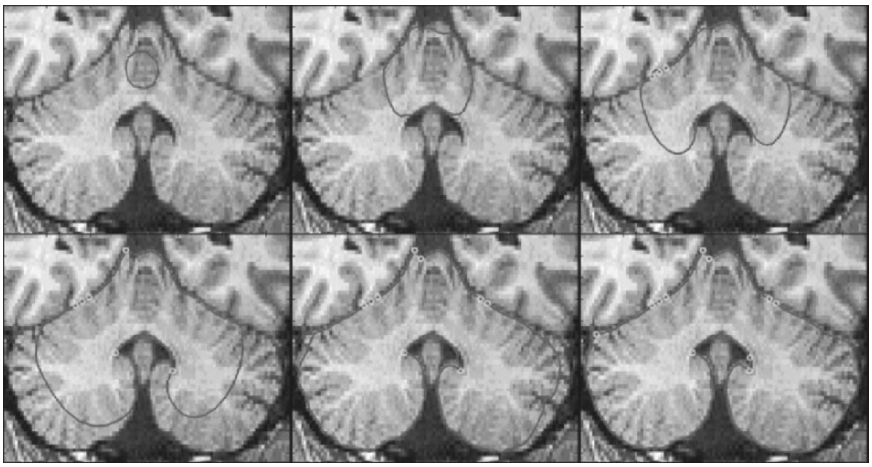
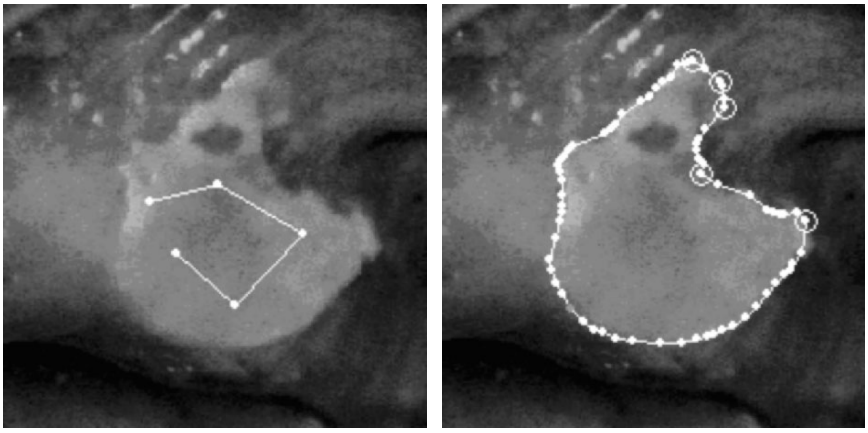


Figure 5. Sample frames showing progress of user-assisted snake segmentation of brain cerebellum. The small green circles are locations of mouse-clicks by the user. See attached CD for color version.



(a) (b)

Figure 6. Segmentation of an oral lesion example using the green band of a digital color image. (a) Initial snake nodes. (b) Final segmentation result (snake nodes shown as white dots and forced points as white circles). Reprinted with permission from [61]. Copyright ©2000, IEEE.

eigenvectors, respectively. In order to apply snakes to segment structures in color images, the external image force becomes $\mathbf{F}_i^{\text{external}}(t) = \nabla P(x_i(t), y_i(t))$,

(a) (b) (c)

Figure 7. Gradient Vector Flow Field: (a) original image depicting a corpus callosum in a midsagittal brain MRI, (b) gradient vector field, (c) GVF field. Note how in (c) the field is less sparse than in (b) and smoothly extends outward into homogenous regions.

where now (compare with (4))

$$P\left(x_i\left(t\right), y_i\left(t\right)\right) = -c\sqrt{\lambda_{\max}(x_i\left(t\right), y_i\left(t\right))}, \tag{21}$$

and $\lambda_{\max}(x_i\left(t\right), y_i\left(t\right))$ is calculated at the location of snake node $x_i\left(t\right), y_i\left(t\right)$ at iteration time t. For more details the reader is referred to [14].

1.6.7. Deformable Spring–Mass Mesh Models

Spring mass meshes can also be used to simulate deformable models that are fitted to object shapes in images. A mesh is made up of nodes (masses or particles) and springs (elastic links or connecting segments). A mass m_i, position $\mathbf{x}_i$, velocity $\mathbf{v}_i$, and acceleration $\mathbf{a}_i$ are associated with each node n_i. Two terminal nodes n_i and n_j, Hooke's spring constant k_s, damping constant k_d, and rest length r_{ij} are associated with each spring s_{ij}. By applying Newton's second law of motion and simulating the dynamics by time integration, the mesh nodes move, deforming the object's shape. Newton's second law of motion for node n_i states that $\mathbf{a}_i = \mathbf{F}_i/m_i$, where $\mathbf{F}_i$ is the total force acting on n_i:

$$\mathbf{F}_i = \mathbf{F}_i^{\text{Hooke}} + \mathbf{F}_i^{\text{viscous}} + \mathbf{F}_i^{\text{user}} + \mathbf{F}_i^{\text{external}}. \tag{22}$$

A spring s_{ij} will cause

$$\begin{aligned}\mathbf{F}_i^{\text{Hooke}} \;=\; & -k_s\left(\|\mathbf{x}_i - \mathbf{x}_j\| - r_{ij}\right)\frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|} - \left(k_d\left(\mathbf{v}_i - \mathbf{v}_j\right)^T \frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|}\right) \\ & \frac{uij}{\|uij\|} \text{ where } uij = xi - xj \end{aligned} \tag{23}$$

to be exerted at n_i and $-\mathbf{F}_i^{\text{Hooke}}$ on n_j. Viscous drag at n_i is given by

$$\mathbf{F}_i^{\text{viscous}} = -k_v\mathbf{v}_i. \tag{24}$$

A single user-applied force $\mathbf{F}_i^{\text{user}}$ is implemented as the dynamic force resulting from a spring connecting a mesh node to the (varying) position of the user's point of application. Image forces can be implemented as

$$\mathbf{F}_i^{\text{external}}(t) \propto \nabla \left\| \nabla \left[G_\sigma * I(\mathbf{x}_i) \right] \right\|, \tag{25}$$

where $I(\mathbf{x}_i)$ is the intensity of a pixel at the location of node n_i (see Eq. (4)). Image forces that attract the model to an image boundary are calculated only for boundary mesh nodes (similarly, image forces that attract medial model nodes to medial features can also be applied). Following the calculation of the node forces we compute the new acceleration, velocity, and position of each node given the old velocity and position values, as follows (explicit Euler solution with time step Δt):

$$\begin{aligned} \mathbf{a}_i &= \mathbf{F}_i / m_i, \\ \mathbf{v}_i &= \mathbf{v}_i^{old} + \mathbf{a}_i \Delta t. \\ \mathbf{x}_i &= \mathbf{x}_i^{old} + \mathbf{v}_i \Delta t. \end{aligned} \tag{26}$$

1.6.8. General Shape Parameters

A vector of shape parameters $\mathbf{u}$ can also be used to represent the continuous geometric contour model $\mathbf{v}(s)$. These shape parameters are generally associated with some local-support basis functions (such as splines and finite elements) or global-support basis functions (such as Fourier bases) [7]. In this case the discrete form of $\xi(\mathbf{v})$ may be written as

$$E(\mathbf{u}) = \frac{1}{2}\mathbf{u}^T \mathrm{K}\mathbf{u} + \mathrm{P}(\mathbf{u}), \tag{27}$$

where K is called the stiffness matrix and $\mathrm{P}(\mathbf{u})$ is the discrete version of external potential $P(\mathbf{v}(s))$. The contour parameters that minimize the energy function can now be obtained by solving the set of algebraic equations

$$\mathrm{K}\mathbf{u} = -\nabla \mathrm{P}. \tag{28}$$

The motion equation for the contour (represented by $\mathbf{u}$) can be written as

$$\mathrm{M}\ddot{\mathbf{u}} + \mathrm{N}\dot{\mathbf{u}} + \mathrm{K}\mathbf{u} = -\nabla \mathrm{P}, \tag{29}$$

where M and N are mass and damping matrices, respectively [7].

1.6.9. A Probabilistic Formulation

An alternative view of deformable models is a probabilistic formulation. The deformable model can be fitted to the image data by finding the model shape

parameters $\mathbf{u}$ that maximize the posterior probability (maximum a posteriori or MAP solution) expressed using Bayes' theorem as

$$p(\mathbf{u}|I) = \frac{p(I|\mathbf{u})p(\mathbf{u})}{p(I)}, \tag{30}$$

where $p(\mathbf{u})$ is the prior probability density of the model shape parameters: a mechanism for probabilistic regularization. $p(I|\mathbf{u})$ is the probability of producing an image I given the parameters $\mathbf{u}$: an imaging (sensor) model. $p(\mathbf{u})$ and $p(I|\mathbf{u})$ can be written (in the form of Gibbs distribution) as

$$p(\mathbf{u}) = k_1 \exp(-\mathrm{A}(\mathbf{u})), \tag{31}$$

$$p(I|\mathbf{u}) = k_2 \exp(-\mathrm{P}(\mathbf{u})), \tag{32}$$

where k_1 and k_2 are normalizing constants and $\mathrm{A}(\mathbf{u})$ is the discrete version of the internal energy $\alpha(\mathbf{v})$ and $\mathrm{P}(\mathbf{u})$ is the discrete version of external potential $P(\mathbf{v}(s))$.

1.6.10. Deformable Surface Models

For segmenting 3D objects in volumetric medical images, one could resort to slice-by-slice segmentation, where in each slice a 2D deformable model is initialized using the results from a neighboring slice. However, this approach does not inherently integrate 3D information. The following steps are generally performed in 3D deformable models. A geometrical 3D shape representation of the deformable model is decided upon; the model is initialized (including initial parameter setting) in the image space, and the model is left to deform to minimize internal and external energy terms by using internal and external forces, for example. Some example implementations of 3D deformable models include Simplex Meshes [15] and finite-element methods [16]. Another implementation is the 3D spring–mass mesh model, which is similar to the 2D spring mass mesh system but now masses are positioned in 3D space so the position, velocity, and acceleration (x, v, and a) become 3-vectors and new positions are calculated as in Eq. (26), where only the dimensionality of the position, force, velocity, and acceleration vectors has changed.

1.7. Optical Flow Snake Forces

In this section we present a detailed look into an extension of the formulation of Active Contour Models (snakes) that includes an additional contour-deforming force [17, 18]. The new force is derived from the optical flow field calculated between two time-consecutive frames in an image sequence. Addition of the new force assists the snake in tracking desired dynamics while the traditional snake forces guarantee the contour's smoothness and attraction to edges. The method is

applied to the problem of tracking the leading edge of an injected contrast agent in an echocardiographic image sequence and is shown to improve the tracking performance. A clinical motivation and previous work on echocardiography and video densitometry are initially presented.

1.7.1. Clinical Motivation

The assessment of human right-ventricular (RV) function is of great importance in the study of many diseases afflicting the heart. For example, disturbances in filling and elimination patterns of RV hemodynamics can be interpreted as signs of abnormal RV function. Arrhythmogenic Right-Ventricular Dysplasia (ARVD) is a rare but clinically important disease that afflicts young adults and may play a role in the etiology of sudden death among the young [19]. Impairments in RV function in this group of patients can be described in terms of wall motion abnormalities, or as localized bulging and sacculations. These abnormalities are mainly located at the inflow, outflow, or apical regions. To study these abnormalities we use sequences of contrast echocardiographic images. Automated medical image analysis techniques are needed to extract clinically relevant information from this of data.

1.7.2. Medical Imaging Procedure

Thirty patients with biopsy-verified ARVD and 18 healthy volunteers (control group) were investigated by use of contrast echocardiography. The investigations were performed with an Acuson XP128 computer system or a Sequoia system equipped with multi-Hertz transducers. As a contrast agent, 2 ml of Haemaccel® (Hoechst) was injected intravenously. Transthoracic apical four-chamber view with focus on the right ventricle was used and continuously recorded during and after injection. The video sequence (Figure 8) of the filling and elimination of the contrast agent was then digitized using a PC with a frame grabber (Matrox, Meteor II), giving about 600 images for each sequence.

1.7.3. Tracking the Contrast Front

In order to characterize the RV flow patterns, we track the front (the leading edge) of the contrast agent during RV filling. Examining the flow of the contrast agent in a typical echocardiographic image sequence (Figure 9) reveals weak ultrasound echoes, echo dropouts, and high levels of noise. Application of simple edge detectors to locate the front of the contrast would therefore result in detecting erroneous edges and gaps.

The strength of active contour models in integrating low-level image information and guaranteeing a smoothly connected contour makes them suitable for this purpose. For tracking the contrast front in a sequence we make use of the motion field obtained by Optical Flow (OF) computation as an additional contour

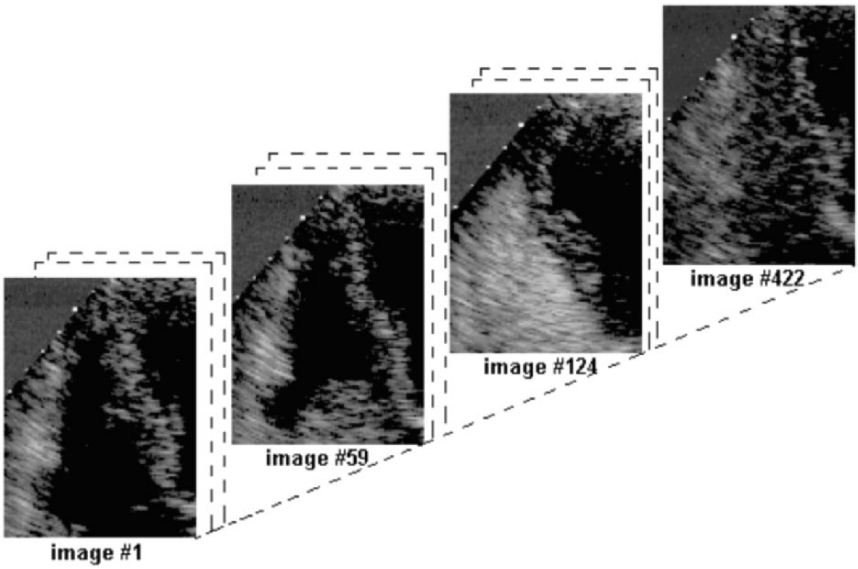


Figure 8. Sample frames from a digitized image sequence. In frame #1, #59, #124, and #422 the contrast agent has not, reached, just reached, totally filled, and washed out from the RV, respectively. Reprinted with permission from [18]. Copyright ©2000, IEEE.

deforming force, in order to speed up tracking and influence the snake nodes to match corresponding contrast front regions between frames.

Other authors have investigated similar approaches. In [20] a method for segmenting and tracking cardiac structures in ultrasound image sequences was presented. In integrating the contour's equation of motion, the method sets the initial velocities of the contour vertices to OF estimates, and sets their positions relative to the final position from the preceding frame. Peterfreund [21] used Kalman filter-based active contours that calculate OF along the contour as system

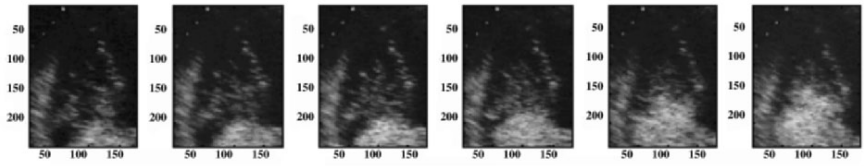


Figure 9. Successive frames of contrast agent entering the right ventricle of the heart. Reprinted with permission from [17]. Copyright ©2000, IEEE.

measurement to detect and reject measurement that may belong to other objects. Akgul et al. presented an application of tracking 2D contours of tongue surfaces from digital ultrasound image sequences [22]. The method makes use of OF to reduce the computational complexity involved when searching for optimal snake node locations in a dynamic programming setting. This is done by considering only a subset of pixels in a search window. The subset is chosen on the basis of the first OF constraint, namely that the intensity of an object's point in a dynamic image does not change with time.

1.7.4. Optical Flow

Optical flow [23] is a well-established method for calculating the velocity field $(u(x,y), v(x,y))$ of the apparent 2D motion of pixels in a dynamic image, $I(x,y,t)$, due to the 3D motion of imaged objects. The OF velocity field is obtained by examining the spatial and temporal changes in intensity values. Classical OF is based on two main constraints. The first states that the brightness of any object point is constant over time. This can be written as

$$I(x+dx, y+dy, t+dt) = I(x,y,t). \tag{33}$$

Using Taylor series expansion and neglecting higher-order terms gives the first OF constraint equation:

$$I_x u + I_y v + I_t = 0, \tag{34}$$

where $u = dx/dt$, and $v = dx/dt$ are the desired velocity field components, I_x and I_y are the spatial image derivatives, and I_t is the temporal image derivative. Equation (33) by itself is insufficient to calculate (u, v); hence a second constraint, the velocity field smoothness constraint, is introduced. The velocity field can now be calculated as that which best satisfies both constraints by minimizing the following square error function:

$$\xi^2(x,y) = (I_x u + I_y v + I_t)^2 + \lambda(u_x^2 + u_y^2 + v_x^2 + v_y^2), \tag{35}$$

where λ is a Lagrange multiplier. The following iterative algorithm detailed in [23] is used to find the velocity field (Figure 10):

$$\begin{aligned} &\text{Initialize: } u(x,y) = v(x,y) = 0 \text{ for all } x, y, \\ &\text{do: } u = \bar{u} - I_x \tfrac{P}{D},\ v = \bar{v} - I_y \tfrac{P}{D} \text{ while } \xi^2(x,y) > \varepsilon, \end{aligned} \tag{36}$$

where $P = I_x\bar{u} + I_y\bar{v} + I_t$, $D = \lambda^2 + I_x^2 + I_y^2$, and ε is a small number.

1.7.5. Optical Flow Snake Forces

In order to track the contrast agent front in an echocardiographic image sequence, we need to accomplish two tasks. The first is to locate the region where

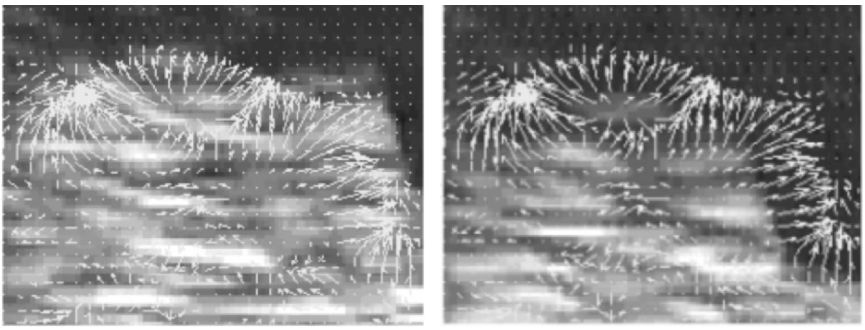


Figure 10. Optical flow (velocity) field shown on two consecutive frames.

the contrast front has moved from one frame to the next, and the second is to detect this front as a smooth and connected boundary. We use the optical flow to address the first task and snakes to address the second. To combine the two techniques we include an additional force term $\mathbf{F}_i^{\text{flow}}(t)$ proportional to the calculated velocity field at the current snake node position $\mathbf{v}_i(t) = (x_i(t), y_i(t))$, yielding

$$\mathbf{v}_i(t) = \mathbf{v}_i(t-1) + w_1\mathbf{F}_i^{\text{tensile}}(t) + w_2\mathbf{F}_i^{\text{flexural}}(t) + w_3\mathbf{F}_i^{\text{external}}(t) + w_4\mathbf{F}_i^{\text{flow}}(t), \tag{37}$$

where w_i are weighting factors,

$$\mathrm{F}_{\mathrm{i}}^{\text{flow}}(t) \propto (u(x_i(t-1), y_i(t-1)), v(x_i(t-1), y_i(t-1))), \tag{38}$$

and $u\,(x_i, y_i)$, $v\,(x_i, y_i)$ are obtained using algorithm (36) in Section 1.7.4.

1.8. Results

We tracked the leading edge of a contrast agent filling the RV in real ultrasonic image sequences. Images were first smoothed using nonlinear diffusion filtering [24]. The front of the contrast agent was tracked in eight sequences during the RV filling process, five from the ARVD group and three from the control group. In each sequence, the front was tracked, on average, in about eight images. In the example depicted in Figure 11, the snake without OF forces needed 19, 20, 23, 22, 22, and 16 iterations (Figure 11a), whereas the snake with OF forces needed 5, 8, 10, 10, 6 and 10 iterations (Figure 11b). We also show an example of both snakes, with and without OF forces, deforming toward the leading edge of the contrast agent in a single frame (Figure 12). The OF snake (with larger nodes in the figure) progresses faster toward the edge and locates it in only 10 iterations compared to 23 iterations needed for the snake without OF forces. Histograms of the number of iterations needed for the contour to find the edge for all tested frames are also calculated (Figure 13). The mean number of iterations needed with

and without the use of information about the OF was 6.3 and 12.3, respectively. Figure 14 shows the snake contour tracking the leading edge of the contrast front and providing clinically relevant RV hemodynamics measurements.

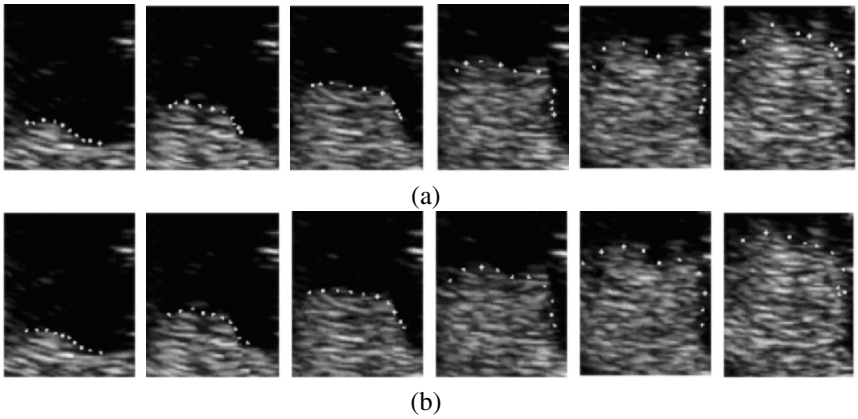


Figure 11. Results of tracking a real sequence. Upper frames: without using optical flow forces obtained after 19, 20, 23, 22, 22, and 16 iterations. Lower frames: with optical flow forces obtained after 5, 8, 10, 10, 6, and 10 iterations. Reprinted with permission from [17]. Copyright ©2000, IEEE.

2. SMART SNAKES: INCORPORATING KNOWLEDGE ABOUT SHAPE

2.1. Introduction

Active shape models (ASMs) or smart snakes are deformable shape modeling techniques for segmentation of objects in images. ASMs ensure that the deformed shape is consistent with a statistical model of shape variability calculated before-

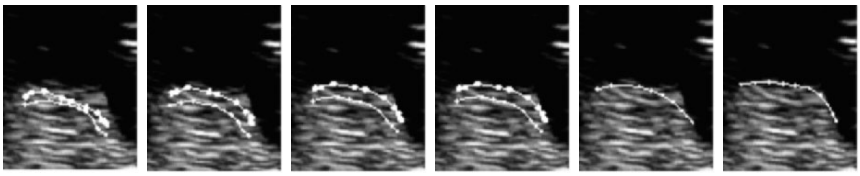
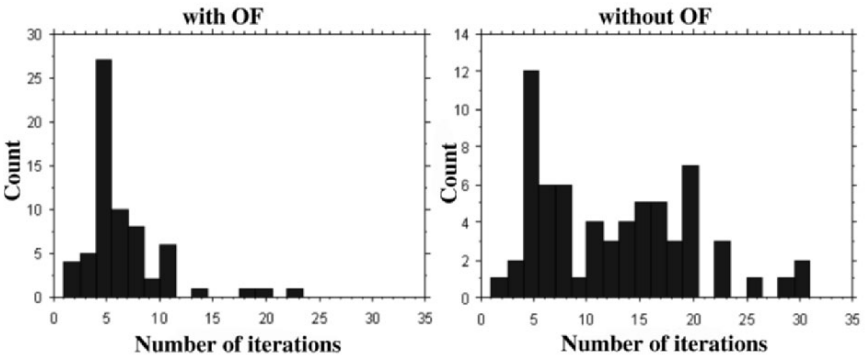


Figure 12. The snake with optical flow forces (large nodes) progresses faster toward the contrast front compared to the snake without optical flow forces. The snake nodes are shown after 1 (left-most), 2, 6, 10, 15, and 23 (right-most) iterations. Reprinted with permission from [17]. Copyright ©2000, IEEE.



(a) (b)

Figure 13. Histogram of the total number of iterations needed for the contour to latch onto the contrast front (a) with optical flow forces and (b) without optical flow forces. Reprinted with permission from [18]. Copyright ©2000, IEEE.

hand in a training stage. In an ASM, an initial shape model estimate is deformed to fit the image data. The deformations are chosen such that they minimize a certain cost function and are consistent with the a priori knowledge about the shape of the target object to be segmented. The cost function is inversely proportional to the fit of the model to certain image features. Although incorporating a priori shape knowledge into an ASM generally improves the segmentation results, the technique requires training, becomes less general, and is not suitable on its own for detecting shapes that are dissimilar to those in the training set, which is problematic in cases where unpredictable changes in shape due to pathology may be observed in new cases. In the following sections we describe the training and application stages of an ASM. Further details are provided in Section 4 that describe the extension of 2D ASM to include temporal variations.

2.1.1. Modeling the a priori Knowledge of Shape Variation

A representative training set of images depicting the shape variations we wish to model is collected. A training set of shapes described by corresponding landmarks (distinctive coordinates) is extracted from the training set of images. Given N training example shapes, each represented by L landmarks, each landmark being an (x, y) coordinate, the next step is to obtain a set of aligned shapes, viewed as a collection of N points in a $2L$-dimensional space. The alignment is done by changing the pose (scaling, rotation, and translation) of the shapes such that the sum of distances between corresponding landmarks is minimized. Principal Component Analysis (PCA) is then applied in order to capture the main modes of shape variations of the aligned training set of shapes. Using the result from

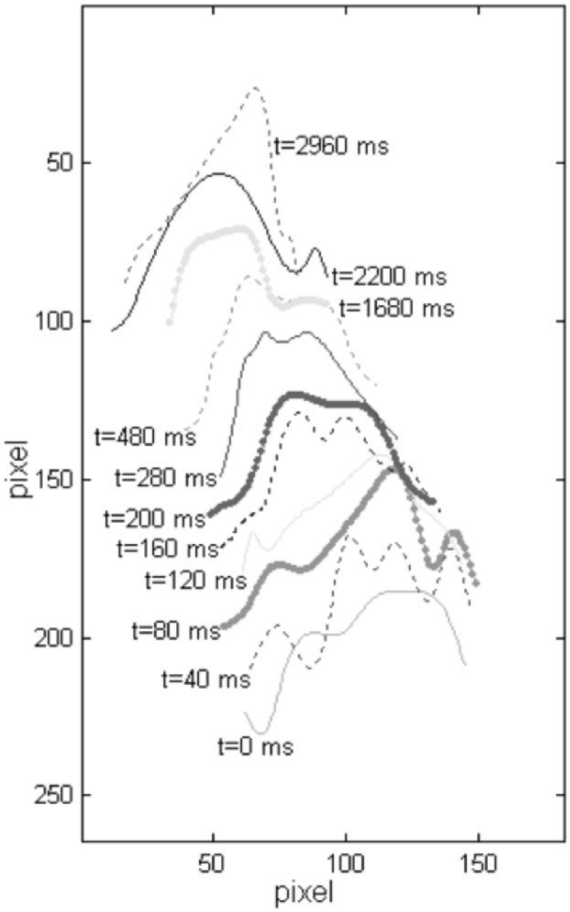


Figure 14. The result of tracking the leading edge in one of the sequences from the ARVD group. Each contour represents the contrast agent front at different times, indicated beside each contour. The contrast front enters the RV ($t = 0$) until the RV is totally filled ($t = 2960$ ms). Note how the contrast front in the initial phase of filling ($t = 0$ to 480 ms) moves faster than the final phase (t =480 to 2960 ms). This is indicative of the inhomogeneous operation of the RV, identified via contrast front tracking. Reprinted with permission from [18].

PCA, we can approximate each shape by the sum of a mean shape and a linear combination of, say t, principal components, as follows:

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}, \tag{39}$$

where

$\mathbf{x}$ is the vector of landmark coordinates,

$\bar{\mathbf{x}}$ is the mean shape,

$\mathbf{P}$ is the matrix of principal components,

$\mathbf{b}$ is a vector of weighting parameters, also called shape parameters, and

$\mathbf{x}$ and $\bar{\mathbf{x}}$ are each of length $2L$. $\mathbf{P}$ is $2L \times t$ and $\mathbf{b}$ is a vector of length t.

Equation (39) can be used to generate new shapes by choosing different values for $\mathbf{b}$. Constraints on these weighting parameters are used to ensure that only allowable shapes are produced. The allowable shapes are those shapes belonging to a region called the Allowable Shape Domain (ASD), which is inferred from the region that the aligned training set occupies. Aside from generating new similar shapes, this model is used to examine the plausibility of other shapes by checking whether or not they lie in the ASD. The above model of the distribution of points (or variation of shapes) is referred to as a Point Distribution Model (PDM).

2.1.2. Modeling the Gray-Level Appearance

During image search or the segmentation procedure, there is a need to find the desired movement of the landmarks of a current shape estimate to new and better locations. The gray-level information (image data) is essential to finding such suggested movements. This implies that we need to model the gray level information and make such a model available during image search. This may be done by examining the intensity profiles at each landmark and normal to the boundary created by the landmark and its neighbors. The intensity profiles are then used to derive a normalized intensity difference (gradient, or derivative) profile giving invariance to the offsets and uniform scaling of the gray levels [25]. With L landmarks for each shape and N shapes in the training set along with N training images, we derive N profiles for each landmark, one from each image, and calculate the mean profile for each landmark using

$$\bar{\mathbf{y}}_j = \frac{1}{N}\sum_{i=1}^{N} \mathbf{y}_{ij}, \tag{40}$$

where $\mathbf{y}_{ij}$ is the normalized derivative profile for the jth landmark in the ith image, and $\bar{\mathbf{y}}_j$ is the mean normalized derivative profile for the jth landmark.

2.1.3. Model Deformation and Attraction to Image Features

To locate an instance of an object in a new image, we start with an initial shape estimate and then examine the neighborhood of the landmarks of our estimate aiming at finding better locations for the landmarks. We then change the shape and pose of our estimate to better fit the new locations of the landmarks while producing in the process a new acceptable or allowable shape.

An instance of an object is described as the sum of the mean shape obtained from the training and a weighted sum of the principal components, with the possibility of this shape being translated, rotated, or scaled. The weights of the principal components are called the shape parameters and the values for the rotation angle and the scaling factor, and the translation values in the x and y directions are called the pose parameters.

Starting with an initial shape, we extract a perpendicular intensity profile for each landmark and use it to derive a normalized derivative search profile in the same way as described in Section 2.1.2. Within this search profile, we seek a subprofile that matches the mean or expected profile obtained in the training. This can be done by defining the following square-error function, which decreases as the match improves, and minimizing it with respect to the location of this subprofile, d:

$$f(d) = (\mathbf{h}(d) - \bar{\mathbf{y}}_j)^T(\mathbf{h}(d) - \bar{\mathbf{y}}_j), \tag{41}$$

where $\bar{\mathbf{y}}_j$ is the mean normalized derivative profile for the jth landmark, and $\mathbf{h}(d)$ is a subprofile along the search profile having a length equal to that of $\bar{\mathbf{y}}_j$ and centered around the point on the search profile that is offset by d from the landmark.

Finding d that minimizes the above function for all the landmarks, we arrive at L new locations for the landmark positions. We may choose to control the suggested movement by, for example, changing a small suggested movement to no movement or large suggested movements to only half what is suggested.

The next step is to update the pose and shape parameters in order to move as closely as possible to the new proposed positions with the restriction that the new produced shape is an allowable shape. A practical, though not optimal, solution is to first find only the pose parameters that move the shape estimate as close as possible to the new proposed positions. Then there will remain residual adjustments that can only be satisfied by deforming the shape and hence changing the shape parameters. Finding the pose parameters is done by aligning the current estimate to the new proposed shape. Alignment details are presented in the section for the spatio-temporal case. The remaining landmark position modifications are in general $2L$-dimensional, whereas the shape variations obtained from the model are only t-dimensional. A least-squares solution can be used to solve the following equation for the changes in shape parameters $d\mathbf{b}$ (with an orthonormal matrix

column of $\mathbf{P}$ we have $\mathbf{P}^T\mathbf{P} = \mathbf{I}$)

$$d\mathbf{x} = \mathbf{P}d\mathbf{b} \Rightarrow d\mathbf{b} = (\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T d\mathbf{x} = \mathbf{P}^T d\mathbf{x}, \tag{42}$$

where $d\mathbf{x}$ is a vector containing the remaining landmark position modifications, $d\mathbf{b}$ is a vector of changes in the shape parameters, and $\mathbf{P}$ is the matrix of principal components.

Finally, we limit the shape variations to obtain an acceptable or allowable shape within the ASD by applying the constraints on the shape parameters. With this step we complete a single iteration. In a similar fashion, we obtain new estimates and re-iterate until we assume convergence, defined when the parameter changes are insignificant.

2.1.4. Improvements and Extensions to ASMs

In this section we present several enhancements and additions to the basic ASM method. We start by presenting an automatic landmark generation algorithm followed by a multi-resolution implementation of the ASM. Active Appearance Models (AAMs) are then discussed. The use of genetic algorithms and the application of Kalman filtering for tracking using ASMs are also presented here. We then show how vibrational modes can be added to the original variation modes of PDMs. A method for estimating the shape distribution is briefly discussed, and finally some comments on classification using ASMs are presented.

2.2. Automatic Landmark Generation for PDMs

In order to obtain the (PDM) when the shapes are represented by landmarks, it is required that all the landmarks in all the training set of images are individually specified. This is typically done by hand, clicking a mouse on the locations of the landmarks on an image. This is obviously time-consuming and affected by inter- and intra-operator variability. It is also particularly difficult to place the landmarks appropriately when building a 3D model from volume images. A method for automatically generating PDMs is proposed in [26]. The suggested method comprises two main stages. First, a pairwise corresponder is used to establish an approximate set of landmarks on each of the example boundaries. The pairwise corresponder identifies corresponding points on different examples of the object, by using the curvature of the boundary of the objects as a basis for a dynamic programming matching algorithm. The dynamic programming algorithm attempts to match points of high curvature on the two boundaries so as to minimize the discrepancy in the curvature over the length of the boundary. Then, the landmarks are refined using an iterative nonlinear optimization scheme to generate a more compact PDM. The refinement is done iteratively by varying the pose and the shape of the landmarks associated with a given member of the training set in order to minimize a cost function. One term of the cost function decreases as the

shape becomes more similar to the mean shape, thus compacting the PDM model, while another term increases when the landmarks move away from a given boundary. More recently this line of work has been formalized through the concept of Minimum Description Length.

Multi-Resolution Image Search An important issue that affects the image search considerably in the ASM is choice of the length of the search profile. In choosing the length of the search profile we are faced with two contradicting requirements: on one hand, the search profile should be long enough to contain within it the target point (the point that we need the landmark point to move to); on the other, we require the search profile to be short for two reasons: first, to reduce the computations required, and second, if the search profile is long and the target point is close to the current position of the landmark, then it will be more probable to move to a farther away noisy point and miss the target. In [27] a multi-resolution approach is suggested where at first the search looks for faraway points and makes large jumps, and as the search homes in on a target structure, it should restrict the search only to close points.

In order to achieve such multi-resolution search, we generate a pyramid of images. This pyramid of images contains the same image but with different resolutions. At the base of the pyramid, Level 0, we have the original image. On the next level we have a lower-resolution version of the image with half the number of pixels along each dimension, and so on, until we reach the highest level of the pyramid. Low pass filtering (smoothing), for example using Gaussian filtering, should always precede image sub-sampling.

The search will begin at the top level of the pyramid (i.e., image with the lowest resolution). The search is then initiated one level below using the search output of the previous level, and so on, until the lowest level of the pyramid (the original image) is reached. In order to be able to perform such a search in each level, we should be equipped with information about the gray-level profiles in each of these levels. This demands that during the training stage we obtain the mean normalized derivative profile for each landmark at all levels of the pyramid.

The criterion utilized in order to change the level of search within the pyramid is as follows. Move to a lower level when a certain percentage of landmarks do not change considerably, for example, when 95% of the landmarks move only within the central 50% of the search profile. A maximum number of iterations can also be devised to prevent getting stuck at a higher level.

Active Appearance Models (AAMs) ASMs are statistical models of the shape variations of an object that are used along with additional information about the gray level values to segment new objects. AAMs are an extension of ASMs that combine information about the shape and intensity of an object into one model describing the appearance [28]. The model parameters are changed to locate new

instances of the object in new images. The changes are found by measuring a residual error between the appearance of the model and the data. The relationship between the changes in the model parameters and the residual error is found in the training stage. More details follow.

A shape model and a gray-level model are produced in similar fashion to ASMs then combined to obtain an appearance model. The shape model is given by

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}_s \mathbf{b}_s. \tag{43}$$

The gray level model is obtained by first warping the examples to the mean shape; then the gray-level values are sampled from the shape-normalized image. In order to reduce the effects of global lighting variation, the sampled values are normalized. The gray level model is written as

$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{P}_g \mathbf{b}_g. \tag{44}$$

The complete appearance parameters are the shape and the gray level parameters, $\mathbf{b}_s$ and $\mathbf{b}_g$, combined with the four pose which perform scaling by s, rotation by θ, and translation by t_x and t_y.

In the AAM search process, the task is to minimize the difference between a new image and one synthesized by the appearance model. The relationship between the difference and the model parameters is approximated linearly. The coefficients of this linear transformation are obtained in a training stage by varying the model parameters and observing the change in the image intensities.

Active Blobs (ABs) [29] constitute an approach similar to AAMs. This method uses the image difference to drive tracking by learning the relationship between image error and parameter offset in an off-line processing stage. The main difference is that ABs are derived from a single example with low-energy mesh deformations (derived using a Finite-Element Method), whereas AAMs use a training set of examples.

Genetic Algorithms Genetic algorithms employ mechanisms analogous to those involved in natural selection to conduct a search through a given parameter space for an extreme value of some objective function. This concept can be employed in ASMs, where the parameter space consists of possible ranges of values for the pose and shape parameters of the model. The objective function to be maximized reflects the similarity between the gray levels related to the object in the search stage and those found from training. In [30] the use of GA for model-based image interpretation is presented and can be used to fit the formulation of ASMs. The use of GA for finding the model parameters that fit a model to image data is presented below.

A point in the search space is encoded as a chromosome and a population of chromosomes is maintained. Combining existing points generates new points

in the search space. Solutions are obtained by iteratively generating new generations of chromosomes by selective breeding based on the relative values of the objective function for different members of the population. The set of shape and pose parameters is encoded as a string of genes to form a chromosome. Each gene can take one of several values called alleles. It has been shown that long chromosomes with few alleles are preferable to shorter chromosomes with many alleles. Consequently, chromosomes usually contain parameters encoded as bit strings. Mutation and crossover are applied to chromosomes to produce children. Crossover is when the two parent chromosomes are cut at a certain position and the opposing sections combined to produce two children. Mutation is done by selecting a gene and changing its value. In the case of binary representation, the bit is complemented. In the iterative search process, the fitter an individual, i.e., the higher the value of the objective function obtained from the parameters of the chromosome of the individual, the more this individual will be used to produce children. Moreover, certain regions in the search space will be allocated more trials due to the existence of some evidence that these regions are fitter.

Tracking Using ASMs and Kalman Filtering A simple implementation of tracking was mentioned in [31], where the investigators simply set the initial approximation of model parameters in a certain frame to the solution from the previous frame in an image sequence. Here we describe the use of Kalman filtering for tracking using an ASM as described by [32].

The state space vector is composed of the shape parameters, the origin of the shape, the velocity of the origin, and alignment terms. The shape parameters are modeled as a simple discrete stochastic process, where the current shape parameter is equal to the previous parameter plus an additive random noise term. The shape parameters are assumed to vary independently, and the variance of the noise is set to be proportional to the variance of the shape parameter (the corresponding eigenvalue obtained in the PCA). The origin is assumed to undergo a uniform 2D motion with a randomly varying force represented by an additive random noise. The alignment parameters (two parameters incorporating rotation and scaling) are assumed constant with additive random noise.

The observations (observed features) are obtained at each new frame and represented by the points of maximum contrast. Measurements are made by searching along the normal to the estimated contour at the boundary points. The measurement model is formulated according to the fact that the curve points are related to the state space parameters as a rotated, scaled, and translated version of the mean shape plus the principal components weighted by the shape parameters. This measurement model is nonlinear; hence the extended Kalman filter can be used. Another approach that simplifies the problem is suggested. The change in origin is first estimated with the assumptions that the shape and alignment parameters are fixed. Then the effect of the origin shift is removed and the alignment parameters are estimated. Finally, the effect of change in alignment is removed

and the shape parameters updated. Following the above assumptions, the standard discrete Kalman filter is used to update the state estimates and covariance.

Augmenting the Statistical Variations with Vibrational Modes In [33] a method is presented for combining two approaches to modeling flexible objects. Modal analysis using Finite-Element Methods (FEMs) generates a set of vibrational modes for a single shape. Point Distribution Models (PDMs) generate a statistical model of shape variation from a set of training example shapes. The proposed method combines the two and generates vibrational modes when few example shapes are available and changes gradually to using more statistical modes of variation when a large set is presented.

Finite-Element Methods take a single shape and treat it as if it were made of a flexible material. The techniques of Modal Analysis give a set of linear deformations of the shape equivalent to the modes of vibration of the original shape. Although a flexible model of a shape with these vibrational modes can be used in image search, the resulting shapes may not represent the real variations that occur in a class of shapes.

Similar to previous discussions, a shape here is represented by a set of landmarks (coordinate points); the landmarks are considered nodes with certain masses that are mutually interconnected by springs with constant stiffness and rest lengths equal to the distance between the points.

In an FEM, new shapes can be generated using

$$\mathbf{x} = \hat{\mathbf{x}} + \mathbf{\Phi}\mathbf{u}, \tag{45}$$

where $\hat{\mathbf{x}}$ is a vector of the original points of the shape include this here, $\mathbf{\Phi}$ is a matrix of eigenvectors representing the vibrational modes, and $\mathbf{u}$ is a set of parameters.

Matrix $\mathbf{\Phi}$ can be found by solving the following generalized eigensystem:

$$\mathbf{K}\mathbf{\Phi} = \mathbf{M}\mathbf{\Phi}\Omega^2, \tag{46}$$

where $\mathbf{K}$ is the stiffness matrix and $\mathbf{M}$ is the mass matrix.

Matrix $\Omega^2 = diag\begin{pmatrix} \omega_1^2 & \omega_2^2 & \dots & \omega_{2L}^2 \end{pmatrix}$ is a diagonal matrix of eigenvalues associated with the eigenvectors of $\mathbf{\Phi}$, and ω_i^2 is the frequency of the ith vibrational mode.

For a shape of L landmarks $\mathbf{\Phi}$, $\mathbf{M}$, and $\mathbf{K}$ are $2L \times 2L$ matrices. Vectors $\hat{\mathbf{x}}$ and $\mathbf{x}$ are both of length $2L$. Matrix $\mathbf{M}$ is set to the identity matrix, and $\mathbf{K}$ can be calculated as

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}_{xx} & \mathbf{K}_{yx} \\ \mathbf{K}_{xy} & \mathbf{K}_{yy} \end{pmatrix}, \tag{47}$$

where $\mathbf{K}_{xx}$, $\mathbf{K}_{yx} = \mathbf{K}_{xy}^T$, and $\mathbf{K}_{yy}$ are $L \times L$ with off-diagonal elements given by

$$\left.\begin{array}{rcl} k_{xxij} & = & d_{xij}^2/d_{ij}^2 \\ k_{yyij} & = & d_{yij}^2/d_{ij}^2 \\ k_{xyij} & = & d_{xij}d_{yij}/d_{ij}^2 \end{array}\right\}, \tag{48}$$

and diagonal elements $k_{yyii} = k_{xxii} = 1.0$ and $k_{xyii} = 0$, where

$$\left.\begin{array}{rcl} d_{xij} & = & x_i - x_j \\ d_{yij} & = & y_i - y_j \\ d_{ij}^2 & = & d_{xij}^2 + d_{xij}^2 \end{array}\right\} \qquad (x, y \text{ are from the vector } x). \tag{49}$$

One suggestion for combining the FEM and PDM is by using an equation of the form

$$\mathbf{x} = \hat{\mathbf{x}} + \mathbf{\Phi u} + \mathbf{Pb}, \tag{50}$$

where $\hat{\mathbf{x}}$ is a mean shape, $\mathbf{P}$ is a matrix representing the statistical variations, and $\mathbf{b}$ is a vector of parameters. Since the statistical modes of variation and the vibrational modes are not independent, this approach is unsatisfactory due to the redundancy in the model.

An alternative approach for combining the models is as follows. If only one shape example exists, then we are left with only the FEM model. Given two shape examples, we calculate the vibrational modes for each shape, and use them to generate more shapes by randomly selecting the parameters $\mathbf{u}$. We then train a PDM on this new set of examples. If more than two shapes exist, then we need to reduce the effect of the vibrational modes and increase the effect of the statistical modes of variation because, as mentioned earlier, the vibrational modes may not represent real shape variations of a certain class of objects.

This idea can be realized by first choosing the distribution of $\mathbf{u}$ to have a zero mean and a diagonal covariance matrix $\mathbf{S}_u = \alpha\mathbf{\Lambda}$, with $(\mathbf{S}_u)_{ii} = \alpha\omega_i^{-2}$ (this gives more variations in the low-frequency vibrational modes, and less variation in the high-frequency vibrational modes), and then calculating the eigenvectors and eigenvalues of the "combined-model" covariance matrix of the generated example shapes. Now, the derivation of this covariance matrix is presented.

Starting with m original example shapes: $\mathbf{x}_i$ for $1 \leq i \leq m$, with a mean

$$\bar{\mathbf{x}} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i. \tag{51}$$

The PDM is obtained by finding the eigenvectors of the covariance matrix:

$$\mathbf{S} = \frac{1}{m-1}\sum_{i=1}^{m}(\mathbf{x}_i - \bar{\mathbf{x}})^T(\mathbf{x}_i - \bar{\mathbf{x}}). \tag{52}$$

One example shape $\mathbf{x}_i$ can be used to produce other shapes from its vibrational modes as

$$\mathbf{x}_{gi} = \mathbf{x}_i + \mathbf{\Phi}_i \mathbf{u}, \tag{53}$$

where $\mathbf{x}_{gi}$ is a shape generated from the vibrational modes of $\mathbf{x}_i$.

The covariance matrix of $\mathbf{x}_{gi}$ is given by

$$\mathbf{S}_{gi} = \mathbf{S} + \mathrm{cov}(\mathbf{\Phi}_i \mathbf{u}) = \mathbf{S} + \mathbf{\Phi}_i \mathbf{S}_u \mathbf{\Phi}_i^T.$$

Using ($\mathbf{S}_u = \alpha \mathbf{\Lambda}$), we obtain

$$\mathbf{S}_{gi} = \mathbf{S} + \alpha \mathbf{\Phi}_i \mathbf{\Lambda} \mathbf{\Phi}_i^T. \tag{54}$$

The "combined-model" covariance, $\mathbf{S}_c$, of the shapes ($\mathbf{x}_{gi}$, $1 \leq i \leq m$) generated from the original shapes ($\mathbf{x}_i$, $1 \leq i \leq m$) can be estimated by averaging the individual covariances as

$$\mathbf{S}_c = \frac{1}{m} \sum_{i=1}^{m} \mathbf{S}_{gi} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{S} + \alpha \mathbf{\Phi}_i \mathbf{\Lambda} \mathbf{\Phi}_i^T.$$

This gives

$$\mathbf{S}_c = \mathbf{S} + \alpha \left(\frac{1}{m} \sum_{i=1}^{m} \mathbf{\Phi}_i \mathbf{\Lambda} \mathbf{\Phi}_i^T \right). \tag{55}$$

Now we find the eigenvectors and eigenvalues of $\mathbf{S}_c$ and proceed as with the normal PDM.

If we have only one example, then from (51) we get $\bar{\mathbf{x}} = \mathbf{x}_1$, and from (52) we get $\mathbf{S} = \mathbf{0}$, and thus we have only the effects of the FEM model in (55). If the number of samples is very large, then we wish to ignore the effects of the vibrational modes. This can be done by setting α to zero, and then $\mathbf{S}_c = \mathbf{S}$, and thus we obtain the PDM for the original example shapes. In the intermediate cases, we wish to have a large value for α when there are few original examples and smaller values as more examples become available. This may be achieved by setting $\alpha \propto 1/m$.

Extension to 3D Data The mathematical foundation formulation of PDMs can be directly extended to 3D. What remains is the specific implementation details of the representation of 3D surfaces and the search algorithm used. In [30] the surfaces are represented by a set of 3D landmarks. A vector of length $3L$ is used to represent a surface with L landmarks and can be written as

$$\mathbf{x} = \left[\; x_1, y_1, z_1 \quad x_2, y_2, z_2 \quad \ldots \quad x_L, y_L, z_L \; \right]^T.$$

In the training stage of the 2D PDM as well as in 3D, we need to provide the coordinates of the landmarks for each shape or surface in the training set. The

problem is solved in 2D, by providing a human expert with a utility to "point and click" landmarks on a 2D image. In 3D it is not trivial to build a utility that allows the operator to navigate in a 3D space of intensity values and locate the landmarks. In the currently discussed work, the segmentation is generated by hand in a slice-by-slice fashion and the aligning procedure allowed for rotation only in the slice plane, in addition to scaling and translation. In the search stage, the strongest edge or a representative profile is sought along a search profile normal to the 3D surface.

Kernel-Based Estimation of the Shape Density Distribution In Section 2.1.1 we represented the training set of shapes as a cloud of points in a $2L$-dimensional space. The ASD was defined to be a hyper-ellipsoid of some reduced dimensionality (given by the main mode of variation obtained by PCA). An alternative approach to model the allowable shape domain and utilize it in image search is as follows. As before, we are first presented with a set of example shapes represented by a vector in a multidimensional space that we align to a common coordinate frame. Now we estimate the shape probability density function (PDF) using a certain kernel, for example, convolving the shape points with a multidimensional Gaussian. In the search process, the gradient of the PDF can be used to lead us to a more plausible (or in this case, more probable) shape. The reader is referred to [27] for more practical details.

Classification Using ASMs The idea of using an ASM for classification can be explained by summarizing the work done in [34] on recognizing human faces. The shape model and the local gray level profile model are augmented with a model for the appearance of a shape-free face obtained by deforming each face to have the same shape as the mean face (using thin plate splines as described in [35]). In the training stage a number of training faces of the same and different individuals were outlined. In the application stage, an ASM is used to segment a face, and then the shape, shape-free, and local gray-level parameters are extracted and used as features for classification.

3. STATISTICALLY CONSTRAINED SNAKES: COMBINING ACMS AND ASMS

In this section we present a method for constraining the deformations of snakes in a way that is consistent with a training set of images. The described methodology is demonstrated by locating the left-ventricular boundary in echocardiographic images.

3.1. Introduction

Ultrasound echocardiography is a valuable noninvasive and relatively inexpensive tool for clinical diagnosis and analysis of heart functions including ventricular wall motion. An important step toward this analysis is segmentation of endocardial boundaries of the left ventricle (LV) [20, 36–40]. Although segmenting anatomical objects in high-SNR images can be done with simple techniques, problems do arise when the images are corrupted with noise and the object itself is not clearly or completely visible in the image. This is clearly the case in heart images obtained by ultrasonography, which are characterized by weak echoes, echo dropouts, and high levels of speckle noise. These image artifacts often result in detecting erroneous object boundaries or failing to detect true ones. Snakes [2] and its variants [11, 41–43] overcome parts of these limitations by considering the boundary as a single, inherently connected, and smooth structure, and also by supporting intuitive, interactive mechanisms for guiding the segmentation. In our application of locating the human LV boundary in echocardiography, human guidance is often needed to guarantee acceptable results. A potential remedy is to present the snake with a priori information about the typical shape of the LV. Statistical knowledge about shape variation can be obtained using PDMs, which are central to the ASMs segmentation technique [33]. PDMs, which are obtained by performing PCA on landmark coordinates labeled on many example images, have been applied to the analysis of echocardiograms [37]. However, this procedure is problematic since manual labeling of corresponding landmark points is required. In our application it is tedious to obtain a training data set delineated by experts with point correspondence, let alone the fact that defining a sufficient number of landmarks on the LV boundary is a challenging task in and of itself.

The method described in this section is similar to both ASMs, but without the landmark identification and correspondence requirement, and ACMs, but enforced with a priori information about shape variation. We adopt an approach similar to PDMs for capturing the main modes of ventricular shape variation. However, in our method, rather than representing the object boundaries by spatially corresponding landmarks, we employ a frequency-based boundary representation, namely Discrete Cosine Transform (DCT) coefficients. These new shape descriptors eliminate the need for spatial point correspondence. PCA, which is central to ASMs, is applied to this set of shape descriptors. An average object shape is extracted along with a set of significant shape variational modes. Armed with this model of shape variation, we find the boundaries in unknown images by placing an initial ACM and allowing it to deform only according to the examined shape variations. A similar approach using Fourier descriptors and applied to locating the corpus callosum in 2D MRI was reported in [60]. Results of segmenting the human LV in real echocardiographic data using the discussed methodology are also presented.

3.2. Methods

3.2.1. Overview

This section presents a general overview of the method (see Figure 15). We used snakes as the underlying segmentation technique. In order to arm the snake model with a priori information about the typical shape variations of the LV that may be encountered during the segmentation stage, a training set of images is provided. This set is manually delineated by medical experts without the requirement of complete landmark correspondence between different images. The entire set of manually traced contours is then studied to model the typical ventricular shape variations. This is done by first applying a re-parametrization of the contours, which gives a set of DCT coefficients replacing the spatial coordinates. We then apply PCA to find the strongest modes of shape variation. The result is an average ventricular shape, represented by a set of average DCT coefficients, in addition to the principal components, along with the fraction of variation each component explains. To segment a new image of the LV, we initialize a snake and, unlike classical snakes, do not allow it to freely deform according to internal and external energy terms, but instead we constrain its deformations in such a way that the resulting contour is similar to the training set. To attain the constrained deformations we obtain the vector of DCT coefficients for the active contour coordinates, project it onto an allowable snake space defined by the main principal components, and then perform an Inverse DCT (IDCT), which converts the constrained DCT coefficients back to spatial coordinates. This is repeated until convergence, which is reached when the majority of snake nodes do not change their locations significantly. The shape models generated are normalized with respect to the similarity transformation parameters: rotation angle, scaling factor, and two translation parameters.

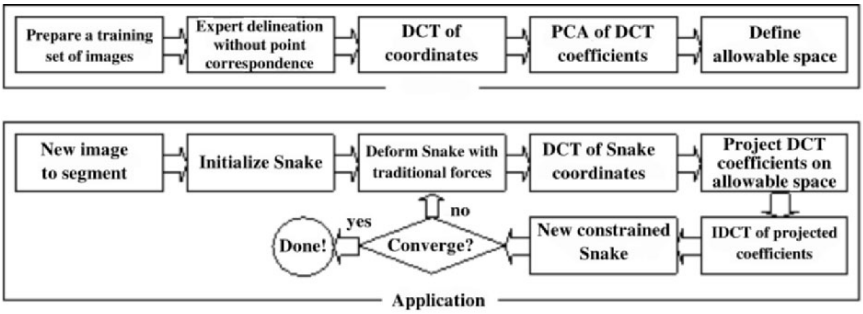


Figure 15. Flowchart depicting the main steps involved in the use of a statistically constrained snake for image segmentation. Reprinted with permission from [62]. Copyright ©2000, IEEE.

3.2.2. Representing Contours by DCT Coefficients

We use snakes as the underlying segmentation technique. A snake contour is originally represented by a set of N nodes $\{\mathbf{v}_i(t) = (x_i(t), y_i(t))\}$, i = 1, 2, ..., N, and is deformed according to internal and external forces. Snake contour re-parametrization is obtained via the use of the one-dimensional Discrete Cosine Transform (DCT) of the snake coordinates. The 1D DCT of sequence x_i of snake contour coordinates is defined as

$$X(k) = w(k) \sum_{i=1}^{N} x_i \cos \frac{\pi (2i-1)(k-1)}{2N}, k = 1, \ldots, N, \tag{56}$$

and the inverse DCT is given as

$$x_i = \sum_{k=1}^{N} w(k) X(k) \cos \frac{\pi (2i-1)(k-1)}{2N}, i = 1, \ldots, N, \tag{57}$$

where

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}}, & k = 1, \\ \sqrt{\frac{2}{N}}, & 2 \leq k \leq N, \end{cases} \tag{58}$$

and $X(k)$ are the DCT coefficients. Similar equations are used for the y_i coordinates and the $Y(k)$ DCT coefficients. The DCT was favored as the new frequency domain shape parametrization because it produces real coefficients, has excellent energy compaction properties, and the correspondence between the coefficients (when transforming contours with no point correspondence) is readily available. The latter property stems from the fact that corresponding DCT coefficients capture specific spatial contour frequencies.

3.2.3. Principal Component Analysis

In order to identify the main modes of shape variation found in the training contours, we perform PCA on the DCT coefficients representing them. The same number, say M, of DCT coefficients is obtained for the set of x and y coordinates that represent each shape in the training set. This is done either by interpolating the spatial coordinates or truncating the DCT coefficients. PCA yields the principal components (PCs) or main variation modes, $\mathbf{a}_j$, and the variance explained by each mode, λ_j. The first t PCs, sufficient to explain most of the variance, are used, i.e., $j = 1, 2, \ldots, t$. The average of the coefficient vectors, $\bar{X}$, is also calculated. The same procedure is performed for the y coordinates.

3.2.4. Constraining Contour Deformation

Subsequent to providing a set of images containing the object of interest, the training set of tracings is obtained (contours represented by coordinate-vectors

of varying length with no point correspondence). DCT coefficients (X) are then obtained followed by PCA. Presented with a new image, a snake contour is first initialized by specifying the starting and endpoints of the contour, and then allowed it to deform by applying forces that minimize traditional energy terms. In order to guarantee a snake contour resembling an acceptable shape (similar to those in the training set), we constrain the resulting deformed contour, $\{\mathbf{v}_i(t), i = 1, \ldots, N\}$, by projecting vector $\mathbf{X}$ (consisting of M DCT coefficients) onto the subspace of principal components (the allowable shape space) according to

$$X_{\text{proj}} = \bar{X} + \mathbf{A}\mathbf{b}, \tag{59}$$

where $\mathbf{b}$ is a vector of scalars weighting the main variation modes in $\mathbf{A}$ and is calculated as

$$\mathbf{b} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\left(X - \bar{X}\right), \tag{60}$$

and $\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \ldots & \mathbf{a}_t \end{bmatrix}$. Prior to performing the IDCT, we restrict the projected coefficients (X_{proj}) to lie within $\pm 3\sqrt{\lambda_j}$, since in this application the population typically lies within three standard deviations from the mean. Again, the same procedure is performed for the y coordinates. The statistical constraints can be applied after each snake deformation step or only when the resulting DCT coefficients of the snake contour are different (using some norm and threshold) from the mean coefficients. Note that the DCT coefficients are obtained and constrained for shapes normalized with respect to similarity transformation parameters utilizing corresponding starting and ending contour points.

3.3. Results

The described methodology was applied for segmenting the LV ventricle in real echocardiographic images. We collected 105 images of the human LV. The ventricular boundaries were manually traced by a medical expert. There was no point correspondence between the frames, with the number of traced points varying between 28 and 312 (Figures 16 and 17). The DCT of the manual tracings was then obtained.

Figure 18 shows an example of the manual tracings and the resulting contour after IDCT of the truncated DCT coefficients. The ratio "energy of truncated contour"/"energy of the original contour" for increasing numbers of DCT coefficients was examined in order to determine how many DCT coefficients to use. This was followed by a PCA of the truncated DCT coefficients. Five variation modes of 56 possible were enough to explain 95% of the total variation, 12 were enough for 99%, and 24 for 99.9%. Figure 19 depicts the first and second shape variation modes found in the training set.

To illustrate applying shape constraints we used test examples. We began with one of the manual tracings, added Gaussian noise, performed DCT, truncated certain DCT coefficients, projected the remaining coefficients on the allowable shape

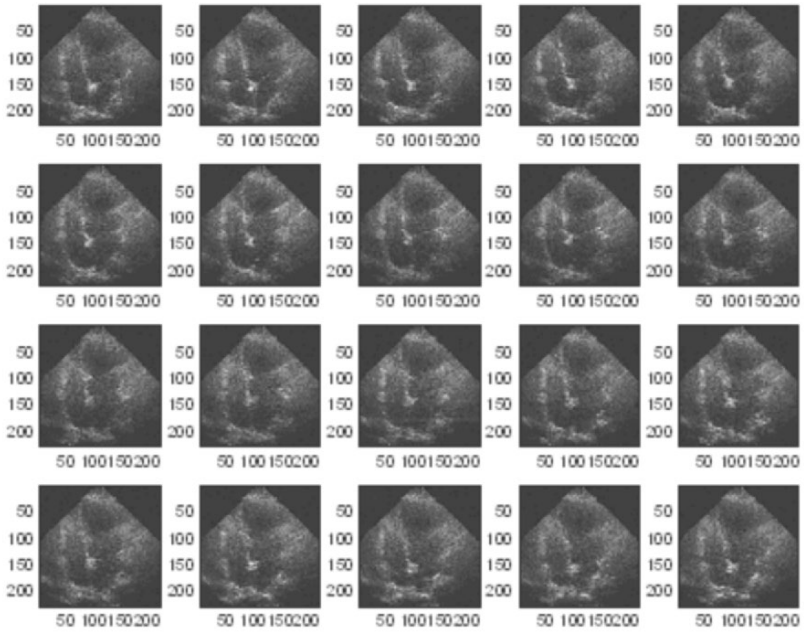


Figure 16. Sample of the echocardiographic training image set.

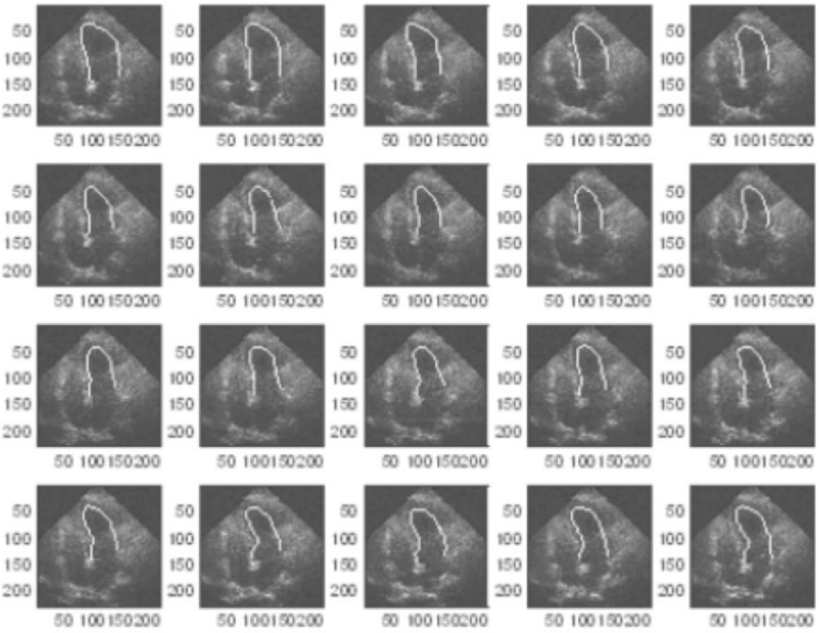


Figure 17. Manual tracings of the LV boundary in the training images.

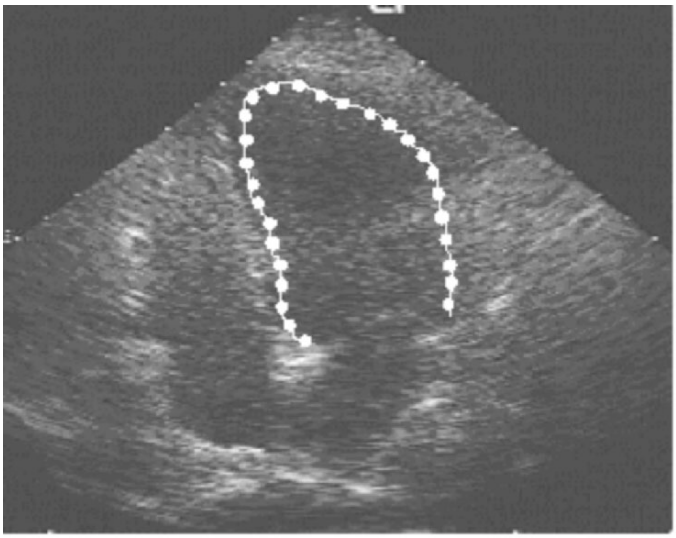


Figure 18. Ultrasound image with manual tracing (continuous) and the contour after IDCT of truncated DCT coefficients (dots). Reprinted with permission from [62]. Copyright ©2000, IEEE.

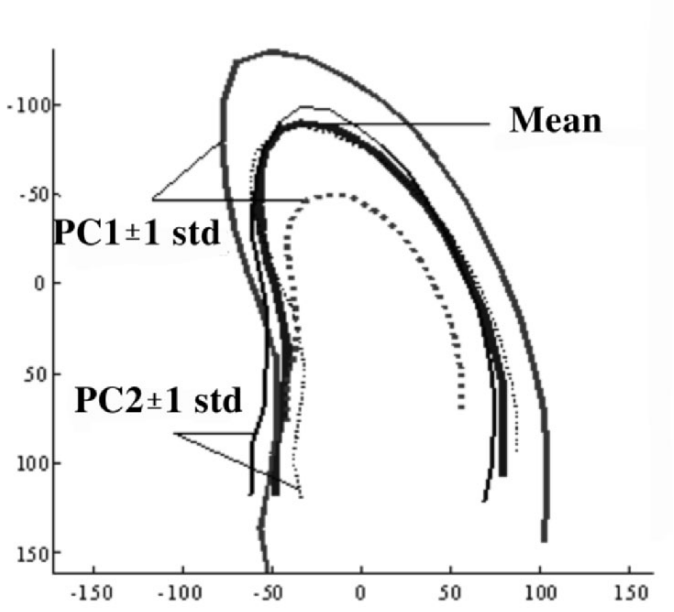


Figure 19. Mean contour and the first and second variation modes (weighted by ± 1 std). Reprinted with permission from [62]. Copyright ©2000, IEEE.

space, and then performed the IDCT. It was visually obvious how the constrained contour resembles a much more plausible boundary of the LV than the noisy one (Figure 20).

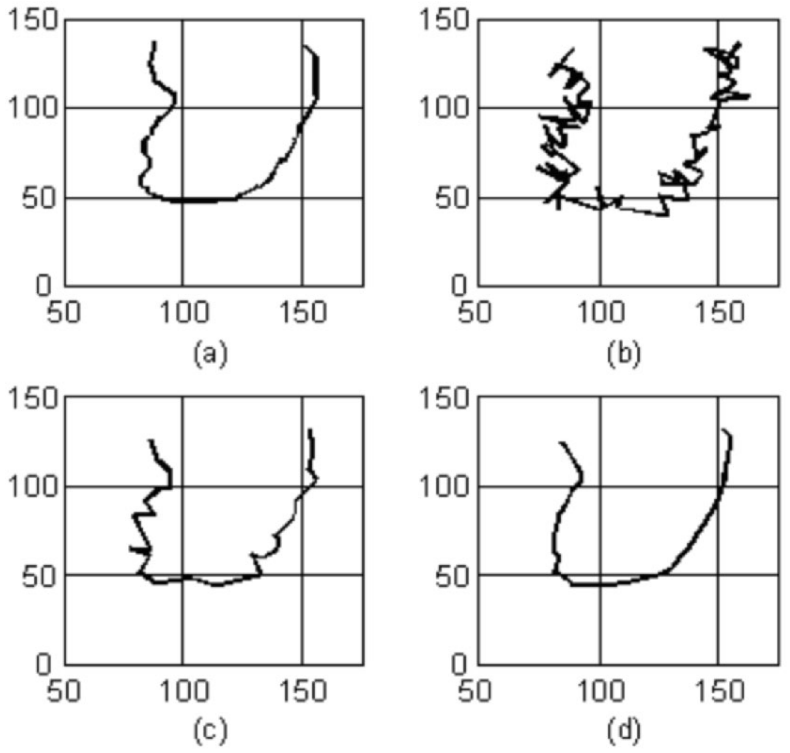


Figure 20. (a) Manual tracing. (b) Noisy version of (a). (c) IDCT of truncated DCT coefficients of (b). (d) Projection of (c) on the allowable shape space (note the similarity to (a)). Reprinted with permission from [62]. Copyright ©2000, IEEE.

More importantly, results on real echocardiographic data were obtained by initializing the snake on an image that was not included in the training set (i.e., cross-validation was used) and then allowing it to deform under forces that minimize its energy. This was followed by a DCT–Truncation–Projection–IDCT procedure. The outcome of the snake segmentation alone, due to noise and echo dropouts in the image, often gave unreasonable and unacceptable LV shapes. Conversely, employing constrained deformations resulted in acceptable LV boundaries (Figures 21 and 22).

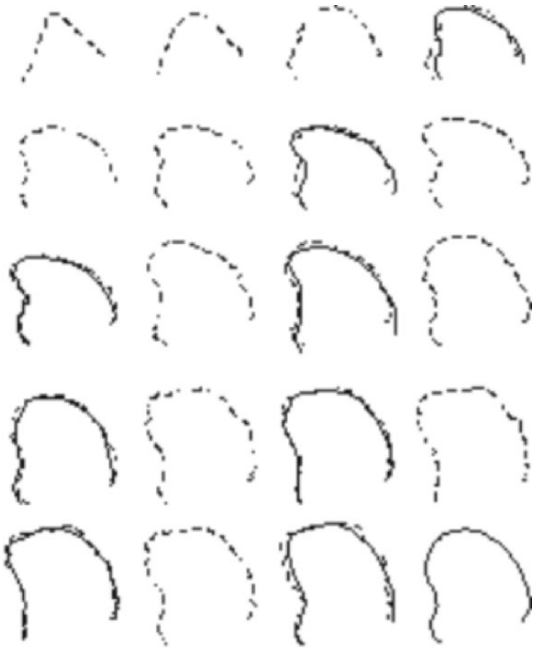


Figure 21. Snake contours (dashed) and constrained contours (continuous) with increasing number of iterations (left to right, top to bottom). Reprinted with permission from [62]. Copyright ©2000, IEEE.

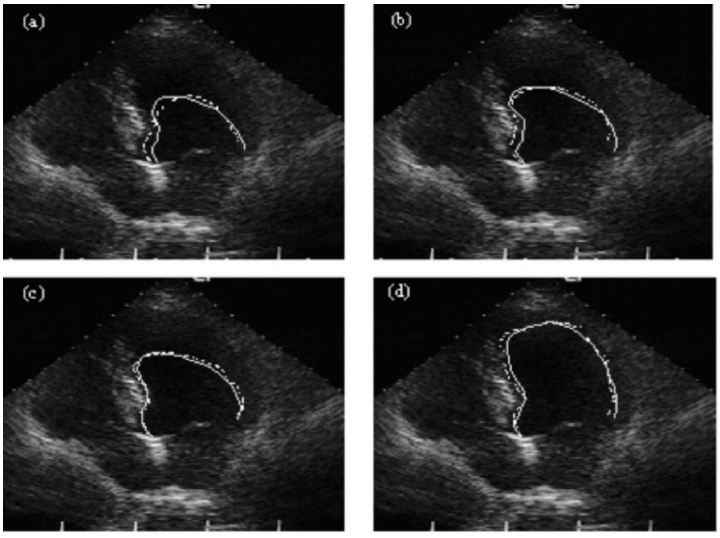


Figure 22. Progress ((a)–(d)) of a snake overlain on an ultrasound image of the left ventricle (dashed), and the result of DCT–Truncation–Projection–IDCT (continuous). Reprinted with permission from [62]. Copyright ©2000, IEEE.

3.4. Summary

We presented a method for constraining the deformations of an active contour according to training examples and applied it to segmenting the human left ventricle in echocardiographic (ultrasound) images. To capture the typical shape variations of the training set, principal component analysis was performed on frequency-domain shape descriptors in order to avoid the drawbacks associated with labeling corresponding landmarks (only the start and endpoints of the contours correspond). The method utilizes the strength of ACMs in producing smooth and connected boundaries along with the strength of ASMs in producing shapes similar to those in a training set. More plausible LV shapes resulted when employing the new method compared to classical snakes.

4. DEFORMABLE SPATIOTEMPORAL SHAPE MODELS: EXTENDING ACTIVE SHAPE MODELS TO 2D+TIME

In this section we extend 2D ASMs to 2D+time by presenting a method for modeling and segmenting spatiotemporal shapes (ST shapes). The modeling part consists of constructing a statistical model of ST shape parameters. This model describes the principal modes of variation of the ST shape in addition to constraints on the allowed variations. An active approach is used in segmentation where an initial ST shape is deformed to better fit the data, and the optimal proposed deformation is calculated using dynamic programming. Segmentation results on both synthetic and real data are presented.

4.1. Introduction

Much work has been done on tracking rigid objects in 2D sequences. In many image analysis applications, however, there is a need for modeling and locating non-rigid time-varying object shapes. One approach for dealing with such objects is the use of deformable models. Deformable models [59] such as snakes [2] and its variants [16, 42–45], have attracted considerable attention and are widely used for segmenting non-rigid objects in 2D and 3D (volume) images. However there are several well-known problems associated with snakes. They were designed as interactive models and therefore rely on a user to overcome initialization sensitivity. They were also designed as general models showing no preference among a set of equally smooth shapes. This generality can cause unacceptable results when snakes are used to segment objects with shape abnormalities arising from occlusion, closely located but irrelevant structures, or noise. Thus, techniques that incorporate a priori knowledge of object shape were introduced [46, 47]. In ASMs [46] the statistical variation of shapes is modeled beforehand in accordance with a training set of known examples. In order to attack the problem of tracking non-rigid time-varying objects, deformable models were extended to dynamic

deformable models [16, 29, 48–51]. These describe the shape changes (over time) in a single model that evolves through time to reach a state of equilibrium where internal forces, representing constraints on shape smoothness, balance the external image forces and the contour comes to rest. Deformable models have been constructed by applying a probabilistic framework and lead to techniques such as "Kalman snakes" [52]. Motion tracking using deformable models has been used for tracking non-rigid structures such as blood cells [48], and much attention has been given to the human heart and tracking of the left ventricle in both 2D and 3D [16, 49, 50, 53]. In addition to tracking rigid objects, previous work has focused on arbitrary non-rigid motion and gave little attention to tracking objects moving in specific motion patterns, without incorporation of statistical prior knowledge in both 2D and time [54].

We present a method for locating ST shapes in image sequences. We extend ASMs [46] to include knowledge of temporal shape variations and present a new ST shape modeling and segmentation technique. The method is well suited to model and segment objects with specific motion patterns, as in echocardiography.

4.2. Method

In order to model a certain class of ST shapes, a representative training set of known shapes is collected. The set should be large enough to include most of the shape variations we need to model. Next, all the ST shapes in the training set are parametrized. A data dimensionality reduction stage is then performed by capturing only the main modes of ST shape variations. In addition to constructing the ST shape model, the training stage also includes the modeling of gray-level information. The task is then to locate an ST shape given a new unknown image sequence. An average ST shape is first initialized, "optimal" deformations are then proposed, and the deformations are constrained to agree with the training data. The proposed changes minimize a cost function that takes into account both the temporal shape smoothness constraints and the gray-level appearance constraints. The search for the optimum proposed change is done using dynamic programming. The following sections present the various steps involved in detail.

4.2.1. Statistical ST Shape Variation

The Training Set We collect N training frame sequences each with F frames. The training set, $\Phi_V = [V_1, V_2, \ldots, V_N]$, displays similar objects and similar object motion patterns. $\Phi_V(i) = V_i = [f_{i1}, f_{i2}, \ldots, f_{iF}]$ is the ith frame sequence containing F frames, and $V_i(j) \equiv \Phi_V(i, j) = f_{ij}$ is the jth frame of the ith frame sequence containing the intensity value $f_{ij}(r, c) \equiv \Phi_V(i, j, r, c)$ at the rth row and cth column of the frame.

The ST Shape Parameters We introduce S_i to denote the parameter vector representing the ith ST shape. Parametrization is done using landmarks (other shape parametrization methods may be utilized, e.g., Fourier descriptors [55] or B-Splines [51]). Landmarks are labeled either manually, as when a cardiologist labels the heart chamber boundaries [30, 46], or (semi-)automatically [26]. Each landmark point is represented by its coordinate. Using L landmarks per frame and F frames per sequence, we can write the training set of ST shapes as $\Phi_S = [S_1, S_2, \ldots, S_N]$, where $\Phi_S(i) = S_i = [r_{i1}, r_{i2}, \ldots, r_{iF}]$ is the ith ST shape containing F shapes and $S_i(j) \equiv \Phi_S(i,j) = r_{ij}$ is the jth shape of the ith ST shape. r_{ij} can be written as $r_{ij} = [x_{ij1}, y_{ij1}, x_{ij2}, y_{ij2}, \ldots, x_{ijL}, y_{ijL}]$, where $x_{ijk} = r_{ij}(k,1) \equiv \Phi_S(i,j,k,1)$ and $y_{ijk} = r_{ij}(k,2) \equiv \Phi_S(i,j,k,2)$ are the coordinates of the kth landmark of the shape r_{ij}.

ST Shape Alignment Next, the ST shapes are aligned in order to allow comparing equivalent points from different ST shapes. This is done by rotating, scaling, and translating the shape in each frame of the ST shape by an amount that is fixed within one ST shape. A weighted least-squares approach is used for aligning two sequences and an iterative algorithm is used to align all the ST shapes. Given two ST shapes,

$$\begin{aligned} S_1 \quad = \quad & [x_{111}, y_{111}, \ldots, x_{11L}, y_{11L}, x_{121}, y_{121}, \ldots, x_{12L}, y_{12L}, \ldots\ldots, \\ & x_{1F1}, y_{1F1}, \ldots, x_{1FL}, y_{1FL}] \end{aligned}$$

and

$$\begin{aligned} S_2 \quad = \quad & [x_{211}, y_{211}, \ldots, x_{21L}, y_{21L}, x_{221}, y_{221}, \ldots, x_{22L}, y_{22L}, \ldots\ldots, \\ & x_{2F1}, y_{2F1}, \ldots, x_{2FL}, y_{2FL}], \end{aligned}$$

we need to find rotation angle θ, scaling factor s, and the value of translation (t_x, t_y) that will align S_2 to S_1. To align S_2 to S_1, S_2 is mapped to

$$\begin{aligned} \hat{S}_2 \quad = \quad & [\hat{x}_{211}, \hat{y}_{211}, \ldots, \hat{x}_{21L}, \hat{y}_{21L}, \hat{x}_{221}, \hat{y}_{221}, \ldots, \hat{x}_{22L}, \hat{y}_{22L}, \ldots\ldots, \\ & \hat{x}_{2F1}, \hat{y}_{2F1}, \ldots, \hat{x}_{2FL}, \hat{y}_{2FL}], \end{aligned}$$

using $\hat{S}_2 = M(s,\theta)[S_2] + \mathbf{t}$, where $M(s,\theta)[S_2]$ is a rotated then scaled version of each coordinate of S_2 (by θ and s, respectively), and $\mathbf{t} = [t_x, t_y, t_x, t_y, ..., t_x, t_y]^T$ is a translation vector of length $2FL$. The weighted distance between S_1 and $\hat{S}_2$ in the $2FL$-dimensional space is given by

$$d^2_{1\hat{2}} = (\hat{S}_2 - S_1)^T \mathbf{W}^T \mathbf{W} (\hat{S}_2 - S_1),$$

where

$$\begin{aligned} \mathbf{W} \quad = \quad & diag(w_{11x}, w_{11y}, \ldots, w_{1Lx}, w_{1Ly}, w_{21x}, w_{21y}, \ldots, w_{2Lx}, w_{2Ly}, \ldots\ldots, \\ & w_{F1x}, w_{F1y}, \ldots, w_{FLx}, w_{FLy}). \end{aligned}$$

The elements of $\mathbf{W}$ reflect our trust in each coordinate and are chosen to be proportional to the "stability" of the different landmarks over the training set [46].

To rotate, scale, and translate a single coordinate, (x_{2kl}, y_{2kl}), we use $\begin{bmatrix} \hat{x}_{2kl} \\ \hat{y}_{2kl} \end{bmatrix} = \begin{bmatrix} a_x & -a_y \\ a_y & a_x \end{bmatrix} \cdot \begin{bmatrix} x_{2kl} \\ y_{2kl} \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$, where $a_x = s\cos(\theta)$ and $a_y = s\sin(\theta)$. $\hat{S}_2$ can now be rewritten as $\hat{S}_2 = \mathbf{Az}$, where $\mathbf{z} = [a_x, a_y, t_x, t_y]^T$ and

$$\mathbf{A}^T = \begin{bmatrix} x_{211} & y_{211} \ldots x_{21L} & y_{21L} x_{221} & y_{221} \ldots x_{22L} & y_{22L} \ldots \ldots x_{2F1} & y_{2F1} \ldots x_{2FL} & y_{2FL} \\ -y_{211} x_{211} \ldots -y_{21L} x_{21L} & -y_{221} x_{221} \ldots -y_{22L} x_{22L} \ldots \ldots -y_{2F1} x_{2F1} \ldots -y_{2FL} x_{2FL} \\ 1 \quad 0 \quad \ldots 1 \quad 0 & 1 \quad 0 \quad \ldots 1 \quad 0 & \ldots \ldots 1 \quad 0 & \ldots 1 \quad 0 \\ 0 \quad 1 \quad \ldots 0 \quad 1 & 0 \quad 1 \quad \ldots 0 \quad 1 & \ldots \ldots 0 \quad 1 & \ldots 0 \quad 1 \end{bmatrix}.$$

Distance $d^2_{1\hat{2}}$ can now be rewritten as $d^2_{1\hat{2}} = (\mathbf{Az} - S_1)^T \mathbf{W}^T \mathbf{W} (\mathbf{Az} - S_1)$, and we can solve for $\mathbf{z}$ (least-squares solution) that minimizes $d^2_{1\hat{2}}$ according to $\mathbf{z} = ((\mathbf{WA})^T (\mathbf{WA}))^{-1} (\mathbf{WA})^T \mathbf{W} S_1) = (\mathbf{A}^T \mathbf{W}^T \mathbf{WA})^{-1} \mathbf{A}^T \mathbf{W}^T \mathbf{W} S_1$. Once $\mathbf{z} = [a_x, a_y, t_x, t_y]^T$ is calculated, $s \quad \theta$ can be found using $\theta = \arctan(\frac{a_y}{a_x})$ and $s = a_x / \cos(\arctan(\frac{a_y}{a_x}))$. We note that when the observed motion patterns in the training sequences span different time intervals, temporal resampling or aligning that incorporates temporal scaling is performed.

Main ST Shape Variation Modes The N aligned ST shapes, each of length $2FL$ and represented by $\{S_1, S_2, \ldots, S_N\}$, map to a "cloud" of N points in a $2FL$-dimensional space. It is assumed that these N points are contained within a hyper-ellipsoid of this $2FL$-dimensional space. We call this region the Allowable ST Shape Domain (ASTSD). We then apply PCA to the aligned training set of ST shapes in order to find the main modes of ST shape variation. The resulting principal components (PCs) are the eigenvectors $\mathbf{p}_k$ ($1 \leq k \leq 2FL$) of the covariance matrix of the observations, C_S, found from $C_S \mathbf{p}_k = \lambda_k \mathbf{p}_k$. λ_k is the kth eigenvalue of C_S ($\lambda_k \geq \lambda_{k+1}$) and is equal to the variance along the kth PC. The mean ST shape is calculated as $\bar{S} = \frac{1}{N} \sum_{i=1}^{N} S_i$. The PCs are normalized to unit length and are mutually orthogonal.

Model Representation We now express each ST shape, S_i, as the sum of the mean ST shape, $\bar{S}$, and a linear combination of the principal modes of variation, $\mathbf{Pb}_i$. This gives $S_i = \bar{S} + \mathbf{Pb}_i$, where $\mathbf{b}_i = [b_{i,1}, b_{i,2}, ..., b_{i,2FL}]^T$ and $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_{2FL}]$. We constrain b_l to $b_{l\min} \leq b_l \leq b_{l\max}$ with $b_{l\min} = -b_{l\max}$ and $1 \leq l \leq 2FL$. $b_{l\max}$ is chosen to be proportional to $\sqrt{\lambda_l}$ ($-3\sqrt{\lambda_l} \leq b_l \leq 3\sqrt{\lambda_l}$ is typically used). In practice, only the first t (out of $2FL$) PCs explaining a sufficiently high percentage of the total variance of the original data are used, and the fundamental equation becomes

$$S = \bar{S} + \mathbf{P}_t \mathbf{b}, \tag{61}$$

where $\mathbf{b} = [b_1, b_2, ..., b_t]^T$, $\mathbf{P}_t = [\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_t]$, and the constraints on $\mathbf{b}$ become $b_{l\,\min} \leq b_l \leq b_{l\,\max}$, where $1 \leq l \leq t$.

4.2.2. Gray-Level Training

The information contained in the ST shape model alone is typically not enough for spatiotemporal segmentation. Therefore, additional representative information about the intensities or gray levels relating to the object is also desired and collected in the gray-level training stage. In the search stage, new estimates of the ST shape are sought that will better match the gray-level prior knowledge. Different gray-level representative information can be used, e.g., gathering the intensity values in the entire patch contained within the object [28] or parametrizing the profiles or patches around the landmark. In this implementation we follow [46] and use a mean normalized derivative (difference) profile, passing through each landmark and perpendicular to the boundary created by the neighboring ones. For the kth landmark this profile is given by

$$\bar{\mathbf{y}}_k = \frac{1}{FN} \sum_{j=1}^{F} \sum_{i=1}^{N} \mathbf{y}_{ijk}, \tag{62}$$

where $\mathbf{y}_{ijk}$ is the representative profile for the kth landmark in the jth shape of the ith ST shape. Using gray-level information, temporal and shape constraints, the model is guided to a better estimate of the dynamic object hidden in the new frame sequence.

4.2.3. ST Shape Segmentation Algorithm

Given a new frame sequence, the task is to locate the object in all the frames or equivalently locate the ST shape. An initial estimate of the ST shape parameters is chosen at first, and then changes to the parameters are proposed. The pose of the current estimate is then changed and suitable weights for the modes of variation chosen in order to fit the model to the proposed changes. This is done with the restriction that the changes can only be made in accordance with the model (with reduced dimensionality) and the training set. New changes are then proposed, and so on. Here we present a detailed discussion of these steps.

Initial estimate. The search starts by guessing an initial ST shape:

$$\hat{S}^{\langle 0 \rangle} = M\left(s^{\langle 0 \rangle}, \theta^{\langle 0 \rangle}\right)\left[\bar{S} + \mathbf{P}_t \mathbf{b}^{\langle 0 \rangle}\right] + \mathbf{t}^{\langle 0 \rangle}, \tag{63}$$

where $\mathbf{t} = [t_x, t_y, t_x, t_y, ..., t_x, t_y]^T$ is of length $2FL$. $M(s, \theta)[S] + \mathbf{t}$ scales, rotates, and translates S by s, θ, and $\mathbf{t}$, respectively. Both $\bar{S}$ and $\mathbf{P}_t$ are obtained from the training stage. A typical initialization would set $\mathbf{b}^{\langle 0 \rangle}$ to zero and set $s^{\langle 0 \rangle}$, $\theta^{\langle 0 \rangle}$, and $\mathbf{t}^{\langle 0 \rangle}$ to values that place the initial sequence in the vicinity of the target.

Proposing a new sequence. For each landmark, say the kth landmark in the jth frame, we define a search profile $\mathbf{h}_{jk} = [h_{jk1}, h_{jk2}, ..., h_{jkH}]$ that is differentiated and normalized as done with the training profiles. This gives H^F possibilities for the proposed positions of the kth landmarks in the F frames (see Figure 23).

Since locating the new positions (1 out of H^F possible) is computationally demanding, we formulate the problem as a multistage decision process and use dynamic programming [41] to find the optimum proposed landmark positions by minimizing a cost function. The cost function comprises two terms: one due to large temporal landmark position changes, and another reflecting the mismatch between the gray-level values surrounding the current landmarks and those expected values found in the gray-level training stage. In the following paragraphs, we detail our implementation of dynamic programming.

We calculate a gray-level mismatch value $M_k(j, l)$ for each point along each search profile in all the frames according to

$$M_k(j,l) = (\mathbf{h}_{jk}(l) - \bar{\mathbf{y}}_k)^T \mathbf{W}^T \mathbf{W} (\mathbf{h}_{jk}(l) - \bar{\mathbf{y}}_k), \tag{64}$$

where $1 \le k \le L$, $1 \le j \le F$, $1 \le l \le H$, $\mathbf{h}_{jk}(l)$ is a subprofile of length $G-1$ anchored at the lth location of the search profile $\mathbf{h}_{jk}$, and $\mathbf{W}$ is a diagonal weighting matrix ($\mathbf{W} = \mathbf{I}$ was used). Additionally, we calculate a temporal discontinuity value, $d_{jk}(l_j, l_{j-1})$, corresponding to moving the kth landmark in frame $j-1$ to location l_{j-1}, and the kth landmark in frame j to location l_j, each along its respective search profile, according to

$$d_{jk}^2(l_j, l_{j-1}) = (\mathbf{c}_{jkx}(l_j) - \mathbf{c}_{j-1kx}(l_{j-1}))^2 + (\mathbf{c}_{jky}(l_j) - \mathbf{c}_{j-1ky}(l_{j-1}))^2, \tag{65}$$

where $\mathbf{c}_{jkx} = [x_{jk1}, x_{jk2}, ..., x_{jkH}]$ and $\mathbf{c}_{jky} = [y_{jk1}, y_{jk2}, ..., y_{jkH}]$ are the search profile coordinates of the kth landmark in the jth frame. We compare the accumulated costs of moving the kth landmark to the lth position in the jth frame, $2 \le j \le F$, from any of the H positions in frame $j-1$ and assign the least value to $A_k(j, l)$, i.e.,

$$A_k(j,l) = \min\{t_{jkl1}, t_{jkl2}, \ldots, t_{jklH}\} \tag{66}$$

$$t_{jklm} = w_d d_{jk}(l,m) + w_m M_k(j,l) + A_k(j-1,m), \tag{67}$$

w_d and w_m satisfy $w_d + w_m = 1$, control the relative importance of temporal discontinuity and gray-level mismatch. We also assign an index or a pointer, $P_k(j, l)$, to the location of the best landmark in the previous frames. Applying the same procedure to the kth landmark in all the F frames yields $F \times H$ accumulated values and $F \times H$ pointers (no temporal discontinuity cost is associated with the first frame).

To find the proposed positions of the kth landmark in all the frames we find the location, call it m_F, of the minimum accumulated cost along the search profile of

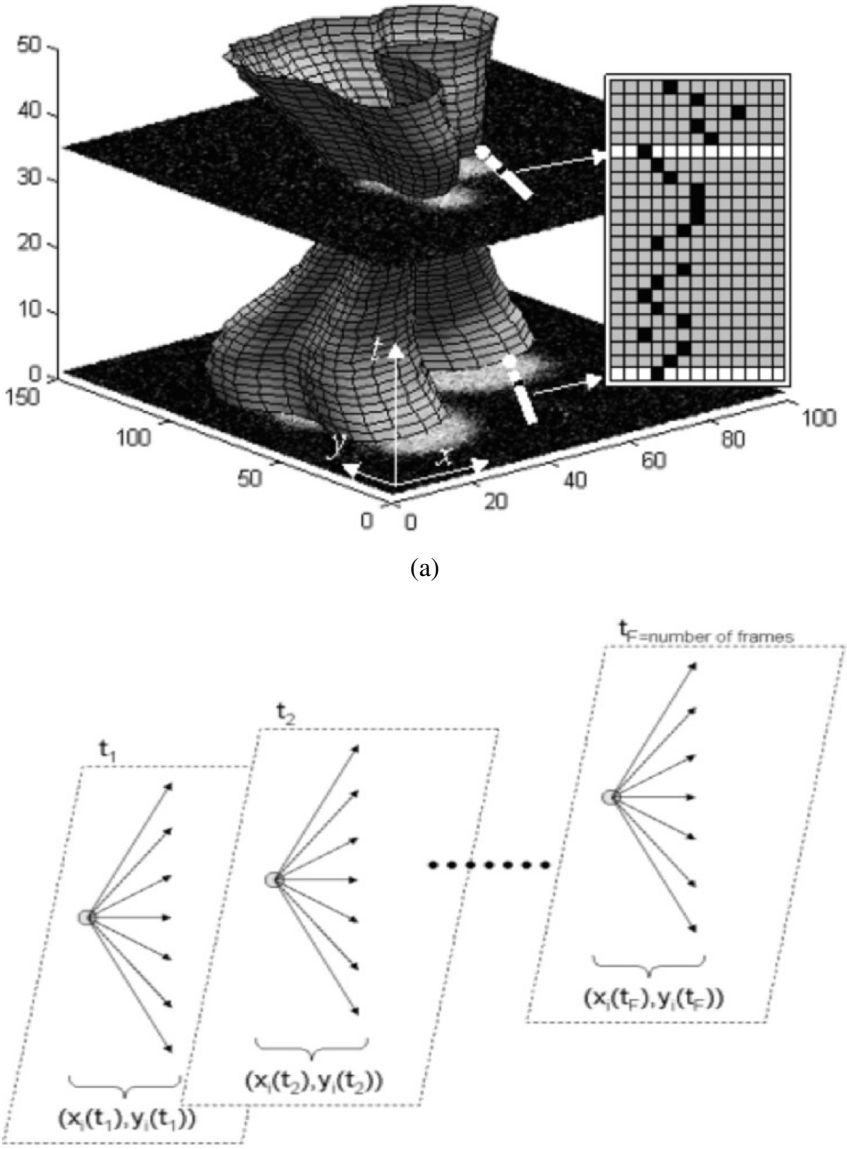


Figure 23. Proposing a new ST shape. (a) An illustration of an ST shape overlain on an image sequence. The search profiles of one landmark in two frames are shown in white. Examples of proposed landmark positions are shown as black squares. (b) The different choices of the new positions of landmark i in all frames. Reprinted with permission from [63]. Copyright ©2004, Elsevier.

the landmark in the last frame, frame F. Then we use m_F to find the proposed landmark position in the second-to-last frame, frame $F-1$, as $m_{F-1} = P_k(F, m_F)$. Its coordinates will be $(\mathbf{c}_{F-1kx}(m_{F-1}), \mathbf{c}_{F-1ky}(m_{F-1}))$. In general the proposed coordinates of the kth landmark of the jth frame will be

$$(x, y) : (\mathbf{c}_{jkx}(m_j), \mathbf{c}_{jky}(m_j)), \tag{68}$$

$$m_j = P_k(j+1, m_{j+1}). \tag{69}$$

Tracking back to the first frame, we acquire the coordinates of the proposed positions of the kth landmark in all frames. Similarly, we obtain the proposed positions for all the landmarks ($1 \leq k \leq L$), which define the ST shape changes $d\hat{S}^{\langle 0\rangle}_{\text{proposed}}$.

Limiting the proposed sequence. Since the proposed ST shape ($\hat{S}^{\langle 0\rangle} + d\hat{S}^{\langle 0\rangle}_{\text{proposed}}$) will generally not conform to our model of reduced dimensionality and will not lie in the ASTSD, it cannot be accepted as an ST shape estimate. Therefore, we need to find an acceptable ST shape that is closest to the proposed one. This is done by first finding the pose parameters ($s^{\langle 1\rangle}$, $\theta^{\langle 1\rangle}$, and $\mathbf{t}^{\langle 1\rangle}$) that will align $\bar{S}$ to $\hat{S}^{\langle 0\rangle} + d\hat{S}^{\langle 0\rangle}_{\text{proposed}}$ by mapping $\bar{S}$ to $M\left(s^{\langle 1\rangle}, \theta^{\langle 1\rangle}\right)\left[\bar{S}\right] + \mathbf{t}^{\langle 1\rangle}$, and then finding the extra ST shape modifications $dS^{\langle 1\rangle}$ that, when combined with the pose parameters, will map exactly to $\hat{S}^{\langle 0\rangle} + d\hat{S}^{\langle 0\rangle}_{\text{proposed}}$. The latter is done by solving the following equation for $dS^{\langle 1\rangle}$:

$$M\left(s^{\langle 1\rangle}, \theta^{\langle 1\rangle}\right)\left[\bar{S} + dS^{\langle 1\rangle}\right] + \mathbf{t}^{\langle 1\rangle} = \hat{S}^{\langle 0\rangle} + d\hat{S}^{\langle 0\rangle}_{\text{proposed}} \Rightarrow \tag{70}$$

$$dS^{\langle 1\rangle} = M\left(s^{\langle 1\rangle}, \theta^{\langle 1\rangle}\right)^{-1}\left[\hat{S}^{\langle 0\rangle} + d\hat{S}^{\langle 0\rangle}_{\text{proposed}} - \mathbf{t}^{\langle 1\rangle}\right] - \bar{S}, \tag{71}$$

where $M\left(s^{\langle 1\rangle}, \theta^{\langle 1\rangle}\right)^{-1} = M\left((s^{\langle 1\rangle})^{-1}, -\theta^{\langle 1\rangle}\right)$. In order to find the new shape parameters, $\mathbf{b}^{\langle 1\rangle}$, we need to solve $dS^{\langle 1\rangle} = \mathbf{P}_t\mathbf{b}^{\langle 1\rangle}$, which, in general, has no solution since $dS^{\langle 1\rangle}$ lies in a $2FL$-dimensional space, whereas $\mathbf{P}_t$ spans only a t-dimensional space. The best least-squares solution is obtained as

$$\mathbf{b}^{\langle 1\rangle} = \mathbf{P}_t^T dS^{\langle 1\rangle}. \tag{72}$$

Finally, using the constraints discussed earlier, $b_{l\,\min} \leq b_l \leq b_{l\,\max}$, where $1 \leq l \leq t$, we limit these ST shape variations and obtain an acceptable or allowable shape within the ASTSD. By updating $\mathbf{b}^{\langle 0\rangle}$ to $\mathbf{b}^{\langle 1\rangle}$, we have the new values for all the parameters $s^{\langle 1\rangle}$, $\theta^{\langle 1\rangle}$, $\mathbf{b}^{\langle 1\rangle}$, and $\mathbf{t}^{\langle 1\rangle}$.

Updating the estimate and reiterating. Similarly, new ST shape estimates can be obtained:

$$\begin{aligned}&\hat{S}^{\langle i\rangle} = M\left(s^{\langle i\rangle}, \theta^{\langle i\rangle}\right)\left[\bar{S} + \mathbf{P}_t\mathbf{b}^{\langle i\rangle}\right] + \mathbf{t}^{\langle i\rangle} \rightarrow \\ &\hat{S}^{\langle i+1\rangle} = M\left(s^{\langle i+1\rangle}, \theta^{\langle i+1\rangle}\right)\left[\bar{S} + \mathbf{P}_t\mathbf{b}^{\langle i+1\rangle}\right] + \mathbf{t}^{\langle i+1\rangle}\end{aligned} \tag{73}$$

for $i = 1, 2, 3, \ldots$. Checking for convergence can be done by examining the changes, i.e., if the new estimate is not much different (according to some predefined threshold), then the search is completed; otherwise we reiterate.

4.3. Results

We present results of locating the spatiotemporal shape of the left ventricle in real echocardiographic image sequences. The training data set consisted of 6 frame sequences, each sequence including 21 frames, and each frame of size 255×254 pixels (i.e., the size of $\Phi_V = 6 \times 21 \times 255 \times 254$). The number of (x, y) landmark coordinates in each frame was 25 (size of $\Phi_S = 6 \times 21 \times 25 \times 2$). Three ST shape parameters were used to explain 94.2% of the total ST shape variations. The gray-level search was conducted on a profile of length 60 pixels, and the training profile was of length 26 pixels. Figure 24 illustrates how statistical spatiotemporal prior knowledge is used to constrain the proposed segmentation and produce the final left-ventricular segmentation. Figure 25 shows additional segmentation results.

We applied this method to segmenting astrocyte cells in a 3D fluorescence image, where the spatial z-axis replaces time. The training data set consisted of 8 volumes (out of 9, leave-one-out validation), and each included 11 image slices, each image being of size 128×128 pixels (i.e., the size of $\Phi_V = 8 \times 11 \times 128 \times 128$). The number of (x, y) landmark coordinates in each slice was 40 (size of $\Phi_S = 8 \times 11 \times 40 \times 2$). Seven shape parameters were used to explain 99.5% of the total shape variations. The gray-level search was conducted on a profile of length 40 pixels, and the training profile was of length 12 pixels. Figure 26 illustrates example segmentation results.

4.4. Summary

Motivated by the fact that many image analysis applications require robust methods for representing, locating, and analyzing non-rigid time-varying shapes, we presented an extension of a 2D ASM to 2D+time. This method models the gray-level information and the spatiotemporal variations of a time-varying object in a training set. The model was then used for locating similar moving objects in a new image sequence. The segmentation technique was based on deforming a spatiotemporal shape to better fit the image sequence data only in ways consistent with the training set. The proposed deformations were calculated by minimizing an energy function using dynamic programming. The energy function included terms reflecting temporal smoothness and gray-level information constraints.

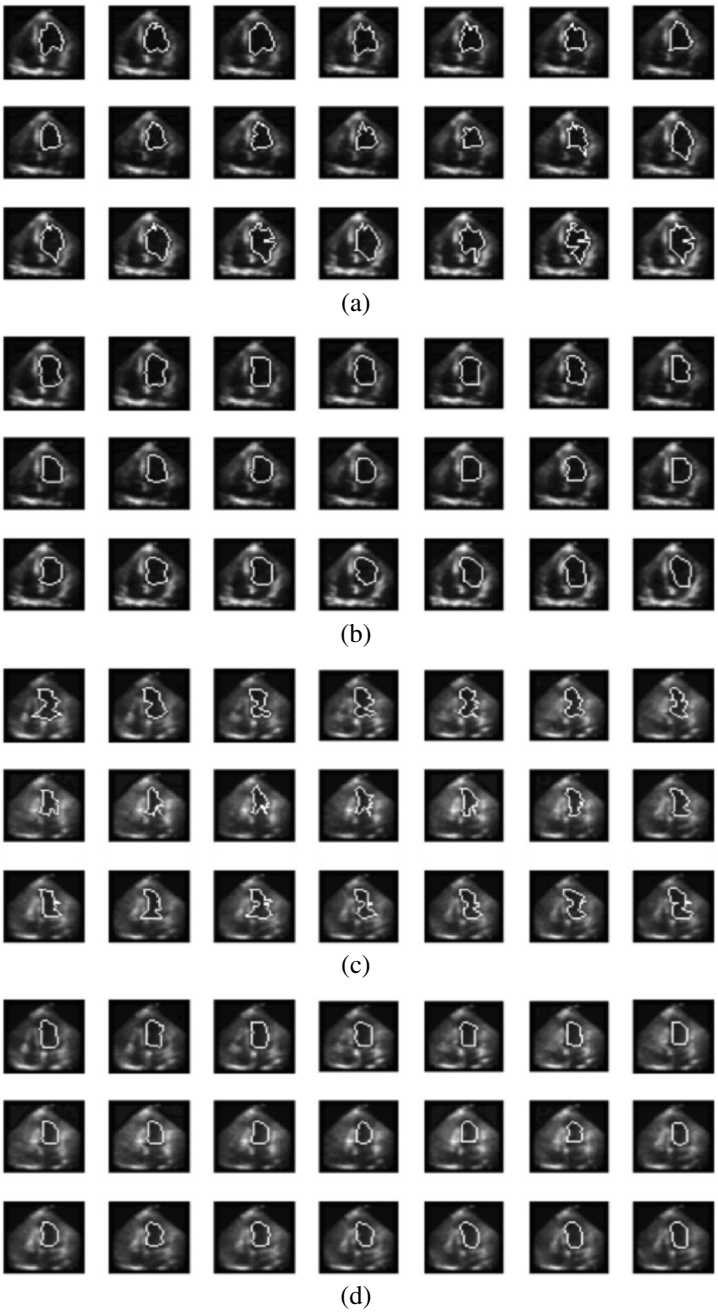


Figure 24. Left-ventricular segmentation result from two echocardiographic image sequences. Ultrasound frames are shown with the ST shape overlain (a,c) before and (b,d) after projection onto the ASTSD (frames progress from left to right, top to bottom). Reprinted with permission from [63]. Copyright ©2004, Elsevier.

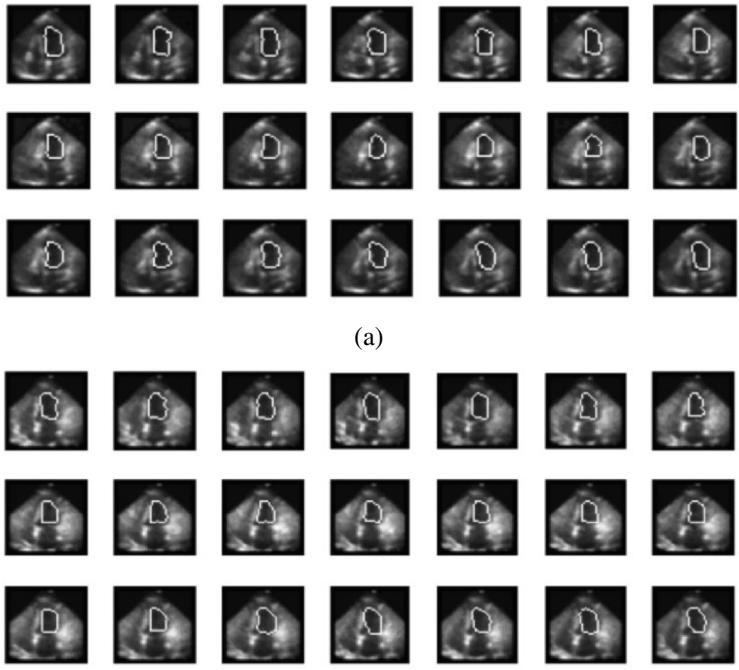


Figure 25. Additional left-ventricular segmentation results from an echocardiographic image sequence (frames progress from left to right, top to bottom). Reprinted with permission from [63]. Copyright ©2004, Elsevier.

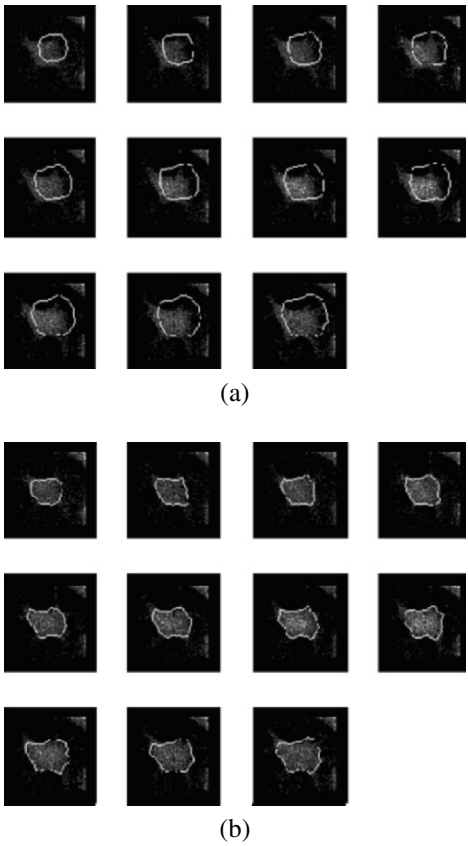


Figure 26. Segmenting a 3D astrocyte cell (spatial z-axis replaces time): (a) initial shape model and (b) segmentation result overlain in white on a fluorescence 3D image. Reprinted with permission from [63]. Copyright ©2004, Elsevier.

5. REFERENCES

1. Terzopoulos D. 1987. *On matching deformable models to images*. Technical Report 60, Schlumberger Palo Alto Research, 1986. Reprinted in *Topical Meeting on MachineVision*, Technical Digest Series, Vol. 12, pp. 160–167.
2. Kass M, Witkin A, Terzopoulos D. 1987. Snakes: active contour models. *Int J Comput Vision* **1**(4):321–331.
3. Widrow B. 1973. The rubber mask technique, part I. *Pattern Recognit* 5(3):175–211.
4. Fischler M, Elschlager R. 1973. The representation and matching of pictorial structures. *IEEE Trans Comput* **22**(1):67–92.

5. Blake A, Isard M. 2000. *Active contours*. Berlin: Springer.
6. Elsgolc LE. 1962. *Calculus of variations*. Reading, MA: Addison-Wesley.
7. McInerney T, Terzopoulos D. 1996. Deformable models in medical image analysis: a survey. *Med Image Anal* **1**(2):91–108.
8. Sonka M, Hlavac V, Boyle R. 1998. *Image processing, analysis, and machine vision*, 2nd ed. Pacific Grove, CA: Brooks/Cole.
9. Cohen LD, Cohen. 1993. Finite-element methods for active contour models and balloons for 2D and 3D images. *IEEE Trans Pattern Anal Machine Intell* **15**:1131–1147.
10. Leroy B, Herlin I, Cohen LD. 1996. Multi-resolution algorithms for active contour models. In *Proceedings of the 12th international conference on analysis and optimization of systems*, pp. 58–65. Washington, DC: IEEE Computer Society.
11. McInerney T, Terzopoulos D. 2000. T-snakes: topology adaptive snakes. *Med Image Anal* **4**(2):73–91.
12. Ivins J, Porrill J. 1994. Statistical snakes: active region models. In *Proceedings of the 5th British machine vision conference (BMVC'94)*, pp. 377–386. Surrey, UK: BMVA Press.
13. Xu C, Prince JL. 1998. Snakes, shapes, and gradient vector flow. *IEEE Trans Image Process* **7**(3):359–369.
14. Sapiro G. 1997. Color snakes. *Comput Vision Image Understand* **68**(2):247–253.
15. Delingette H. 1999. General object reconstruction based on simplex meshes. *Int J Comput Vision* **32**(2):111–146.
16. McInerney T, Terzopoulos D. 1995. A dynamic finite-element surface model for segmentation and tracking in multidimensional medical images with application to cardiac 4D image analysis. *Comput Med Imaging Graphics* **19**(1):69–83.
17. Hamarneh G, Althoff K, Gustavsson T. 2000. Snake deformations based on optical flow forces for contrast agent tracking in echocardiography. In *Proceedings of the Swedish Symposium on image analysis*, pp. 45–48. Available online at http://db.s2.chalmers.se/download/publications/hamarneh_824.pdf.
18. Althoff K, Hamarneh G, Gustavsson T. 2000. Tracking contrast in echocardiography by a combined snake and optical flow technique. *IEEE Proc Comput Cardiol* **27**:29–32.
19. McKenna WJ, Thiene G, Nava A, Fontaliran F, Blomström-Lundqvist C, Fontaine G, Camerini F. 1994. Diagnosis of arrhythmogenic right ventricular dysplasia/cardiomyopathy. *Br Heart J* **71**215–218.
20. Mikic I, Krucinski S, Thomas JD. 1998. Segmentation and tracking in echocardiographic sequences: active contours guided by optical flow estimates. *IEEE Trans Med Imaging* **17**(2):274–284.
21. Peterfreund N. 1999. Robust tracking of position and velocity with Kalman snakes. *IEEE Trans Pattern Anal Machine Intell* **21**(6):564–569.
22. Akgul YS, Kambhamettu C, Stone M. 1998. Extraction and tracking of the tongue surface from ultrasound image sequences. *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR 1991)*, pp. 298–303. Washington, DC: IEEE Computer Society.
23. Horn B, Schunk B. 1981. Determining optical flow. *Artif Intell* **17**185–204.
24. Perona P, Malik J. 1990. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans Pattern Anal Machine Intell* **12**:629-639.
25. Cootes T, Taylor C, Hill A, Halsam J. 1993. The use of active shape models for locating structures in medical images. In *Proceedings of the 13th international conference on information processing in medical imaging*, pp. 33–47. Ed HH Barrett, AF Gmitro. Berlin: Springer.
26. Hill A, Taylor C. 1994. Automatic landmark generation for point distribution models. In *Proceedings of the 5th British machine vision conference (BMVC'94)*, pp. 429–438. Surrey, UK: BMVA Press.

27. Cootes T, Taylor C, Lanitis A. 1994. Active shape models: evaluation of a multi-resolution method for improving image search. In *Proceedings of the 5th British machine vision conference (BMVC'94)*, pp. 327–336. Surrey, UK: BMVA Press.
28. Cootes T, Edwards G, Taylor C. 1998. Active appearance models. In *Proceedings of the fifth European conference on computer vision (ECCV 1998)*, Volume 2. *Lecture notes in computer science*, Vol. 1407, pp. 484–498. Ed H Burkhardt, B Neumann. Berlin: Springer.
29. Sclaroff S, Isidoro J. 1998. Active blobs. In *Proceedings of the sixth international conference on computer vision (ICCV)*, pp. 1146–1153. Washington, DC: IEEE Computer Society.
30. Hill A, Thornham A, Taylor C. 1993. Model-based interpretation of 3d medical images. In *Proceedings of the fourth British machine vision conference (BMVC'93)*, pp. 339–348. Surrey, UK: BMVA Press.
31. Lanitis A, Taylor A, Cootes T. 1994. Automatic tracking, coding and reconstruction of human faces using flexible appearance models. *Electron Lett* **30**(19):1578–1579.
32. Baumberg A, Hogg D. 1994. An efficient method for contour tracking using active shape models. In *Proceedings of the 1994 IEEE workshop on motion of non-rigid and articulated objects*, pp. 194–199. Washington, DC: IEEE Computer Society.
33. Cootes T, Taylor C. 1995. Combining point distribution models with shape models based on finite element analysis. *Image Vision Comput* **13**(5):403–409.
34. Lanitis A, Taylor C, Cootes T. 1994. Recognising human faces using shape and grey-level information. In *Proceedings of the third international conference on automation, robotics and computer vision*, pp. 1153–1157. Washington, DC: IEEE Computer Society.
35. Bookstein F. 1989. Principal warps: thin-plate splines and the decomposition of deformations. *IEEE Trans Pattern Anal Machine Intell* **11**(6):567–585.
36. Hunter IA, Soraghan JJ, Christie J, Durrani JS. 1993. Detection of echocardiographic left ventricle boundaries using neural networks. *IEEE Proc Comput Cardiol* **20**:201–204.
37. Parker A, Hill A, Taylor C, Cootes T, Jin X, Gibson D. 1994. Application of point distribution models to the automated analysis of echocardiograms. *IEEE Proc Comput Cardiol* **21**:25–28.
38. Taine M, Herment A, Diebold B, Peronneau P. 1994. Segmentation of cardiac and vascular ultrasound images with extension to border kinetics. *Proceedings of the 8th IEEE symposium on ultrasonics*, Vol. 3, pp. 1773–1776. Washington, DC: IEEE Computer Society.
39. Papadopoulos I, Strintzis MG. 1995. Bayesian contour estimation of the left ventricle in ultrasound images of the heart. *Proceedings of the IEEE conference on engineering in medicine and biology*, Vol. 1, pp. 591–592. Washington, DC: IEEE Computer Society.
40. Malassiotis S, Strintzis MG. 1999. Tracking the left ventricle in echocardiographic images by learning heart dynamics. *IEEE Trans Med Imaging* **18**(3):282–290.
41. Amini A, Weymouth T, Jain R. 1990. Using dynamic programming for solving variational problems in vision. *IEEE Trans Pattern Anal Machine Intell* **12**(9):855–867.
42. Cohen L. 1991. On active contour models and balloons. *Comput Vision Graphics Image Process: Image Understand* **53**(2):211–218.
43. Grzeszczuk R, Levin D. 1997. Brownian strings: segmenting images with stochastically deformable contours. *IEEE Trans Pattern Anal Machine Intell* **19**(10):1100–1114.
44. Herlin I, Nguyen C, Graffigne C. 1992. A deformable region model using stochastic processes applied to echocardiographic images. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR 1992)*, pp. 534–539. Washington, DC: IEEE Computer Society.
45. Lobregt S, Viergever M. 1995. A discrete dynamic contour model. *IEEE Trans Med Imaging* **14**(1):12–24.
46. Cootes T, Taylor C, Cooper D, Graham J. 1995. Active shape models: their training and application. *Comput Vision Image Understand* **61**(1):38–59.
47. Staib L, Duncan J. 1992. Boundary finding with parametrically deformable models. *IEEE Trans Pattern Anal Machine Intell* **14**(11):1061–1075.

48. Leymarie F, Levine M. 1993. Tracking deformable objects in the plane using an active contour model. *IEEE Trans Pattern Anal Machine Intell* **15**(6):617–634.
49. Niessen W, Duncan J, Viergever M, Romeny B. 1995. Spatiotemporal analysis of left-ventricular motion. *Proc SPIE* **2434**:250–261.
50. Signh A, Von Kurowski L, Chiu M. 1993. Cardiac MR image segmentation using deformable models. In *SPIE proceedings on biomedical image processing and biomedical visualization*, Vol. 1905, pp. 8–28. Bellingham, WA: SPIE.
51. Stark K, Fuchs S. 1996. A method for tracking the pose of known 3D objects based on an active contour model. *Proceedings of the international conference on pattern recognition (ICPR'96)*, pp. 905–909. Washington, DC: IEEE Computer Society.
52. Terzopoulos D, Szeliski R. 1992. Tracking with Kalman snakes. In *Active vision*, pp. 3–20. Ed A Blake, A Yuille. Cambridge: MIT Press.
53. Lelieveldt B, Mitchell S, Bosch J, van der Geest R, Sonka M, Reiber J. 2001. Time-continuous segmentation of cardiac image sequences using active appearance motion models. *Proceedings of the 17th international conference on information processing in medical imaging (ICIPMI'01). Lecture Notes in computer science*, Vol. 2082, pp. 446–452. Berlin: Springer.
54. Black M, Yacoob Y. 1997. Recognizing facial expressions in image sequences using local parametrized models of image motion. *Int J Comput Vision* **25**(1):23–48.
55. Bonciu C, Léger C, Thiel J. 1998. A Fourier-Shannon approach to closed contour modeling. *Bioimaging* **6**:111–125.
56. Cootes T, Taylor C. 1997. A mixture model for representing shape variation. In *Proceedings of the eighth British machine vision conference (BMVC'97)*, pp. 110–119. Surrey, UK: BMVA Press.
57. Hill A, Taylor CJ. 1992. Model-based image interpretation using genetic algorithms. *Image Vision Comput* **10**(5):295–300.
58. Metaxas D, Terzopoulos D. 1991. Constrained deformable superquadrics and nonrigid motion tracking. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR 1991)*, pp. 337–343. Washington, DC: IEEE Computer Society.
59. Singh A, Goldgof D, Terzopoulos D. 1998. *Deformable models in medical image analysis*. Washington, DC: IEEE Computer Society.
60. Székely G, Kelemen A, Brechbühler C, Gerig G. 1996. Segmentation of 3D objects from MRI volume data using constrained elastic deformations of flexible Fourier surface models. *Med Image Anal* **1**(1):19–34.
61. Hamarneh G, Chodorowski A, Gustavsson T. 2000. Active contour models: application to oral lesion detection in color images. *Proceedings of the IEEE international conference on systems, man, and cybernetics*, Vol. 4, pp. 2458–2463. Washington, DC: IEEE Computer Society.
62. Hamarneh G, Gustavsson T. 2000. Statistically constrained snake deformations. In *Proceedings of the IEEE international conference on systems, man, and cybernetics*, Vol. 3, pp. 1610–1615. Washington, DC: IEEE Computer Society.
63. Hamarneh G, Gustavsson T. 2004. Deformable spatiotemporal shape models: extending ASM to 2D + time. *J Image Vision Comput* **22**(6):461–470.

12

DEFORMABLE ORGANISMS FOR MEDICAL IMAGE ANALYSIS

Ghassan Hamarneh and Chris McIntosh

School of Computing Science, Simon Fraser University
Burnaby, British Columbia, Canada

In medical image analysis strategies based on deformable models, controlling the deformations of models is a desirable goal in order to produce proper segmentations. In Chapter 11—"Physically and Statistically Based Deformable Models for Medical Image Analysis"—a number of extension were demonstrated to achieve this, including user interaction, global-to-local deformations, shape statistics, setting low-level parameters, and incorporating new forces or energy terms. However, incorporating expert knowledge to automatically guide deformations can not be easily and elegantly achieved using the classical deformable model low-level energy-based fitting mechanisms. In this chapter we review Deformable Organisms, a decision-making framework for medical image analysis that complements bottom–up, data-driven deformable models with top–down, knowledge-driven mode-fitting strategies in a layered fashion inspired by artificial life modeling concepts. Intuitive and controlled geometrically and physically based deformations are carried out through behaviors. Sensory input from image data and contextual knowledge about the analysis problem govern these different behaviors. Different deformable organisms for segmentation and labeling of various anatomical structures from medical images are also presented in this chapter.

1. INTRODUCTION AND MOTIVATION

Due to the important role of medical imaging in the understanding, diagnosis, and treatment of disease, potentially overwhelming amounts of medical image data

Address all correspondence to: Dr. Ghassan Hamarneh, School of Computing Science, Simon Fraser University, 8888 University Drive, Burnaby, BC, V5A 1S6, Canada. Phone: +1.604.291.3007, Fax: +1.604.291.3045, hamarneh@cs.sfu.ca.

are continuously being acquired. This is creating an increasing demand for medical image analysis (MIA) tools that are not only robust and highly automated, but also intuitive for the user and flexible to adapt to different applications. Medical image segmentation in particular remains one of the key tasks indispensable to a wide array of subsequent quantification and visualization goals in medicine, including computer-aided diagnosis and statistical shape analysis applications. However, the automatic segmentation and labeling of anatomical structures in medical images is a persistent problem that continues to defy solution. Several classifications of segmentation techniques exist, including edge, pixel, and region-based techniques, clustering, graph theoretic, and model correlation approaches [1–5]. However, no one method can yet handle the most general case with sufficient accuracy.

It is important to note that a substantial amount of knowledge is often available about anatomical structures of interest — shape, position, orientation, symmetry, relationship to neighboring structures, landmarks, etc. — as well as about the associated image intensity characteristics. However, medical image analysis researchers have struggled to develop segmentation techniques that can take full advantage of such knowledge.

The development of general-purpose automatic segmentation algorithms will require not only powerful bottom–up, data-driven processes, but also equally powerful top–down, knowledge-driven processes within a robust decision-making framework that operates across multiple levels of abstraction. A flexible framework is needed that can operate at the appropriate level of abstraction and is capable of incorporating and applying all available knowledge effectively (i.e., at the correct time and location during image analysis). A critical element in any such viable highly automated solution is the decision-making framework itself. Top–down, hierarchically organized models that shift their focus from structures associated with stable image features to those associated with less stable features are promising examples [6, 7]. Although Knowledge-based [8–14] and agent-based segmentation techniques [15–19] have been proposed in the past, their use of high-level contextual knowledge remains largely ineffective because it is intertwined much too closely with the low-level optimization-based mechanisms. We revisit ideas for incorporating knowledge that were explored in earlier systems and develop a new frameworks that focuses on top–down reasoning strategies that may best leverage the powerful bottom–up feature detection and integration abilities of deformable models and other modern model-based medical image analysis techniques.

Deformable models, one of the most actively researched model-based segmentation techniques [20], feature a potent bottom–up component founded in estimation theory, optimization, and physics-based dynamical systems, but their top–down processes have traditionally relied on interactive initialization and guidance by knowledgeable users (see Section 1.4, Snakes Drawbacks, in Chapter 11). Since their introduction by Terzopoulos et al. [83, 84], deformable models for medical image segmentation have gained increasing popularity. In addition

to physics-based explicit deformable models [20, 21], geometry-based implicit implementations have also attracted attention [22–24].

Although most deformable shape models are capable of deforming into a variety of shapes, they lack the ability to undergo intuitive deformations such as bending, stretching, and other global motions such as sliding and backing up. The reason is that at the geometric level deformable models are typically boundary-based and are not designed with intuitive, multiscale, multi-location deformation controllers or deformation handles. Their inability to perform controlled and intuitive deformations makes it difficult to develop reasoning and planning knowledge-driven strategies for model-to-data fitting at the appropriate level of abstraction.

In more sophisticated deformable models, prior information in the form of measured statistical variation is used to constrain model shape and appearance [25–27]. However, these models have no explicit awareness of where they or their neighbors are, and the effectiveness of these constraints is consequently dependent upon model initialization conditions. The lack of awareness also prevents the models from taking proper advantage of neighborhood information via model interaction and prevents them from knowing when to trust the image feature information and ignore the constraint information and vice versa. The constraint information is therefore applied arbitrarily. Furthermore, because there is no active, explicit search for stable image features, the models are prone to latching onto incorrect features [26], simply due to their myopic decision-making abilities and the proximity of spurious features. Once this latching occurs, the lack of explicit control of the fitting procedure prevents the model from correcting such missteps. The result is that the local decisions that are made do not add up to intelligent global behavior.

For example, when segmenting the corpus callosum (CC) in 2D midsagittal images[1], the vocabulary that one uses should preferably contain words that describe principal anatomical features of the CC, such as the genu, splenium, rostrum, fornix, and body (Figure 1), rather than pixels and edges. The deformable model should match this natural descriptiveness by grouping intuitive model parameters at different scales and locations within it, rather than providing localized boundary-based parameters only.

Attempts to fully automate deformable model segmentation methods have so far been less than successful at coping with the enormous variation in anatomical structures of interest, the significant variability of image data, the need for intelligent initialization conditions, etc. It is difficult to obtain intelligent, global (i.e., over the whole image) model behavior throughout the segmentation process from fundamentally local decisions. In essence, current deformable models have no explicit awareness of where they are in the image, how their parts are arranged, or what they or any neighboring deformable models are seeking at any time during the optimization process.

An explicit search for anatomical landmarks requires powerful, flexible, and intuitive model deformation control coupled with appropriate feature detection

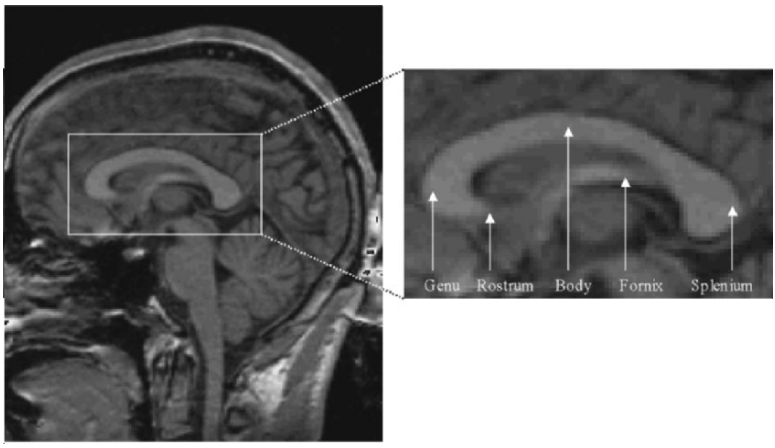


Figure 1. Corpus callosum anatomy overlain on a midsagittal MRI brain slice.

capabilities. Consequently, controlling the deformations of an object's shape in a way that is based on the natural geometry of the object is highly desirable in medical image segmentation and interpretation. This intuitive deformation ability reflects the flexibility of clay to be shaped in a sculptor's hands and naturally lends itself to guidance by high-level controllers. Furthermore, the performance of the controllers can be greatly enhanced by keeping the deformations consistent with prior knowledge about the possible object shape variations. Several techniques exist that provide some control of model deformations by incorporating new energy terms [29, 30] or by allowing only feasible deformations to be produced through the incorporation of prior shape knowledge [25, 31–34]. However, the deformation control these earlier methods provide is tightly linked to low-level cost or energy terms, in which high-level anatomical knowledge is difficult to encode.

We facilitate the merger of higher-level control mechanisms with powerful controlled deformations through a layered architecture, where the high-level reasoning layer has knowledge about and control over the low-level model (or models) at all times. The reasoning layer should apply an active, explicit search strategy that first looks for the most stable image features before proceeding to less stable image features, and so on. It should utilize contextual knowledge to resolve ambiguity in regions where there is a deficiency of image feature information. By constructing the framework in a layered fashion, we are able to separate and complement the data-driven, local image feature integration functionality with the knowledge-driven, model-fitting control functionality, exploiting both for maximal effectiveness. This separation allows us to construct a model-fitting controller from an extensible set of standardized subroutines. The subroutines are defined in terms of deformation maneuvers and high-level anatomical landmark detec-

tors rather than low-level image features. Different subroutines are invoked based on high-level model-fitting decisions by integrating image features, prior contextual anatomical knowledge, and a pre-stored segmentation plan. Furthermore, by combining a layered architecture with a set of standard subroutines, powerful and flexible "custom-tailored" models can be rapidly constructed, thus providing general-purpose tools for automated medical image segmentation and labeling.

2. DEFORMABLE ORGANISMS: AN ARTIFICIAL LIFE MODELING PARADIGM FOR MEDICAL IMAGE ANALYSIS

To realize the ideas and achieve the goals mentioned above in Section 1, we introduced a new paradigm for automatic medical image analysis that adopts concepts from the emerging field of artificial life $(AL)^2$. In particular, we developed *deformable organisms*, autonomous agents whose objective is the segmentation and analysis of anatomical structures in medical images [38]. The AL modeling-based approach provides us with the required flexibility to adhere to an active, explicit search strategy that takes advantage of contextual and prior knowledge of anatomy. The organisms are aware of the progress of the segmentation process and of each other, allowing them to effectively and selectively apply knowledge of the target objects throughout their development.

Viewed in the context of the AL modeling hierarchy (Figure 2a), current *automatic* deformable model-based approaches to medical image analysis utilize geometric and physical modeling layers only (Figure 2b). In *interactive* deformable models, such as snakes, the human operator is relied upon to provide suitable behavioral and cognitive level support (Figure 2c). At the physical level, deformable models interpret image data by simulating dynamics or minimizing energy terms, but the models themselves do not monitor or control this optimization process except in a most primitive way.

To overcome the aforementioned deficiencies while retaining the core strengths of the deformable model approach, we add high-level controller layers (a "brain") on top of the geometric and physical layers to produce an autonomous deformable organism (Figure 2d). The planned activation of these lower deformation layers allows us to control the fitting or optimization procedure. The layered architecture approach allows the deformable organism to make deformation decisions at the correct level of abstraction utilizing prior knowledge, memorized information, sensed image features, and inter-organism interaction.

Specifically, a deformable organism is structured as a "muscle"-actuated "body" whose "behaviors" are controlled by a "brain" (Figure 3) that makes decisions based on perceived image data and extraneous knowledge. The brain is the organism's cognitive layer, which activates behavior routines (e.g., for a CC organism: find-splenium, find-genu, find-upper-boundary, etc. (Figure 1)) according to a plan or schedule (Figure 3).

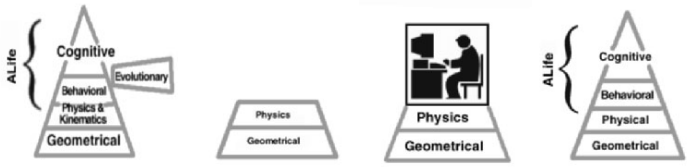


Figure 2. AL, Deformable Models, and Deformable Organisms. (a) AL modeling pyramid. Adapted with permission from [39]. Copyright ©1999, ACM. (b) Automatic deformable models (incorporating geometry and physics layers only). (c) Deformable models guided by an expert human operator. (d) Intelligent deformable models (deformable organisms) provide a model of the cognitive abilities of human operators (by including higher cognitive layers).

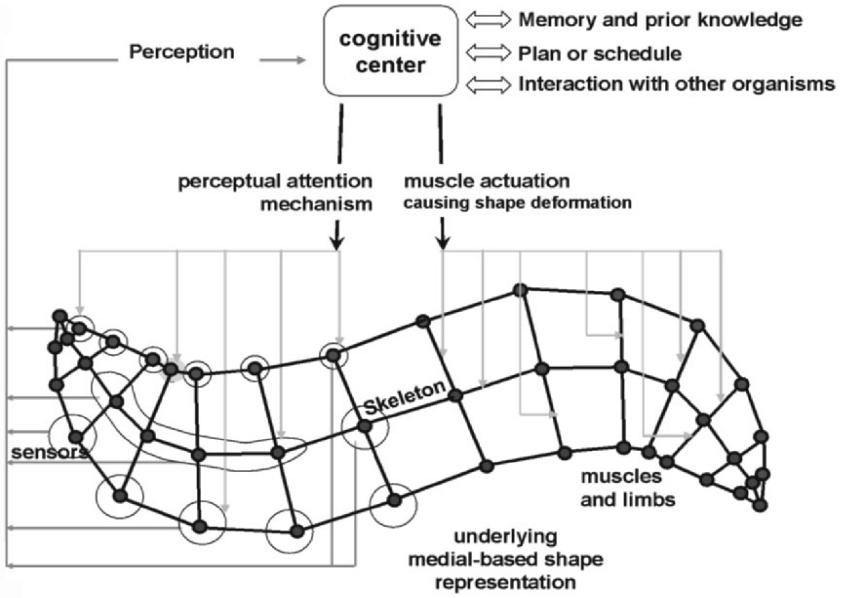


Figure 3. A Deformable Organism. The brain issues "muscle" actuation and perceptual attention commands. The organism deforms and senses image features, whose characteristics are conveyed to the brain. The brain makes decisions based on sensory input, memorized information and prior knowledge, and a pre-stored plan, which may involve interaction with other organisms. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

Carrying out a behavior routine requires image information in order for the proper shape deformation to take place toward achieving the goal of the current behavior. The deformable organism perception system is responsible for gathering image information and comprises a set of sensors that are adaptively tuned to specific image features (edge strength, texture, color, etc) in a task-specific way. Hence, the organism can disregard sensory information superfluous to its current behavioral needs.

The organism carries out a sequence of active, explicit searches for stable anatomical features, beginning with the most stable anatomical feature and then proceeding to the next best feature. This allows the organism to be "self-aware" (i.e., knows where it and its parts are and what it is seeking at every stage) and is therefore able to perform these searches intelligently and effectively by utilizing a conflux of contextual knowledge, perceived sensory data, an internal mental state, memorized knowledge, and a cognitive plan. For example, it need not be satisfied with the nearest matching feature, but can look further within a region to find the best match, thereby avoiding globally suboptimal solutions. The plan (or plans) can be generated with the aid of a human expert, since the behavior routines are defined using familiar anatomical terminology.

An organism may "interact" with other organisms to determine optimal initial conditions. Once stable features are found and labeled, an organism can selectively use prior knowledge or information from the neighbor organisms to determine the object boundary in regions known to offer little or no feature information. Interaction among organisms may be as simple as collision detection and avoidance, or one or several organisms supplying intelligent initial conditions to another, or the use of inter-organism statistical shape/image appearance constraint information.

Furthermore, by carrying out explicit searches for features, correct correspondences between the organism and the data are more readily assured. If a feature cannot be found, an organism may "flag" this situation (Figure 4b). If multiple plans exist, another plan can be selected and/or the search for the missing feature postponed until further information is available (e.g., from a neighboring organism). Alternatively, the organism can retrace its steps and return to a known state and then inform the user of the failure. A human expert can intervene and put the organism back on course by manually identifying the feature. This strategy is possible because of the sequential and spatially localized nature of the model fitting process.

Customized behavioral routines and explicit feature search requires powerful, flexible and intuitive model deformation control. The behavior routines activate "motor" (i.e., deformation) controller routines or growth controller routines, enabling the organism to fulfill its goal of object segmentation. An organism may begin in an "embryonic" state with a simple proto-shape, and then undergo controlled growth as it develops into an "adult," proceeding from one stable object feature to the next. Alternatively, an organism may begin in a fully developed

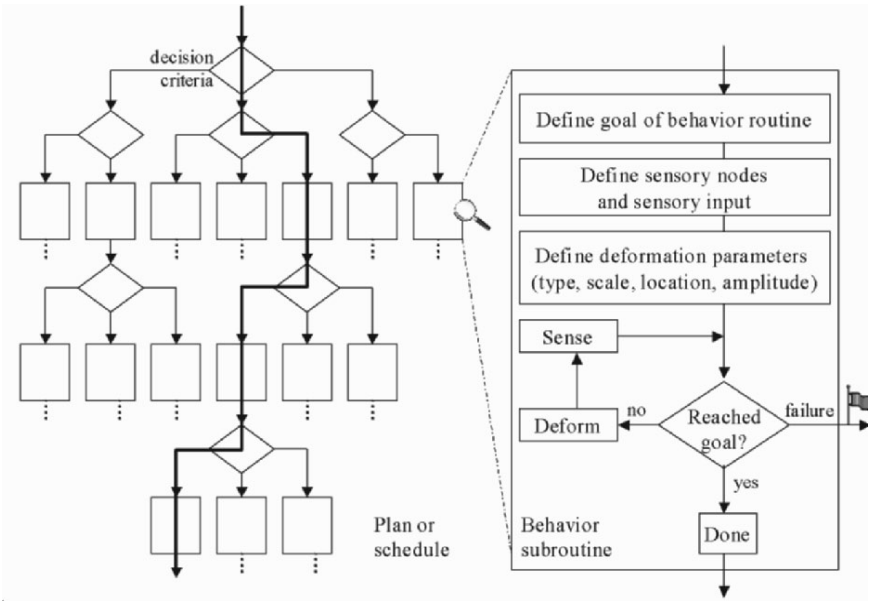


Figure 4. (a) Procedural representation of a fragment of a deformable organism's plan or schedule. The organism goes through several behavior subroutines (bold path in (a)). (b) Generic example of a behavior routine. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

state and undergo controlled deformations as it carries out its model-fitting plan. The type of organism to use, or whether to use, some sort of hybrid organism, is dependent on the image and shape characteristics of the target anatomical structure (different examples are presented in Section 4).

Deformation controllers are parametrized procedures dedicated to carrying out a complex deformation function, such as successively bending a portion of the organism over some range of angle or stretching part of the organism forward some distance. They translate natural control parameters such as <bend_angle, location, scale> or <stretch_length, location, scale> into detailed deformations.

To summarize, the AL layered architecture provides the needed framework to complement the geometrical and physical layers of classical deformable organisms (the model shape, topology, and deformations) with behavioral and cognitive layers (the high-level controllers that activate the routines) utilizing a perception system (the source of the image data) and contextual-knowledge (knowledge about the anatomy's shape, appearance, neighborhood relationships, etc.).

3. THE LAYERED ARCHITECTURE OF DEFORMABLE ORGANISMS

This section describes the layers of the deformable organism architecture, starting with the lowest layers (geometrical and physical), and working up to the higher layers (behavioral, cognitive, and sensory). Throughout these sections we will provide examples of each layer to solidify the abstract terminology. To this end, we will draw upon particular applications of deformable organisms, each of which is briefly outlined below.

- **Geometrically based deformable CC organism**: This CC organism relies on pure geometry-based shape deformations and was designed to begin as a small "worm," and then grow and expand itself to fit the data (Figure 5a). Details of its deformation module are presented in Section 3.3.1, sensors in Section 3.4, and with behaviors and results in Section 4.1.

- **Physically based deformable CC organism:** Incorporates a true physically based deformation layer (Section 3.3.3) to allow for increased robustness and intuitive interaction (Figure 5b). Example physics, behavioral, and cognitive layers of this organism are presented in Sections 3.3.3, 3.5.1, and 3.6.1, respectively, with results in Section 4.4.

- **2D vessel crawler:** The 2D vessel crawler is a worm-like deformable organism (Figure 5c) that grows to segment vessels in 2D angiograms by continuously sensing which direction to grow in, then expanding and fitting to the walls. Its distinguishing feature is its ability to discriminate between bifurcations and overlaps (Figure 41) using an off-board sensor (Section 3.4).

- **3D vessel crawler:** This organism extends the 2D vessel crawler to 3D geometry, incorporates a true physical deformation layer (Section 3.3.4), and is equipped with new behavioral and sensory modules (Section 3.4) [86]. It crawls along vasculature in 3D images (Figure 5d), accurately segmenting vessel boundaries, detecting and exploring bifurcations, and providing sophisticated, clinically relevant structural analysis. Results are presented in Section 4.6.

3.1. The Geometrical Layer (Shape and Topology)

The geometrical layer of the deformable organism houses the shape representation and morphological and topological constraints of the organism. The deformable organisms we developed to date are based on medial-axis or medial-sheet deformable shape representations. The medial-based shape representation allows for a variety of intuitive and controlled changes in the model's shape that are described relative to the natural geometry of the object rather than as displacements

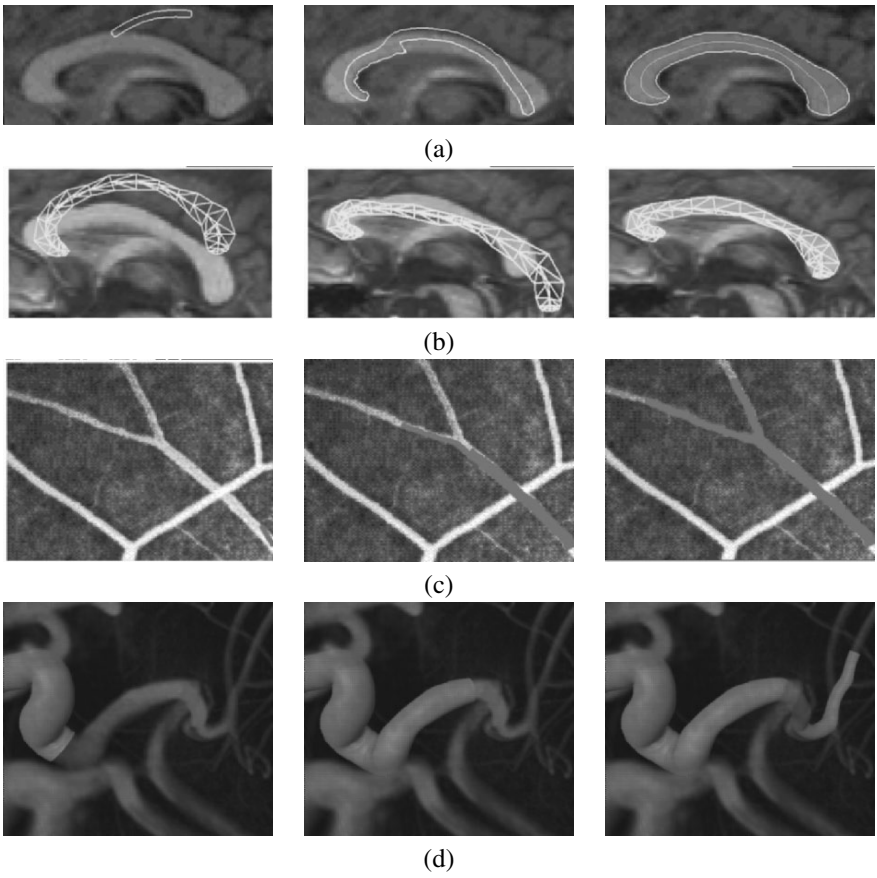


Figure 5. Example deformable organisms for medical image analysis. (a) Geometrically based and (b) physically based deformable CC organisms. (c) 2D and (d) 3D vessel crawler. Progress of segmentation is shown from left to right. See attached CD for color version.

to its boundary. These changes in shape include stretching, thickening, or bending and are realized as either pure geometric deformations or a result of deformations simulated at the physics layer of the organism. In Section 3.3 we present examples of different shape representations that allow such controlled deformations.

3.2. The Physical Layer (Motor System and Deformations)

The physical layer is responsible for deforming the shape of the organism and changing its geometry. Deformations carried out by the physical layer of the organism are divided into two classes: simple deformation capabilities and

more complex locomotion maneuvers. The simple deformation capabilities constitute the low-level motor skills or the basic shape deformation actuators of the organism. Simple deformations include global deformations such as affine transformation (translate, rotate, scale) or localized deformations such as a simple bulge or a stretch. The complex locomotion maneuvers are the high-level motor skills of the organism constructed from several basic motor skills. These are parametrized procedures that carry out complex deformation functions such as sweeping over a range of rigid transformation parameters, sweeping over a range of stretch/bend/thickness parameters, bending at increasing scales, moving a bulge on the boundary, etc. Other high-level deformation capabilities include smoothing the medial or the boundary of the model, or moving the medial axis or medial sheet to a position exactly midway between the boundaries on both sides of the medial. In the following section (3.3) we present different examples of deformation enabled by a number of medial-based geometrical representations.

3.3. Controlling Shape Deformation

Top–down, knowledge-driven model fitting strategies require underlying shape representations responsive to high-level controlled shape deformation commands. As the organism's cognitive layer decides on a behavior, the behavior routine will involve carrying out shape deformations realized through high-level motor skills, which in turn are realized through simpler low-level motor skills. The goal here is the ability to intelligently control the different types and extent of model deformations during the model-to-data fitting process in an effort to focus on the extraction of stable image features before proceeding to object regions with less well-defined features.

The choice of shape representation is undeniably crucial for segmentation, recognition, and interpretation of medical images. The study of shape is unsurprisingly attracting a great deal of attention within the medical image analysis community [40–42]. A desirable trait in a shape representation is its ability to control non-rigid object deformations at multiple locations and scales in an interactive and intuitive manner. Deformable shape models [20], in particular, are mostly boundary based, and therefore multiscale deformation control is constructed upon arbitrary boundary point sets and not upon object-relative geometry. Although they provide excellent local shape control, they lack the ability to undergo intuitive global deformation and decompose shape variability into intuitive deformations. As a result, it is difficult to incorporate intelligent deformation control operating at the right level of abstraction into the typical deformable model framework of energy minimization. Consequently, these models remain sensitive to initial conditions and spurious image features in image interpretation tasks.

Hierarchical boundary-based shape models [43–46] and volume-based shape representations or free-form deformation mechanisms have been proposed [47–51]; however, as their deformation is not based on object-relative geometry, they are

limited either by the type of objects they can model, or by the type and intuitiveness of the deformations they can carry out. They are also typically not defined in terms of the object but rather the object is unnaturally defined (or deformed) in terms of the representation or deformation mechanism. Other 3D shape representations with similar drawbacks include spherical harmonics, FEM, NURBS, and wavelet-based representations [52–55].

Shape models founded upon the use of the medial-axis transform [56] are emerging as a powerful alternative to the earlier boundary-based and volume-based techniques [57–68]. Medial representations provide both a local and global description of shape. Deformations defined in terms of a medial axis are natural and intuitive and can be limited to a particular scale and location along the axis, while inherently handling smoothness and continuity constraints.

Statistical models of shape variability have been used for medical image interpretation [25, 27, 31, 32, 69]. These typically rely on principal component analysis (PCA) and hence are only capable of capturing global shape variation modes. Statistical analysis of medial-based shape representation has been the focus of much recent research [70–74].

In the following subsections we present the details of a variety of shape representation and controlled deformation techniques that are crucial to the operation of deformable organisms and modeling their lower geometrical and physical layers. These include 2D medial profiles (Section 3.3.1), 3D shape medial patches (3.3.2), 2D spring-mass systems with physics-based deformations (3.3.3), and their 3D extension (3.3.4).

3.3.1. 2D Shape Representation and Deformation with Medial Profiles

We describe a multiscale, medial-based approach to shape representation and controlled deformation in an effort to meet the requirements outlined above in the previous sections. In this shape representation and deformation scheme, an anatomical structure (e.g., the CC) is described with four shape profiles derived from the primary medial axis of an organism's boundary contour (e.g., CC boundary contour). The medial profiles describe the geometry of the structure in a natural way and provide general, intuitive, and independent shape measures. These profiles are: length, orientation, left (with respect to the medial axis) thickness, and right thickness. Once the profiles are constructed, various deformation functions or *operators* can be applied at certain locations and scales on a profile, producing intuitive, controlled deformations: stretching, bending, and bulging. In addition to the general deformation operators, we wish to use as much knowledge as possible about the object itself and to generate statistically proven feasible deformations from a training set. We would like to control these statistical deformations locally along the medial shape profiles to support our goal of intelligent deformation scheduling. Since general statistically derived shape models only produce global shape variation modes [25, 27], we present spatially localized feasible deforma-

tions at desired scales by utilizing hierarchical (multiscale) and regional (multi-location) principal component analysis to capture shape variation statistics.

In the following sections, we demonstrate the ability to produce controlled shape deformations by applying them to 2D medial-based representations of the CC, derived from 2D midsagittal MRI slices of the brain. We begin by describing the generation and use of medial-based profiles for shape representation and describe a set of general operators that act on the medial shape profiles to produce controlled shape deformations. 3.3.1). We then present a technique for performing a multiscale multi-location statistical analysis of the shape profiles and describe statistics-based deformations based on this analysis. We present a simple application of the controlled shape deformations and demonstrate their use in an automatic medical image analysis system (Section 4.1).

Medial Profiles for Shape Representation We use a boundary representation of an object to generate the medial-based profiles. Generation of the profiles begins with extraction of a sampled pruned skeleton of the object to obtain a set of medial nodes. Four medial profiles are constructed: a length profile $L(m)$, an orientation profile $O(m)$, a left (with respect to the medial axis) thickness profile $T^l(m)$, and a right thickness profile $T^r(m)$, where $m = 1, 2, \ldots, N$, N being the number of medial nodes, with nodes 1 and N being the terminal nodes. The length profile represents the distances between consecutive pairs of medial nodes, and the orientation profile represents the angles of the between edges connecting consecutive pairs of medial nodes. The thickness profiles represent the distances between medial nodes and their corresponding boundary points on both sides of the medial axis (Figure 6). Corresponding boundary points are calculated by computing the intersection of a line passing through each medial node in a direction normal to the medial axis, with the boundary representation of the object. Example medial profiles are shown in Figure 7.

Shape Reconstruction from Medial Profiles To reconstruct the object's shape given its set of medial profiles, we calculate the positions of the medial and boundary nodes by following these steps:

1. Specify affine transformation parameters: orientation angle θ, translation values (t_x, t_y), and scale (s_x, s_y).

2. Using medial node 1 as the base or reference node, place it at location $x_1 = (t_x, t_y)$.

3. Repeat steps 4 and 5 for $m = 1, 2, \ldots, N$.

4. Compute locations x_m^l and x_m^r of boundary points l and r at either side of the mth medial node (Figure 6) as

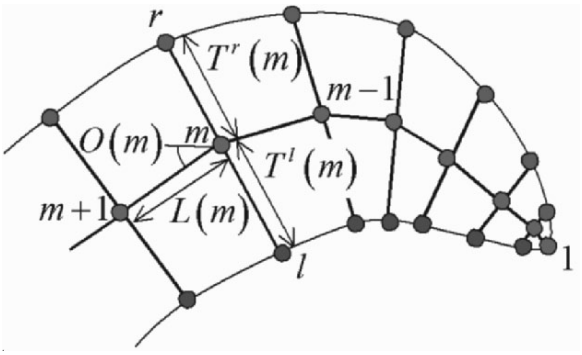


Figure 6. Diagram of shape representation. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

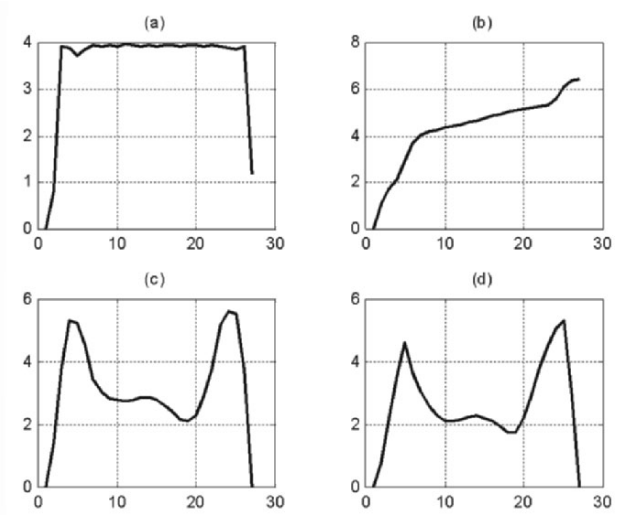


Figure 7. Example medial shape profiles: (a) length profile $L(m)$, (b) orientation profile $O(m)$, (c) left thickness profile $T^l(m)$, and (d) right thickness profile $T^r(m)$. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

$$x_m^l = x_m + T^l(m) \begin{pmatrix} s_x \cos\left(\theta + O(m) + \frac{\pi}{2}\right) \\ s_y \sin\left(\theta + O(m) + \frac{\pi}{2}\right) \end{pmatrix} \tag{1}$$

and, similarly,

$$x_m^r = x_m + T^r(m) \begin{pmatrix} s_x \cos\left(\theta + O(m) - \frac{\pi}{2}\right) \\ s_y \sin\left(\theta + O(m) - \frac{\pi}{2}\right) \end{pmatrix}. \tag{2}$$

5. If $m < N$, compute location x_{m+1} of the next medial node $m+1$ as

$$x_{m+1} = x_m + L(m) \begin{pmatrix} s_x \cos(\theta + O(m)) \\ s_y \sin(\theta + O(m)) \end{pmatrix}. \tag{3}$$

An example shape reconstruction is shown in Figure 8.

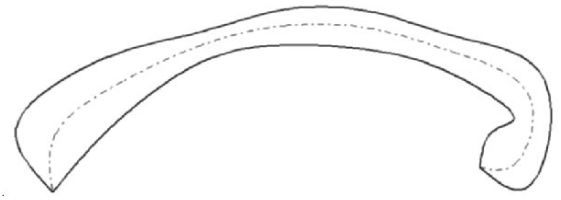


Figure 8. Object reconstruction resulting from the shape profiles in Figure 7.

Shape Deformations Using Medial-Based Operators Once the shape profiles have been generated, we can construct deformation operators and apply these operators to the shape profiles. This results in intuitive deformations of the object upon reconstruction. That is, by applying an operator to the length, orientation, or thickness shape profile, we obtain a stretch, bend, or bulge deformation, respectively. Each deformation operator is implemented by defining a medial-based operator profile, $k(m)$, of a particular type (Figure 9) and specifying an amplitude, location, and scale.

The operator profile is then added to (or blended with) the medial shape profile corresponding to the desired deformation. For example, to introduce a bulge on the right boundary, an operator profile with a specific amplitude, type, location, and scale is generated and added to the right thickness medial profile $T^r(m)$ to obtain $T^r(m) + k(m)$ (Figure 10).

In general the application of a deformation operator $k(m)$ alters the desired shape profile according to

$$p_d(m) = \bar{p}_d(m) + \alpha_{dlst} k_{dlst}(m), \tag{4}$$

where p is the shape profile, d is the deformation type (stretch, bend, left/right bulge), i.e., $p_d(m) : \left\{L(m), O(m), T^l(m), T^r(m)\right\}$, $\bar{p}$ is the average shape

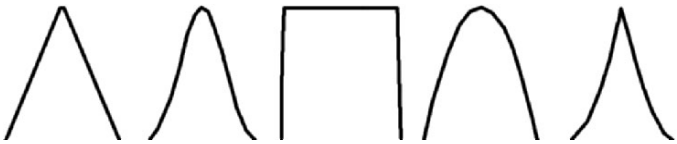


Figure 9. Examples of operator types: (left to right) triangular, Gaussian, flat, bell, and cusp [75]. Reprinted with permission from [63]. Copyright ©2004, World Scientific.

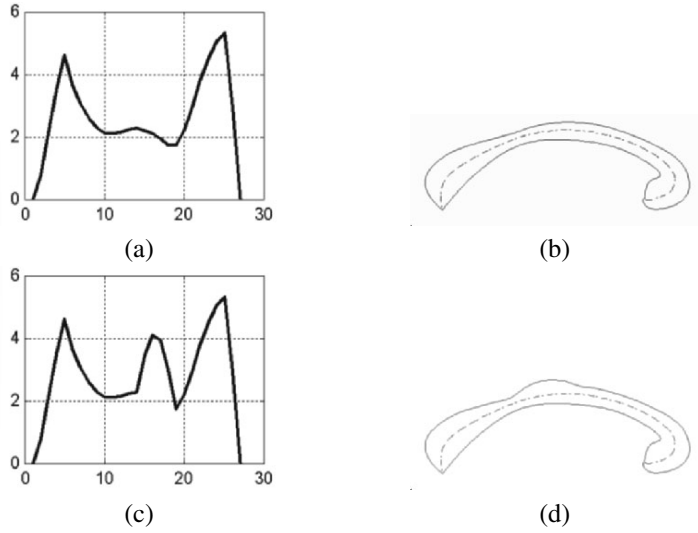


Figure 10. Introducing a bulge on the right boundary by applying a deformation operator on the right thickness profile: (a) $T^r\,(m)$ before and (c) after applying the operator. (b) Reconstructed shape before and (d) after the operator. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

profile, k is the operator profile (with unity amplitude), l is the location, s is the scale, t is the operator type (Gaussian, triangular, ..., etc.), and α is operator amplitude.

Altering one shape profile only affects the shape property associated with that profile and does not affect any other object shape properties. For example, applying an operator to the orientation profile results in a bend deformation only and does not result in a stretch or bulge. This implies the ability to perform successive operator-based object deformations of varying amplitudes, types, locations or scales, which can be expressed as

$$p_d\,(m) = \bar{p}_d\,(m) + \sum_l \sum_s \sum_t \alpha_{dlst} k_{dlst}\,(m). \tag{5}$$

Examples of operator-based deformations are shown in Figure 11a–d.

Statistical Shape Analysis by Hierarchical Regional PCA In many applications, prior knowledge about object shape variability is available or can be obtained by studying a training set of shape examples. The training set is typically created by labeling corresponding landmark points in each shape example. Principal Component Analysis (PCA) is then applied to the training set, resulting in a point distribution model (PDM) [25]. The PDM describes the main modes of variation of the landmark positions and the amount of variation each mode explains. A drawback of this original approach is that the result of varying the weight of a single variation mode generally causes all the landmark positions to change. In other words, although the original PDM model produces only feasible shape deformations, a desirable trait, it generally produces global deformations over the entire object. Our goal is to utilize prior knowledge and produce feasible deformations, while also controlling the scale and location of these deformations. Toward this end, we perform a multiscale (hierarchical) multi-location (regional) PCA on a training set of medial shape profiles. To achieve this, we collect spatially corresponding subprofiles from the shape profiles. The length of a subprofile reflects the scale over which the analysis is performed. The principal component analysis is now a function of the location, scale, and type of shape profile (length, orientation, or thickness). Thus, for each location, scale, and shape profile type, we obtain an average medial subprofile, the main modes of variation, and the amount of variation each mode explains. The result is that we can now generate a feasible stretch, bend, or bulge deformation at a specific location and scale.

A shape profile can now be written as the sum of the average profile and the weighted modes of variation as follows:

$$p_d\left(m\right) = \bar{p}_d\left(m\right) + M_{dls} w_{dls}, \tag{6}$$

where p, d, $\bar{p}$, $p_d(m)$, l, and s are defined in (4); M_{dls} are variation modes (columns of M) for specific d, l, and s; and w_{dls} are weights of the variation modes, where the weights are typically set such that the variation is within three standard deviations.

For any shape profile type, multiple variation modes can be activated by setting the corresponding weighting factors to nonzero values. Each variation mode acts at a certain location and scale; hence we obtain

$$p_d\left(m\right) = \bar{p}_d\left(m\right) + \sum_l \sum_s M_{dls} w_{dls}. \tag{7}$$

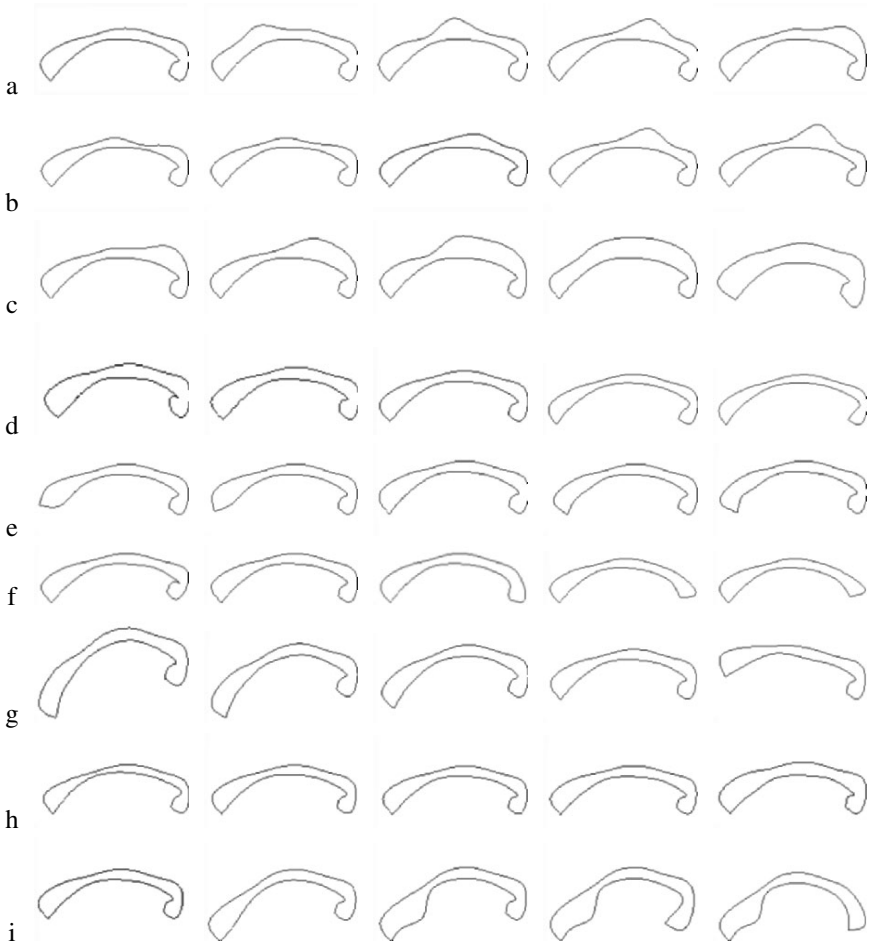


Figure 11. Examples of Controlled Deformations. (a)–(c) Operator-based bulge deformation at varying locations, amplitudes, and scales. (d) Operator-based stretching with varying amplitudes over entire CC. (e)–(g) Statistics-based bending of the left end, the right end, and the left half of the CC. (h) Statistics-based bulge of the left and right thickness over the entire CC. (i) From left to right: (1) mean shape, (2) statistics-based bending of the left half, followed by (3) locally increasing the left thickness using an operator, followed by (4) applying an operator-based stretch and (5) an operator-based bend to the right side of the corpus callosum. Reprinted with permission from [63]. Copyright ©2004, World Scientific.

In summary, varying the weights of one or more of the variation modes alters the length, orientation, or thickness profiles and generates statistically feasible stretch, bend, or bulge deformations at specific locations and scales upon reconstruction. Examples of statistics-based deformations are shown in Figure 11e–h.

Combining Operator- and Statistics-Based Deformations In general, operator- and statistics-based deformations ((5) and (7)) can be combined as

$$p_d = \bar{p}_d + \sum_l \sum_s \left(M_{dls} w_{dls} + \sum_t \alpha_{dlst} k_{dlst} \right). \tag{8}$$

It is worth noting that several deformations, whether operator- or statistics-based, may spatially overlap. Furthermore, adding profiles of different scales, and hence different vector lengths, is possible by padding the profiles with zeros. Figure 11i shows an example of combining operator- and statistics-based deformations.

Hand-Crafted Application To demonstrate the potential of statistics- and operator-based controlled deformations, we handcrafted a deformation schedule for fitting the CC shape model to a midsagittal MRI slice of the brain. Figure 12 shows the resulting medial shape profiles after applying the fitting schedule (compare with the initial profiles in Figure 10). The initial and final CC shapes are shown in Figure 13. The schedule steps are shown in Table 1 and the resulting deformed CC shapes for each step of the schedule are depicted in Figure 14.

3.3.2. 3D Shape Representation and Deformation using Medial Patches

We now extend the idea of the medial profiles method for 2D shape representation and deformation (presented in Section 3.3.1) to 3D. Instead of a single 1D orientation profile and a 1D elongation profile describing a primary medial axis, we now require two 2D orientation patches and a 2D elongation patch that together describe a primary medial sheet. Further, instead of two 1D thickness profiles describing the boundary of a 2D shape, we now require two 2D thickness patches to describe the surface the 3D object. Furthermore, rather than using 1D operators to change the profiles and generate new 2D shapes upon reconstruction, we now require 2D operators to alter the patches and generate new 3D shapes. As before, an operator applied to the orientation, elongation, or thickness components will cause a bend, stretch, or bulge shape deformation, respectively. In a simple manner, the location, extent, type, and amplitude of deformation operators can be specified and combined to capture local and global intuitive shape deformations. Figure 15 demonstrates the capability of producing different types of spatially localized deformations by varying different operator parameters. Deformation parameters include deformation type (e.g., bending, stretching, bulging), location, extent, amplitude, and type (e.g., Gaussian, rectangular, pyramidal, spherical).

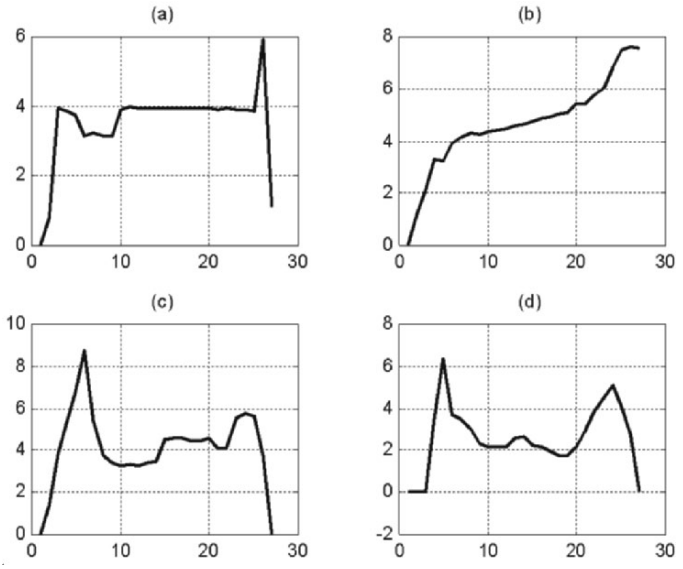


Figure 12. Resulting medial shape profiles after applying the fitting schedule: (a) length profile $L\,(m)$, (b) orientation profile $O\,(m)$, (c) left thickness profile $T^l\,(m)$, and (d) right thickness profile $T^r\,(m)$. Reprinted with permission from [63]. Copyright ©2004, World Scientific.

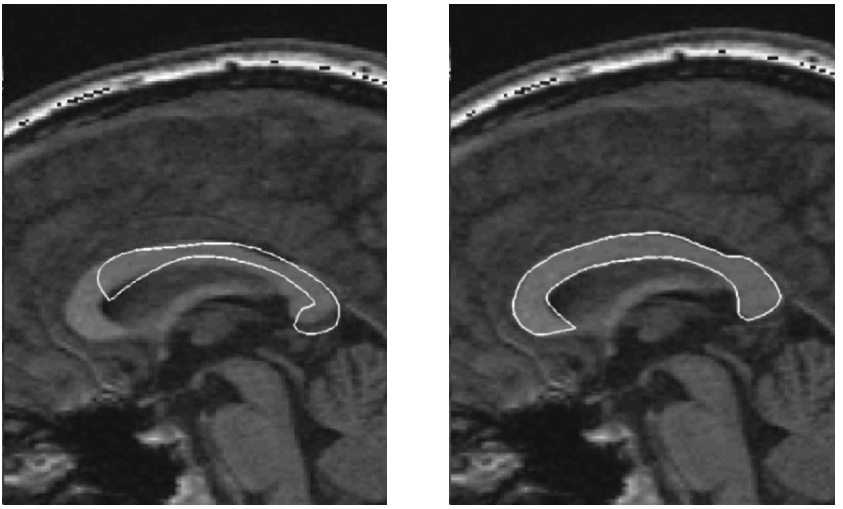


Figure 13. Close-up of the initial and final stages of the handcrafted fitting schedule. Reprinted with permission from [63]. Copyright ©2004, World Scientific.

Table 1. Deformation Schedule Used to Fit the Corpus Callosum Shape Model to MRI Data

Step	Deformation	Location	Scale	Variation mode/ Operator type	Variation mode weight/ Operator amplitude
1	Translation by ▼ 74, ▶ 24)				
2	Rotation counterclockwise by 10^o				
3	Scaling by 1.2				
4	Bend	1	8	2	$w = 0.5$
5	Bend	20	8	2	$w = -0.8$
6	Bend	22	6	2	$w = -0.75$
7	Bend	24	4	1	$w = 2.2$
8	Bend	1	4	2	$w = 1$
9	Stretch	6	4	1	$w = -1.5$
10	Stretch	26	1	1	$w = 2$
11	Left-bulge	15	7	1	$w = 3$
12	Left-bulge	18	3	1	$w = 2$
13	Left-bulge	6	12	1	$w = 3$
14	Left-bulge	5	3	1	$w = 3$
15	Right-squash	9	3	1	$w = -1$
16	Right-bulge	13	2	1	$w = 0.5$
17	Left-bulge	21	3	Gaussian	$\alpha = 0.3$
18	Left-bulge	21	7	Gaussian	$\alpha = 0.1$
19	Right-squash	24	2	Gaussian	$\alpha = -0.5$
20	Right-bulge	4	2	Bell	$\alpha = 1.7$
21	Right-bulge	6	3	Gaussian	$\alpha = 0.4$
22	Right-squash	1	3	Gaussian	$\alpha = -2.2$
23	Right-squash	25	1	Gaussian	$\alpha = -0.8$

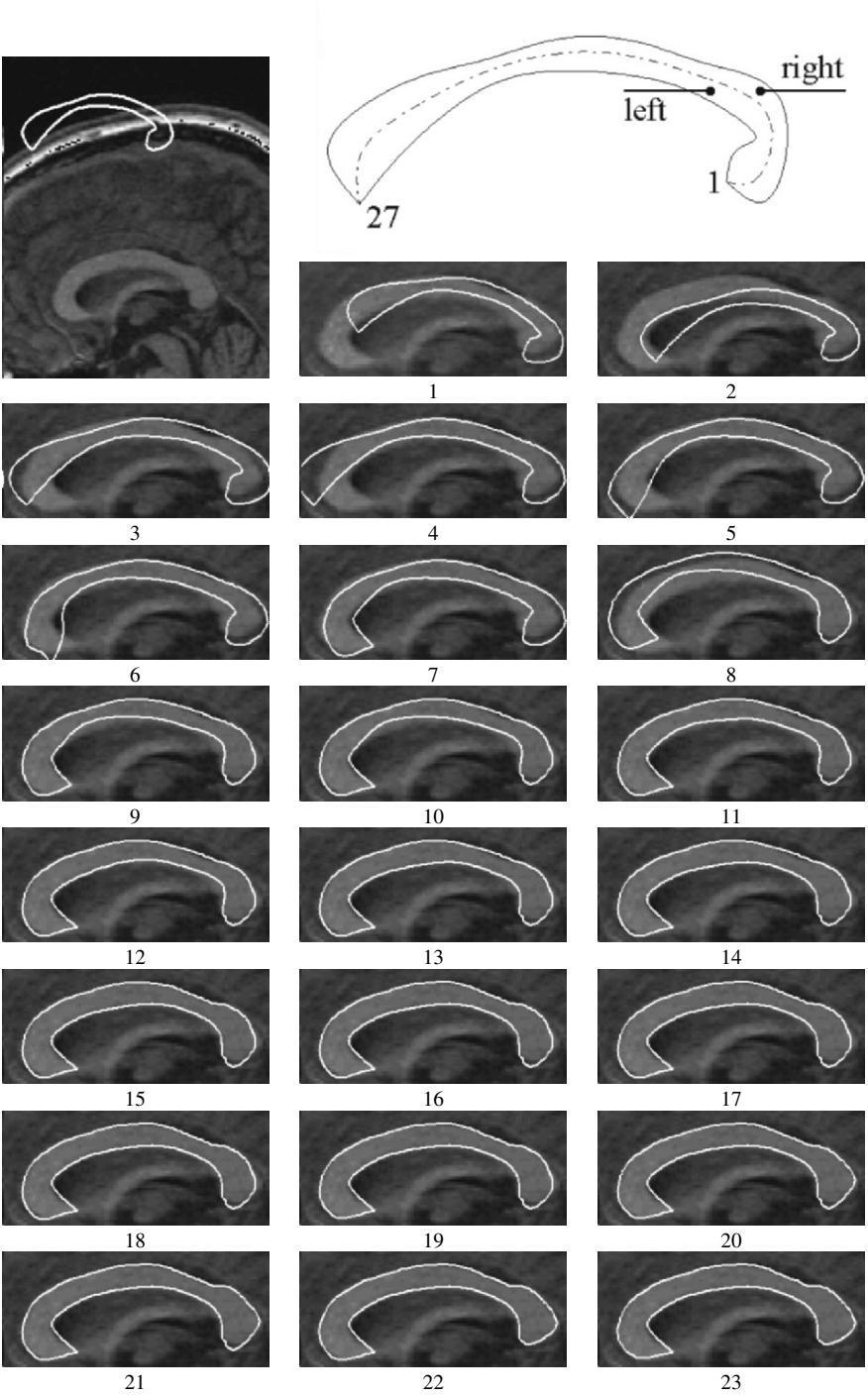


Figure 14. Progress of the handcrafted fitting schedule (fitting steps listed in Table 1). Reprinted with permission from [63]. Copyright ©2004, World Scientific.

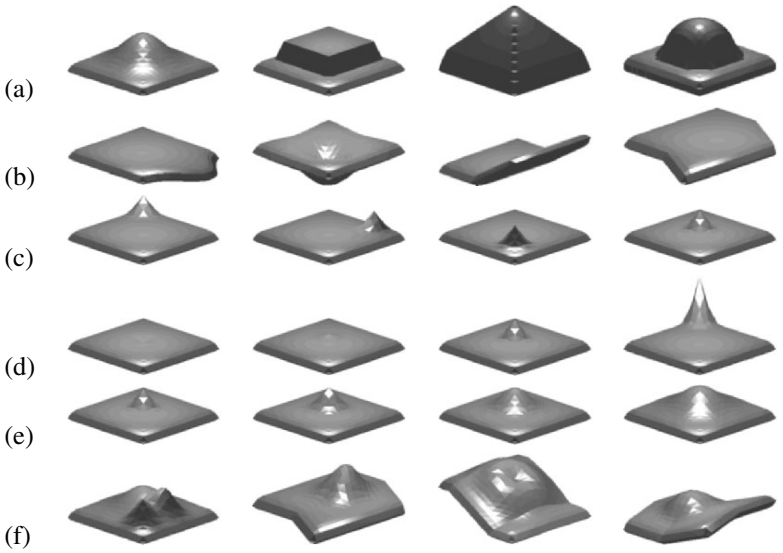


Figure 15. Deformations on a synthetic (slab) object. (a) Different operator types (left to right): Gaussian, rectangular, pyramidal, spherical. (b) Different deformation types: stretching, longitudinal bend, latitudinal bend. Different (c) locations, (d) amplitudes, and (e) extent of a bulge operator. (f) Combining stretching, longitudinal and latitudinal bend, and bulging deformations.

Because the effects of the deformation parameters are easily understood even to the user not familiar with the details of the shape representation, intuitive and accurate production of desired deformations is made possible. Results on real brain caudate nucleus and ventricle structures are presented in Section 4.5.

3.3.3. 2D Physics-based Shape Deformations

In this section we introduce the use of the physics-based shape representation and deformations method that is used for the deformable organism framework's geometrical and physical layers. This yields additional robustness by allowing intuitive real-time user guidance and interaction when necessary and addresses several of the requirements for an underlying shape representation and deformation layers stated previously. The deformations are realized through a physics-based framework implemented as meshes of connected nodes (mass-spring models). This inherently handles smoothness and continuity constraints, maintains the structural integrity of the body as it deforms, and facilitates intuitive user interaction. The mesh nodes and mesh connectivity are based on the medial axis of the object, yielding a shape representation and deformation method that naturally follows the geometry of the object. Various types of deformations at multiple locations and scale are controlled via operator- or statistics-based deformations that

actuate springs. The method also provides statistics-based feasible deformations that are derived from a hierarchical (multiscale) regional (multi-location) principal component analysis.

In the following sections we present the construction of the dynamic mesh model, operator-based shape deformations, similarity transformations, and statistically based shape deformations, using internal spring actuation and external forces. We demonstrate results on both synthetic data and on a spring-mass model of the CC, obtained from 2D midsagittal brain MRI slices.

Dynamic Spring-Mass Mesh Model For our physics-based deformable organisms we use spring-mass models to represent object shapes (Figures 16 and 17). The model is made up of nodes (masses or particles) and springs (elastic links or

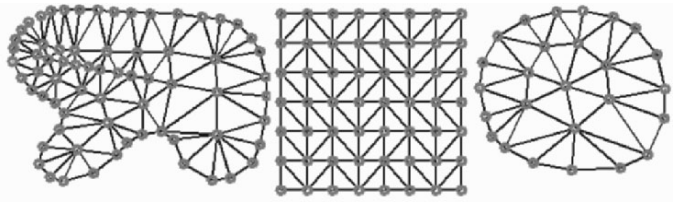
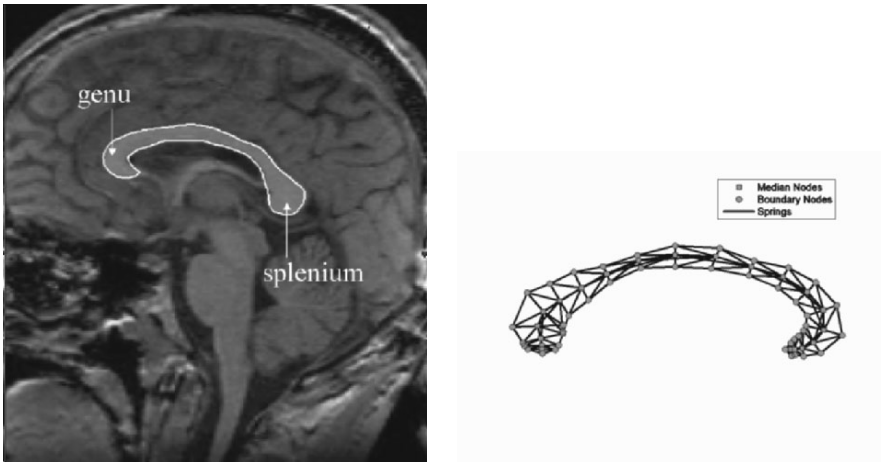


Figure 16. Examples of different synthetic spring-mass structures. Reprinted with permission from [80]. See attached CD for color version.



(a) (b)

Figure 17. (a) Midsagittal MR brain image with the corpus callosum (CC) outlined in white. (b) CC mesh model showing medial and boundary masses. Reprinted with permission from [80]. Copyright ©2005, SPIE.

connecting segments), and can be deformed according to Eqs. (22)–(26) in Section 1.6.7 in Chapter 11.

Specifically, the deformations can be caused by internal or external forces. External forces are those external to the organism, including image forces and user-driven mouse forces (Figure 18), whereas internal forces result from spring actuation (in a manner analogous to muscle actuation in animals), causing the attached nodes to change position, thereby propagating the force throughout the model (Figure 19). Spring actuation is accomplished by changing the spring's rest length while continuously simulating the mesh dynamics.

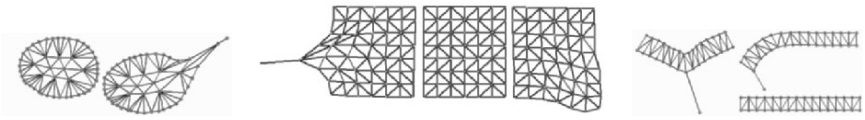


Figure 18. Examples of deformations via user interaction ("mouse" forces). Reprinted with permission from [62]. Copyright ©2003, SPIE. See attached CD for color version.

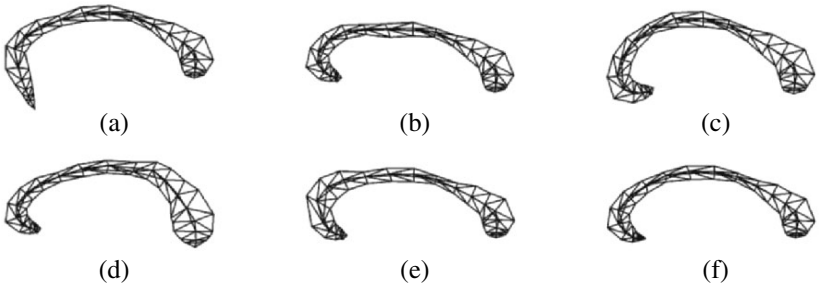


Figure 19. Examples of physics-based deformations of the CC organism using (a) user applied and (b) rotational external forces. Operator based (c) bending, (d) bulging, and (e) stretching deformations. (f) Statistics-based spring actuation. Reprinted with permission from [62]. Copyright ©2003, SPIE.

In order for deformable organisms to explore the image space, fit to specified regions, and take new shapes, they must be able to undertake a sequence of deformations. Some of these deformations take place independent of the topological design of the organism (general deformations), while others are designed specifically for medial-axis based worm-type organisms (medial-based deformations). Furthermore, in specific applications the general deformations can be modified to apply to specific anatomical regions of the model. By automatically fixing a group of nodes in place, the organism can perform deformations on specific regions of its geometrical model without affecting others. We use the terms *regional rotation/translation* and *boundary expansion* to refer to these types of deformations.

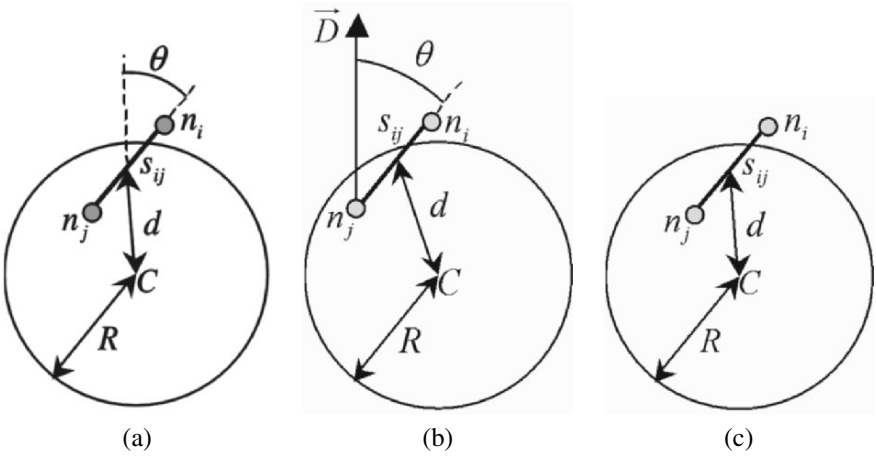


Figure 20. Definition of variables for (a) radial bulge, (b) directional bulge, and (c) localized scaling.

Boundary expansion refers to a sequence of stretching deformations whose direction is along thickness springs, while all nodes aside from the concerned boundary node are fixed. What follows is a description of the various types of deformations enabled.

Operator-Based Localized Deformations Using Internal Spring Actuation Localized bulging (radial bulge), stretching (directional bulge), scaling, bending, and tapering deformations are implemented using spring actuation. These operator-based deformations can be applied at different locations and scales with varying amplitudes. In the following paragraphs we describe how spring rest length must be changed to produce these different deformations.

To perform a (radial) bulge deformation we specify a center C and a radius R of a deformation region (Figure 20a) as well as a deformation amplitude K. We then update the rest length r_{ij} of each spring s_{ij} if at least one of its terminal nodes, n_i or n_j, lies within the deformation region, as follows:

$$r_{ij} = \left(\left(1 - \frac{d}{R}\right)\left(1 - \frac{2\theta}{\pi}\right)(K-1) + 1\right) r_{ij}^{\text{old}}, \tag{9}$$

where $\theta \in \left[0, \frac{\pi}{2}\right]$ is the angle between s_{ij}, and line L connecting the midpoint of the spring with C and d is the length of L (Figure 20a). The resulting effect of the above equation is that springs closer to C and with directions closer to the radial direction are affected more (Figure 21).

To perform a stretch (directional bulge) we again specify a deformation region and amplitude as well as a direction $\vec{D}$ (Figure 20b). We update the rest length of

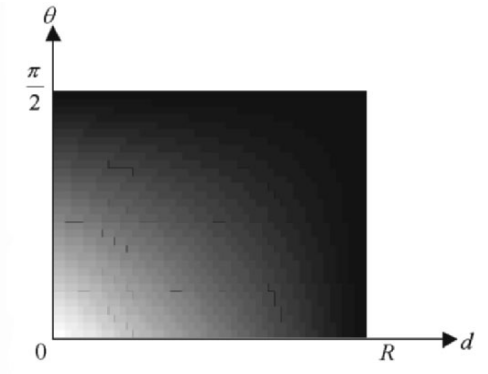


Figure 21. The coefficient (white = K, black = 1) by which r_{ij}^{old} is multiplied as a function of θ and d. Reprinted with permission from [62]. Copyright ©2003, SPIE.

each spring as in Eq. (9), where $\theta \in \left[0, \frac{\pi}{2}\right]$ is now defined as the angle between s_{ij} and $\vec{D}$ (Figure 20b). The resulting effect in this case is that springs closer to C and with directions closer to the stretch deformation direction are affected more (Figure 21).

A localized scaling deformation is independent of direction and requires only specification of a deformation region and amplitude (Figure 20c). The rest length update equation then becomes

$$r_{ij} = \left(\left(1 - d/R\right)\left(K - 1\right) + 1\right) r_{ij}^{\text{old}}. \tag{10}$$

To perform localized bending, we specify bending amplitude K and two regions surrounding the medial axis (Figure 22). The rest lengths of the springs on one side of the medial are increased according to

$$r_{ij}^{1} = \left(\frac{d_1}{R_1}\left(1 - \frac{2\theta_1}{\pi}\right)\left(K - 1\right) + 1\right) r_{ij}^{1,\text{old}}, \tag{11}$$

while the rest lengths on the other side are decreased according to

$$r_{ij}^{2} = \left(\frac{d_2}{R_2}\left(1 - \frac{2\theta_2}{\pi}\right)\left(\frac{1}{K} - 1\right) + 1\right) r_{ij}^{2,\text{old}}. \tag{12}$$

To perform localized tapering, we specify tapering amplitude K and a region with a base (Figure 22). The rest lengths on one side of the base are increased according to

$$r_{ij}^{1} = \left(\frac{d_1}{R_1}\left(K - 1\right) + 1\right) r_{ij}^{1,\text{old}}, \tag{13}$$

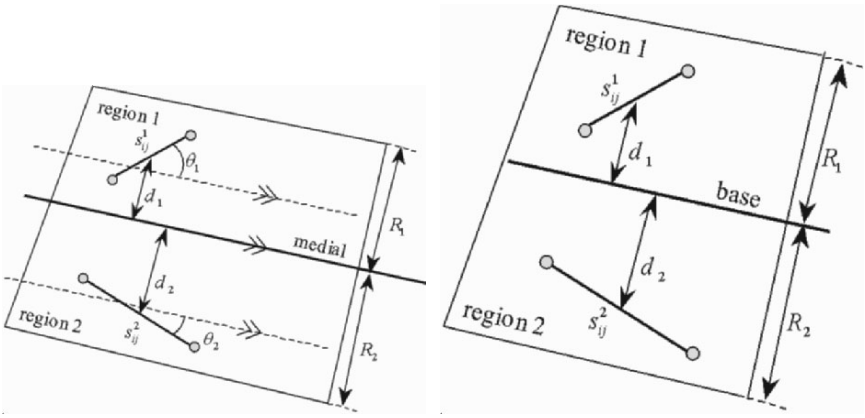


Figure 22. Definition of variables for deformation operators: bending (left), and tapering (middle). Reprinted with permission from [62]. Copyright ©2003, SPIE.

while those on the other side are decreased according to

$$r_{ij}^2 = \left(\frac{d_2}{R_2} \left(\frac{1}{K} - 1 \right) + 1 \right) r_{ij}^{2,\text{old}}. \tag{14}$$

Different examples of localized operator-based deformations are shown in Figure 25.

Similarity Transformations Using Internal and External Forces Simple rotation and translation are implemented via the application of external forces, while scaling is implemented by muscle actuation. Rotation forces are applied on all nodes in a direction normal to the line connecting each node with the center of mass of the model, with a consistent clockwise/counterclockwise direction (Figure 23a). While translation forces are applied on all nodes in the direction of the desired translation (Figure 23b). Scaling by a factor of S is performed by changing the rest length of all the springs, i.e., $r_{ij} = S \cdot r_{ij}^{\text{old}}$. Examples are shown in Figure 24.

Learned Deformations Using Internal Spring Actuation Learned or statistically based deformations are implemented via spring actuation. To facilitate intuitive deformations, springs are designed to be of different types: stretch springs, bend springs, or thickness springs. Stretch springs connect neighboring medial nodes, bending springs are hinge springs that connect nonconsecutive medial nodes, and thickness springs connect medial nodes with boundary nodes (Figure 26). Actuating the stretch springs causes stretch deformations, actuating hinge springs causes bend deformations, and actuating thickness springs causes bulging, squashing, or tapering deformations.

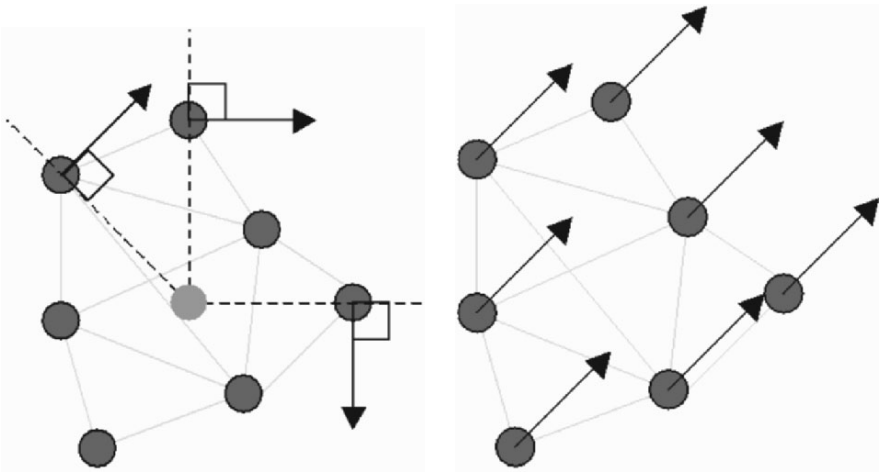


Figure 23. External forces for performing a (a) rotation (light gray circle marks center of mass) and a (b) translation. Reprinted with permission from [62]. Copyright ©2003, SPIE. See attached CD for color version.

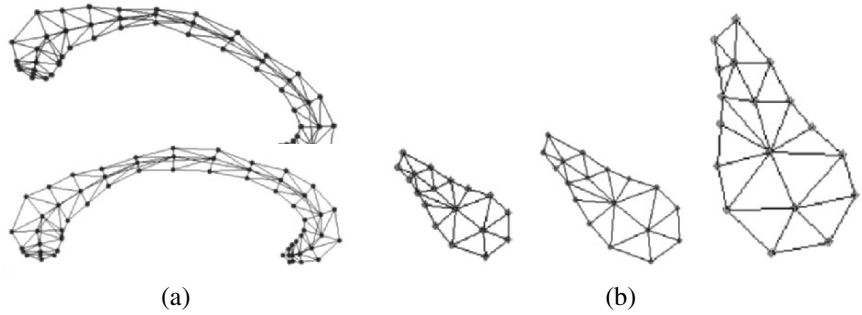


Figure 24. Similarity transformation via external forces. (a) Rotating a model of the CC. (b) Scaling and rotating a synthetic model. Reprinted with permission from [62]. Copyright ©SPIE 2005.

Feasible mesh deformations (i.e., similar to what has been observed in a training set) at different locations and scales are obtained by actuating springs according to the outcome of a statistical analysis. Specifically, PCA is applied to a training set of spring rest lengths in the region corresponding to the desired localized deformations.

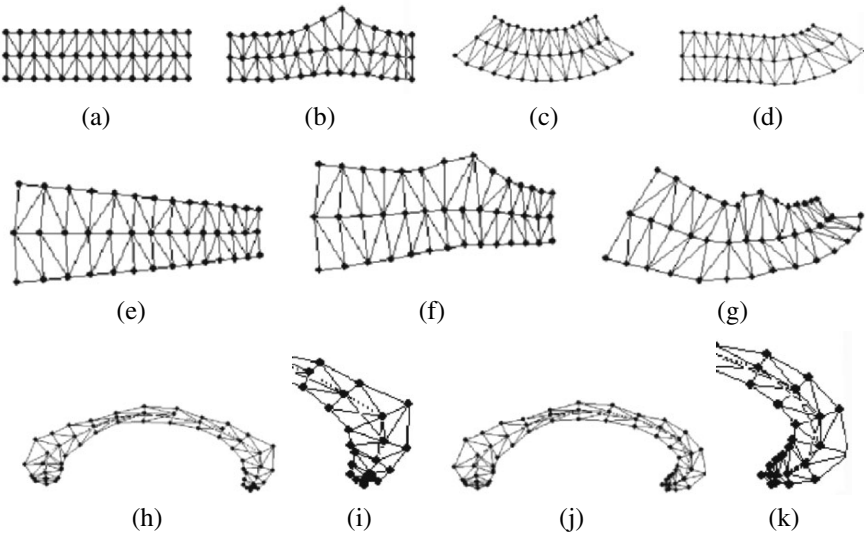


Figure 25. Examples of localized deformations: (a) initial synthetic object, (b) bulge, (c) bend, (d) bend at another location, (e) tapering, (f) tapering followed by a bulge, and (g) tapering followed by a bulge and a bend deformations. CC model (h) before and (j) after a localized bend. (i,k) Close-up versions of (h,j). Reprinted with permission from [62]. Copyright ©2003, SPIE.

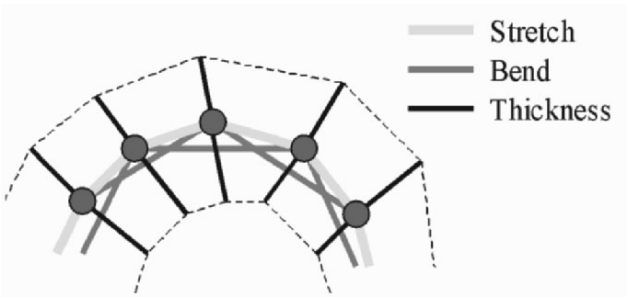


Figure 26. Spring types used for statistics-based deformations. Reprinted with permission from [62]. Copyright ©2003, SPIE. See attached CD for color version.

The set of rest lengths for the stretch springs (Figure 26) in a single example model are collected in a vector $\mathbf{r}_S$, i.e.,

$$\mathbf{r}_S = \{r_{ij} \forall i, j : s_{ij} \in \text{stretch springs}\}, \tag{15}$$

and similarly for the bending and left and right thickness springs: (Figure 26)

$$\begin{aligned} \mathbf{r}_B &= \{r_{ij} \forall i, j : s_{ij} \in \text{bend springs}\}, \\ \mathbf{r}_{TL} &= \{r_{ij} \forall i, j : s_{ij} \in \text{left thickness springs}\}, \\ \mathbf{r}_{TR} &= \{r_{ij} \forall i, j : s_{ij} \in \text{right thickness springs}\}. \end{aligned} \tag{16}$$

This gives

$$\begin{aligned} \mathbf{r}_S &= \left[\mathbf{r}_S^1, \mathbf{r}_S^2, \ldots, \mathbf{r}_S^{N_S}\right], \\ \mathbf{r}_B &= \left[\mathbf{r}_B^1, \mathbf{r}_B^2, \ldots, \mathbf{r}_B^{N_B}\right], \\ \mathbf{r}_{TL} &= \left[\mathbf{r}_{TL}^1, \mathbf{r}_{TL}^2, \ldots, \mathbf{r}_{TL}^{N_T}\right], \\ \mathbf{r}_{TR} &= \left[\mathbf{r}_{TR}^1, \mathbf{r}_{TR}^2, \ldots, \mathbf{r}_{TR}^{N_T}\right], \end{aligned} \tag{17}$$

where N_S, N_B, N_T are the numbers of stretch, bend, and left/right thickness springs, and the springs are ordered spatially (i.e., moving from one end of the medial to the other we encounter $\mathbf{r}_S^1, \mathbf{r}_S^2, \ldots, \mathbf{r}_S^{N_S}$). Performing global (traditional) PCA on corresponding variables in a training set gives

$$\begin{aligned} \mathbf{r}_S &= \bar{\mathbf{r}}_S + M_S \mathbf{w}_S, \\ \mathbf{r}_B &= \bar{\mathbf{r}}_B + M_B \mathbf{w}_B, \\ \mathbf{r}_{TR} &= \bar{\mathbf{r}}_{TR} + M_{TR} \mathbf{w}_{TR}, \\ \mathbf{r}_{TL} &= \bar{\mathbf{r}}_{TL} + M_{TL} \mathbf{w}_{TL}, \end{aligned} \tag{18}$$

where the columns of M_S, M_B, M_{TR}, M_{TL} are the main modes of spring length variation. Associated with each mode is the variance it explains.

For capturing the shape variations at different locations and scales, we study the variations in the rest lengths of the springs in the desired localized region. Furthermore, to decompose the variations into different types of general deformations, each statistical analysis of the spring length in a localized region is restricted to a specific type of deformation springs (Figure 26). Accordingly, the PCA becomes a function of the deformation type, location, and scale. For example, to analyze the local variation in object length (stretch), we perform a statistical analysis on the lengths of the stretch springs of that local region. In general, for a single deformation/location/scale- specific PCA we obtain

$$\mathbf{r}_{\text{def,loc,scl}} = \bar{\mathbf{r}}_{\text{def,loc,scl}} + M_{\text{def,loc,scl}} \mathbf{w}_{\text{def,loc,scl}}, \tag{19}$$

where def is the deformation type, being either S (for stretch), B (for bend), TL (for left thickness), or TR (for right thickness). The location and scale, determined by

the choice of loc and scl, respectively, determine which springs are to be included in the analysis according to

$$\mathbf{r}_{\mathrm{def,loc,scl}} = \left[\mathbf{r}_{\mathrm{def}}^{\mathrm{loc}}, \mathbf{r}_{\mathrm{def}}^{\mathrm{loc}+1}, \ldots, \mathbf{r}_{\mathrm{def}}^{\mathrm{loc}+\mathrm{scl}-1}\right]. \tag{20}$$

For example, for the bending deformation at location "five" with scale "three" $(\mathrm{def}, \mathrm{loc}, \mathrm{scl} = B, 5, 3)$ we have

$$\mathbf{r}_{\mathrm{def,loc,scl}} = \mathbf{r}_{B,5,3} = \left[\mathbf{r}_B^5, \mathbf{r}_B^6, \mathbf{r}_B^7\right]. \tag{21}$$

The average values of the spring lengths are calculated according to

$$\bar{\mathbf{r}}_{\mathrm{def,loc,scl}} = \frac{1}{N}\sum_{j=1}^{N} \mathbf{r}\left(j\right)_{\mathrm{def,loc,scl}}, \tag{22}$$

where $\mathbf{r}\left(j\right)_{\mathrm{def,loc,scl}}$ is $\mathbf{r}_{\mathrm{def,loc,scl}}$ obtained from the jth training example, and N is the number of training examples. The columns of $M_{\mathrm{def,loc,scl}}$ are the eigenvectors, $m_{\mathrm{def,loc,scl}}$, of covariance matrix $C_{\mathrm{def,loc,scl}}$. That is,

$$\left\{C\mathbf{m} = \lambda\mathbf{m}\right\}_{\mathrm{def,loc,scl}}, \tag{23}$$

where

$$\left\{C = \frac{1}{N-1}\sum_{j=1}^{N}\left(r(j)-\bar{r}\right)\left(r(j)-\bar{r}\right)^T\right\}_{\mathrm{def,loc,scl}}, \tag{24}$$

and where $\{\}_{\mathrm{def,loc,scl}}$ denotes deformation type-, location-, and scale-specific PCA variables.

The data set needs to be aligned only with respect to scale. The statistical analysis of spring lengths is independent of orientation and translation. See the different examples in Figures 27–30.

3.3.4. 3D Physics-Based Shape Deformations

Increasing the dimensionality of the position, force, velocity, and acceleration vectors to 3D in Eqs. (22)–(26) (Section 1.6, Chapter 11) enables calculation of the 3D deformations, as opposed to 2D. External forces are provided by the volumetric image gradient and a drag force, while internal forces are supplied through Hooke's law and dampening spring forces.

Here we present a tubular geometric module parametrized by the distance between neighboring medial masses and the number of circumferential boundary masses (Figure 31). This layered medial shape representation enables intuitive deformations [62, 63, 76] wherein the medial axis governs the bending and

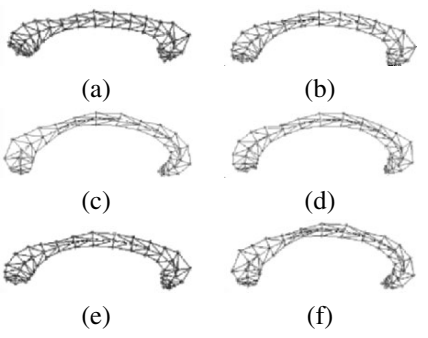


Figure 27. Sample corpus callosum mesh model deformations (1st PC for all deformation types over the entire CC) derived from the hierarchical regional PCA. Reprinted with permission from [62]. Copyright ©2003, SPIE.

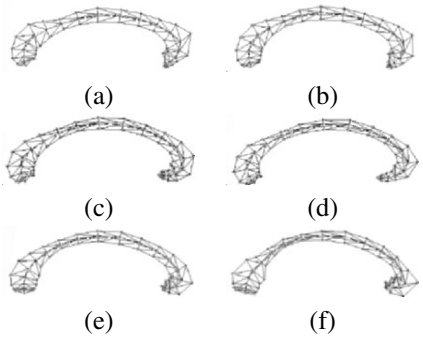


Figure 28. Sample CC mesh model deformations (2nd PC for all deformation types over the entire CC) derived from the hierarchical regional PCA. Reprinted with permission from [62]. Copyright ©2003, SPIE.

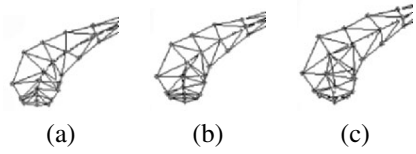


Figure 29. Statistical CC mesh model deformations: stretching the splenium. Reprinted with permission from [62]. Copyright ©2003, SPIE.

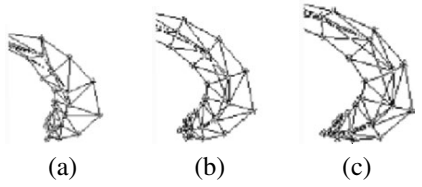


Figure 30. Statistical CC mesh model deformations: bending the genu. Reprinted with permission from [62]. Copyright ©2003, SPIE.

Figure 31. Topology of the vessel crawler showing: Masses (left), radial springs (middle), and stability springs (right) across sequential layers of the organism. See attached CD for color version.

stretching of the vessel crawler and links to boundary nodes to control thickness. As deformable organisms are typically modeled after their target structures, we employ this tubular topology to the task of vascular segmentation and analysis in volumetric medical images [86]. We provide results in Section 4.6.

3.4. Perception System

The perception system of the deformable organism consists of a set of sensors that provide image information to the organisms. Using virtual sensors an organism can collect data about the world they live in, such as image data or data from interaction with other organisms. For example, a single organism may have image intensity sensors, edge sensors, or texture sensors. A wide variety of image processing techniques can be used to enable the organism to perceive relevant features of their surroundings. Sensors can be focused or trained for a specific image feature in a task-specific way and hence the organism is able to disregard sensory information superfluous to its current behavioral needs. Different parts of the organism are dynamically assigned sensing capabilities and thus act as sensory organs or receptors. A sensor can either be on-board or off-board. On-board sensors are confined to the organism's body such as at its medial or boundary nodes, curves or segments connecting different nodes, or at internal geometric subregions, while off-board sensors are free floating. Once the sensory data are gathered, they are fed to the cognitive center of the brain for processing. In the following sections we highlight the main sensors used in a variety of deformable organisms.

- **Geometrical CC deformable organism:** In our implementation of the geometrical CC deformable organism, the sensors are made sensitive to different stimuli, including image intensity, image gradient magnitude and direction, multiscale edge-preserving diffusion filtered images [77, 78] a Canny-edge detected version of the image, and the result of a Hough transform applied to locate the top of the human skull in the brain MR image (results in Section 4.1).
- **Physics-based CC deformable organism:** This deformable organism was equipped with sensors for collecting statistical measures of image data such as mean and variance, in addition to intensity and edge strength and direction sensors (results in Section 4.4).
- **2D Vessel Crawler:** Once vessel-crawling deformable organisms are initialized in a target vascular system, they use a variety of sensory input modules to make decisions about which direction to grow in, and where bifurcations are located. For example, the 2D angiogram vessel crawler is equipped with an off-board arc sensor (Figure 32) in order to differentiate between overlapping vessels and authentic branch points. This crawler is also equipped with simple sensors for measuring image intensity and edge magnitude and direction (results in Section 4.3).

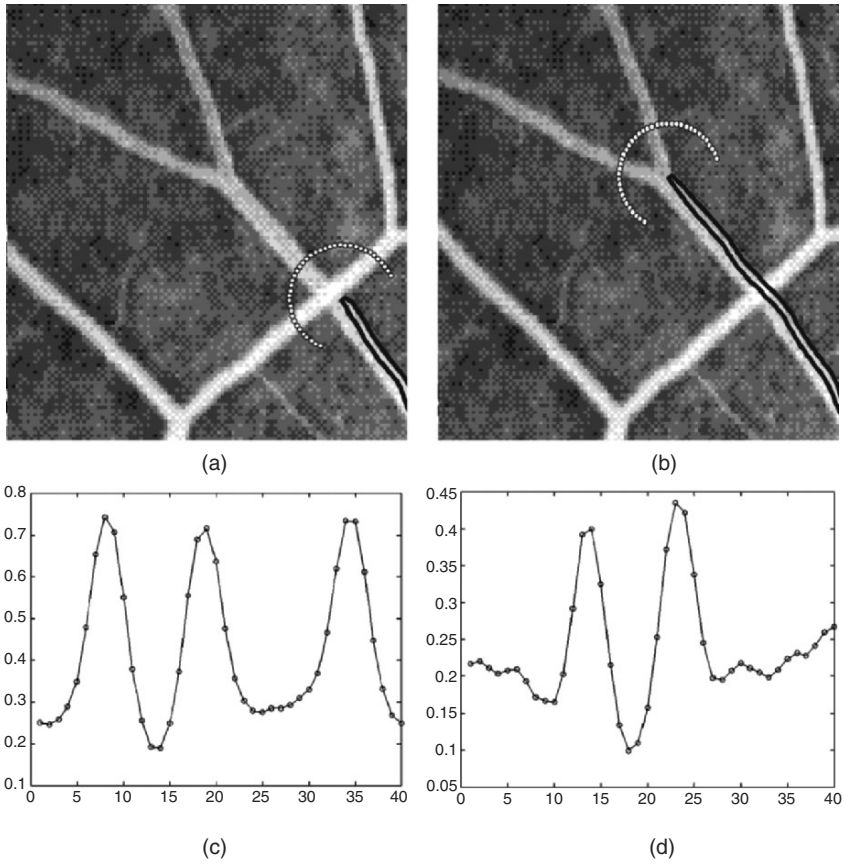


Figure 32. Off-board sensors (arc of white nodes in (a) and (b)) measure image intensity (along the arc). This results in an intensity profile exhibiting three distinct peaks when an overlapping vessel is ahead (c) and only two peaks in the case of a bifurcation (d).

- **3D vessel crawler:** The 3D vessel crawler utilizes two primary sensory modules, a vesselness sensor and a Hessian sensor, which both aide in guiding the crawler to the direction in which it should grow. The vesselness sensor is an extension of the 2D arc intensity sensor (Figure 32) to a 3D hemispherical sensor collecting vesselness values [79] (Figure 33). The Hessian sensor uses the smallest eigenvalued eigenvector $\vec{v}_1$ of Hessian matrix H.

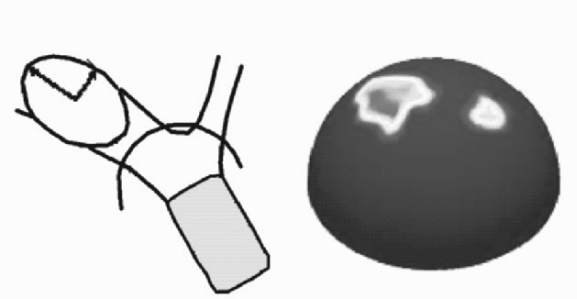


Figure 33. A vessel crawler (left, in gray) utilizing an off-board hemispherical sensor (shown as an arc in the left-hand image). The sensor (shown in 3D on the right) collects vesselness measures, guiding it as it crawls along the vessel and detects branching points. See attached CD for color version.

3.5. Behavioral Layer

A behavior is classically defined as the action or reaction of a life form in response to external or internal stimuli. The deformable organism's behaviors are a set of actions performable in response to perceived input or user control. Specifically, behavioral routines are designed based on available organism motor skills, perception capabilities, and available anatomical landmarks. For example, the routines implemented for the geometrically based CC deformable organism include: find-top-of-head, find-upper-boundary-of-CC, find-genu, find-rostrum, find-splenium, latch-to-upper-boundary, latch-to-lower-boundary, find-fornix, thicken-right-side, thicken-left-side, back-up (more details in Section 4.1). Each behavior routine subsequently activates the appropriate deformation or growth controllers to complete a stage in the plan and bring an organism closer to its target shape. For example, the latch-to-upper-boundary behavior causes the CC organism to latch the upper-half of itself to the top of the CC (Figure 34). In what follows we describe the primary behaviors of two example deformable organisms.

3.5.1. Example Behaviors of the Physics-Based CC Deformable Organism

We describe below the different behavioral routines necessary to achieve successful segmentation of the CC using the physics-based deformable organism (Figure 5b). Its anatomically tailored behaviors include: model initialization, global and regional alignment, expansion and contraction, medial alignment, fitting to boundary, and detecting and repairing segmentation inaccuracies. We describe the key behaviors below; for complete details refer to [80].

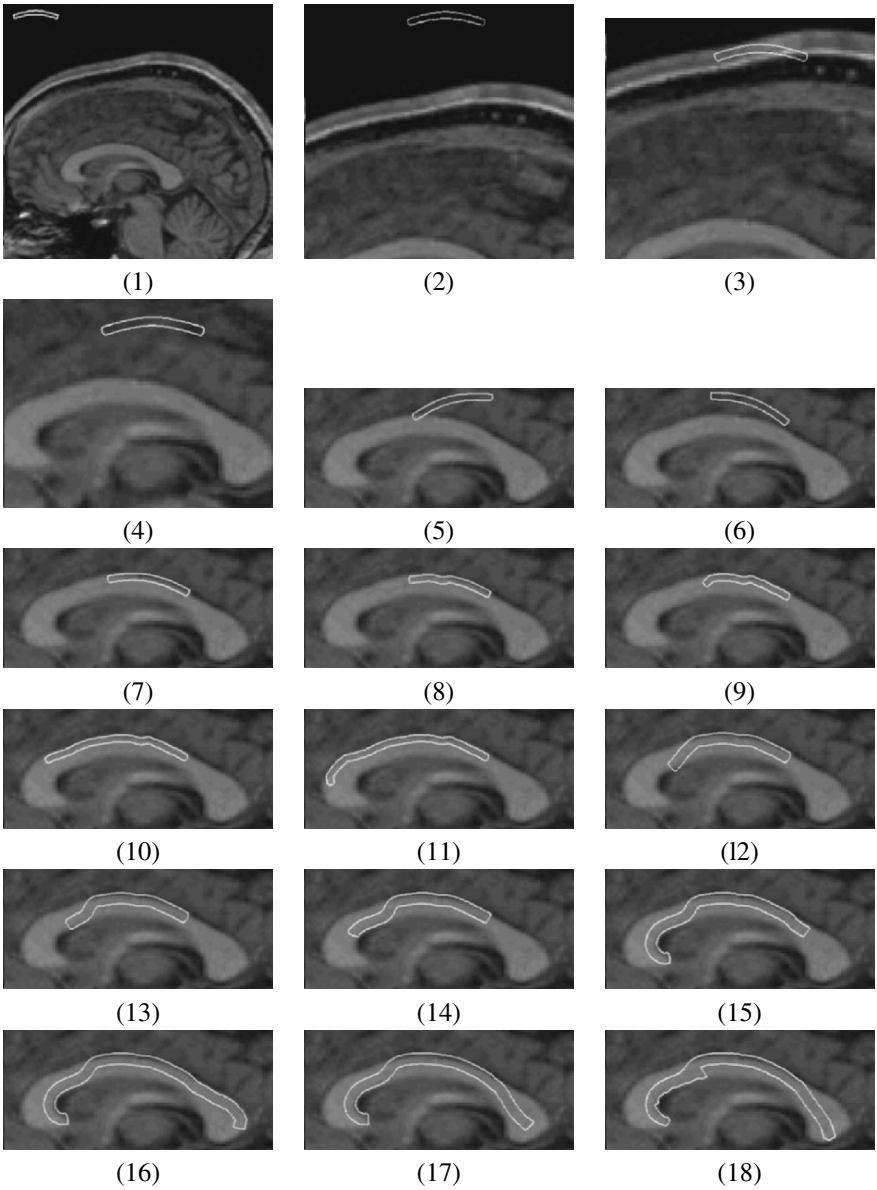


Figure 34. Deformable corpus callosum organism progressing through a sequence of behaviors to segment the CC (results continued on next page). Reprinted with permission from [85]. Copyright ©2002, Elsevier.

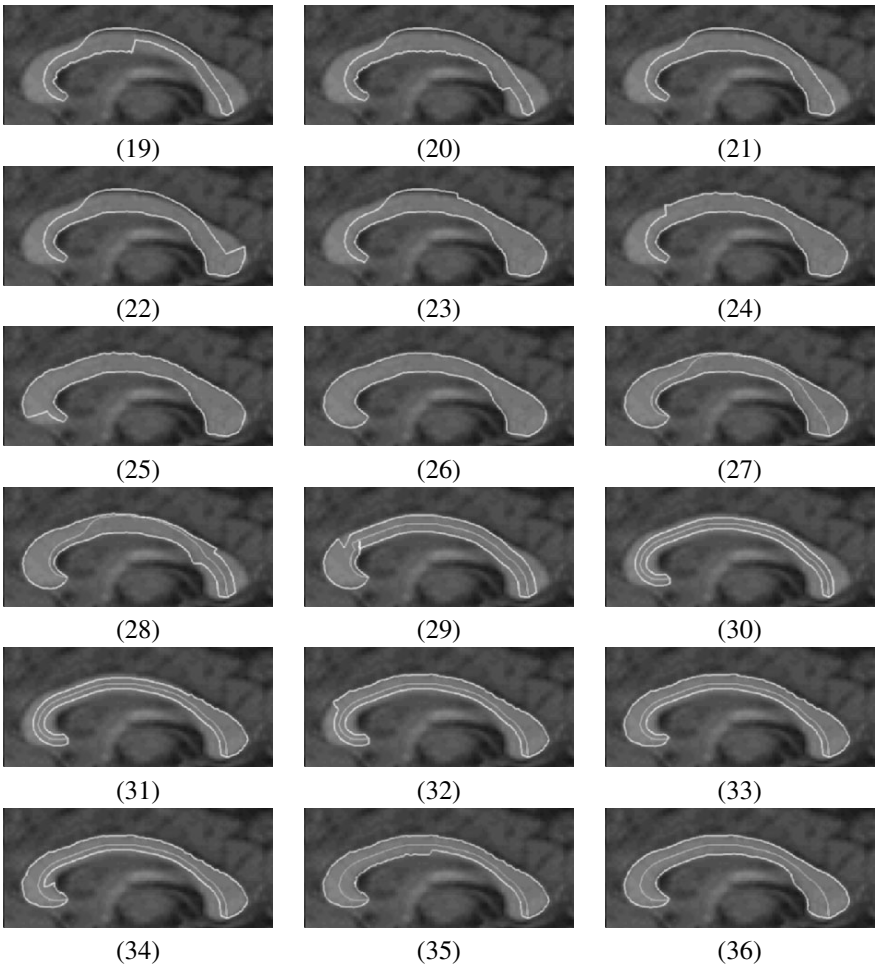


Figure 34. (continued). Results continued from previous page.

- **Model Initialization:** To begin its search for the CC, the deformable organism must be initialized to an appropriate location in the image using robust anatomical features. The organism's first step is to locate the top of the head using a modified Hough transform [81] to search for the largest ellipse (skull). Then, in response to the output of the skull sensor, a CC template is built consisting of a rectangular window connecting two squares to approximate the main body, genu, and splenium, respectively (see Figure 1). Maximizing average image intensity and minimizing intensity variance over the shape model yields several candidate locations for the three CC

parts. Finally, a decision is made and the set of candidate parts that exhibits the strongest edge connectivity and exhibits maximal distance between the parts is selected.

- **Global and Regional Alignment:** The global alignment phase is designed to find the best overall location for the model using external forces applied to the entire organism to rotate and position it as necessary. The organism first locates the optimal horizontal and vertical location using translational forces, and then finds the best alignment using rotational forces. After searching its local space, optimal translations are chosen using a position decision based upon a sensory module configured to monitor local image mean, and variance across the entire model. As the CC is generally a bright, homogenous region, the decision function attempts to maximize the image mean, while minimizing the variance (more details presented in Section 3.6.1). During the regional alignment behavior, the organism aligns its anatomical parts to corresponding sections of the CC through successive phases. During each phase, rotational and translational forces are applied to only one part of the CC model. The phases are ordered as splenium-, genu-, and finally rostrum-alignment, as this particular ordering favors segmenting regions with stable features before proceeding to other regions. Optimal positions are decided upon using a position decision function utilizing the same sensory module as used in the global alignment phase.

- **Detecting and Repairing Segmentation Inaccuracies:** An example of the organism's ability to incorporate anatomical knowledge is its ability to detect and repair the fornix dip (Figure 1). Typically, the anatomical structure of a corpus callosum exhibits a symmetrical property about its medial axis. Consequently, if the organism is nonsymmetrical in the region where the fornix dip is located, then that area needs repair. In order to repair itself, the organism simply mirrors its top half about the medial axis throughout the affected area. To ensure validity, the organism then checks if the deformation has been beneficial using a position decision function designed to maximize local image mean, and minimize local image variance (Section 3.6.1).

3.5.2. Example Behaviors of the 3D Vessel Crawler

As the behaviors vary from organism to organism in both design and complexity, we also provide examples of those used in our 3D vessel crawler. Each of the vessel crawler's key decisions results in the execution of the appropriate behavior, using the concluded locally optimal parameters such as scale, estimated vessel mean and variance, etc. The primary behaviors available to the organism are to grow, to fit the vessel wall, and to spawn new organisms.

- **Growing:** Once initialized at a seed point, the vessel crawler must grow outward along the vasculature by sequentially adding new layers centered around the final sensor position using the 3D hemispherical and Hessian-based sensors (described in Section 3.4). As it grows, each new layer must be created and subsequently connected to the current end most layer (Figure 31). The newest layer is aligned to the provided direction vector, and then connected via a consistent clockwise ordering to prevent mesh twisting. Once connected the model can be fit to the image data.

- **Fitting**: The organism fits itself to the vessel boundary using 3D image gradient driven deformations simulated by the physics layer (Section 3.3.4). Connections to the previous layer provide smoothness, while stiffer circumferential springs provide local stability to noise, and flexible radial springs allow deformation to the vessel boundary (Figure 31).

- **Spawning new organisms**: In order to branch off and explore bifurcations, the organism must be able to spawn new vessel crawlers. Each spawned vessel crawler is initialized based on the optimal parameters detected by the parent. For example, the initial radius is based on the parent's detected radius at the spawn point using the spherical sensory module (Section 3.4).

3.6. Cognitive System

The cognitive layer of the architecture combines memorized information, prior anatomical knowledge, a segmentation plan, and an organism's sensory data (Section 3.4) in order to initiate behaviors, carry out shape deformations, change sensory input, and make decisions toward segmenting the target structure (Figure 4). A single fixed path or multiple paths with a plan selection scheme can be implemented. In the first case the organism follows a sequential flow of control, proceeding directly from one behavior to the next. In the latter case, the organism decides between different options within the plan, thus taking a different path than it might have given a different image or different initialization conditions.

Most often the segmentation plan is subdivided into different behavioral stages with subgoals that are easy to define and attain (e.g., locating the upper boundary of an anatomical structure). Consequently, the segmentation plan provides a means for human experts to intuitively incorporate global contextual knowledge. It contains instructions on how best to achieve a correct segmentation by optimally prioritizing behaviors. If we know, for example, that the superior boundary of the CC is consistently and clearly defined in an MRI image, then the find-upper-boundary behavior should be given a very high priority as segmenting stable features will provide good initialization for less-stable ones. Adhering to the segmentation plan and defining it at a behavioral level infuses the organism with awareness of the segmentation process. This enables it to make effective use of prior shape

knowledge — it is applied only in anatomical regions of the target object where a high level of noise or gaps in the object's boundary edge are known to exist.

In the following two subsections we first describe a sequential deformable organism using a pre-designed segmentation plan with a known and fixed sequence of behaviors. As previously noted, a general and more flexible approach is to have different options available for which behaviors the organisms can carry out next. We then present and describe a deformable organism with an adaptive segmentation plane, under which the organism's sequence of behaviors is adaptively selected during its life span.

3.6.1. Sequential Behavior Selection: Physics-Based CC Deformable Organism

Sequential control takes form in the way of event or schedule driven actions, and sensory or knowledge-based decisions. The actions are initiated by the behavioral layer and simulated by the physics layer (Section 3.3.3). Decisions are made at scheduled times according to user-provided decision functions. Using specialized decision functions, the deformable organism can decide on the best position across a sequence of performed deformations, and whether or not to continue its expansion/contraction behavior (Figure 42). We provide one such example below; for complete details see [80].

- **Shape Configuration Decision.** The organism decides on the best new shape configuration (for the whole organism or for a part of its body) by first capturing sensory input as it is deforming to different shape configurations, and then choosing the configuration at which the sensory input maximizes a certain objective function. For example, the following strategy causes the deformable organism to prefer bright homogenous regions (like the CC) over dim nonuniform regions. The organism first collects n values for $d_i = \alpha m_i + (1 - \alpha)v_i^{-1}$, where m_i and v_i are image intensity and variance, respectively, at n different shape configurations; it then decides to take on the shape configuration that gives the maximum d.

3.6.2. Adaptive Behavior Selection: 3D Vessel Crawler

The vessel crawler can make a number of key decisions at any point in its life span, where each decision can be based on sensory input, anatomical knowledge, or user interaction. The outcome of each decision can affect the vessel crawler's internal settings, or how it should navigate the segmentation plan. It should be noted that the user is able to override any of these key decision functions at any time during the organism's life cycle, and hence can illicit intuitive real-time control over the segmentation process. A particular decision that affects the choice of subsequent behaviors of the organism is whether a bifurcation has been detected or not.

- **Bifurcation Verification:** The analysis of the vesselness values returned by the hemispherical sensor (Figure 33) is used to detect bifurcations. Basically, if two disjoint high-vesselness magnitude regions appeared on the hemispherical slice, then this would indicate a bifurcation. If a bifurcation is verified, then two new organisms are spawned, one for each branch, while a single high-vesselness region indicates a single branch and therefore a "grow behavior" is initiated.

4. RESULTS AND APPLICATIONS

4.1. Geometrical CC Deformable Organism

We first present a detailed segmentation plan for the CC organism that serves to illustrate an ability to harness global contextual knowledge. A CC organism is released into a 2D midsagittal MRI brain image from an initial default position (Figure 34.1). It then goes though different "behaviors" as it progresses toward its goal. As the upper boundary of the CC is very well defined and can be easily located with respect to the top of the head, the cognitive center of the CC activates behaviors to first locate the top of the head (Figure 34.2–3); it then moves downward through the gray and white matter in the image space to locate the upper boundary (Figure 34.4–7). The organism then bends to latch onto the upper boundary (Figure 34.8) and activates a find-genu routine (refer to Figure 1 for CC anatomy), causing the CC organism to stretch and grow along this boundary toward the genu (Figure 34.9–11). It then activates the find-rostrum routine causing the organism to back up, thicken (Figure 34.12), and track the lower boundary until reaching the distinctive rostrum (Figure 34.13–15). Once the rostrum is located, the find-splenium routine is activated and the organism stretches and grows in the other direction (Figure 34.15–16). The genu and splenium are easily detected by looking for a sudden change in direction of the upper boundary toward the middle of the head. At the splenium end of the CC, the organism backs up and finds the center of a circle that approximates the splenium end cap (Figure 34.17). The lower boundary is then progressively tracked from the rostrum to the splenium while maintaining parallelism with the organism's medial axis in order to avoid latching onto the potentially occluding fornix (Figure 34.18–21). Nevertheless, the lower boundary might still dip toward the fornix, so a successive step of locating where, if anywhere, the fornix does occlude the CC is performed by activating the find-fornix routine (making use of edge strength along the lower boundary, its parallelism to the medial axis, and statistical thickness values). Thus, prior knowledge is applied only when and where required. If the fornix does indeed occlude the CC, any detected dip in the organism's boundary is repaired by interpolation using neighboring thickness values (Figure 37). The thickness of the upper boundary is then adjusted to latch onto the corresponding boundary in the image (Figure 34.22–26). At this point the boundary of the CC is located (Figure 34.26), and the CC organism has almost

reached its goal. However, at this stage the medial axis is not in the middle of the CC organism (Figure 34.27) so it is re-parametrized until the medial nodes are halfway between the boundary nodes (Figure 34.28–30). Finally, the upper and lower boundaries, which were reset in the previous step, are relocated (Figure 34.31–36) to obtain the final segmentation result (Figure 34.36). Other CC segmentation (Figure 35), validation results (Figure 36), and a demonstration of the organism's self-awareness (Figure 38) are presented.

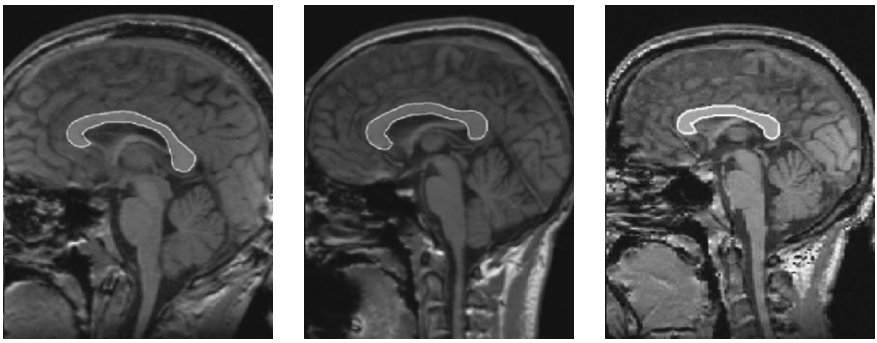


Figure 35. Segmentation results. Reprinted with permission from [38]. Copyright ©2002, Springer.

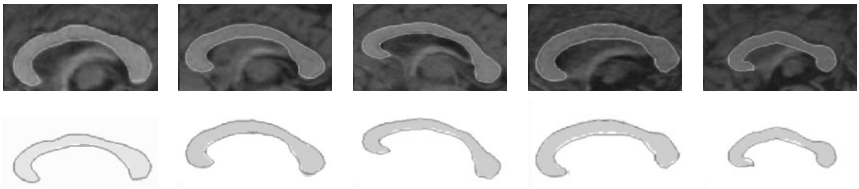


Figure 36. Segmentation results (top), also shown (in black) over manually segmented (gray) corpora callosa (bottom). Reprinted with permission from [85]. Copyright ©2002, Elsevier.

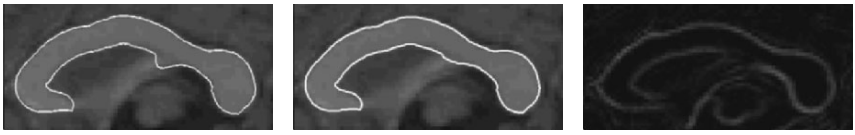


Figure 37. Segmentation result (a) before and (b) after detecting and repairing the fornix dip. (c) Note the weak gradient magnitude where the fornix overlaps the CC. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

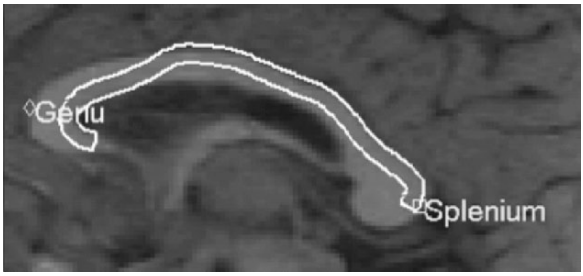


Figure 38. The CC organism's self-awareness enables it to identify landmark parts. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

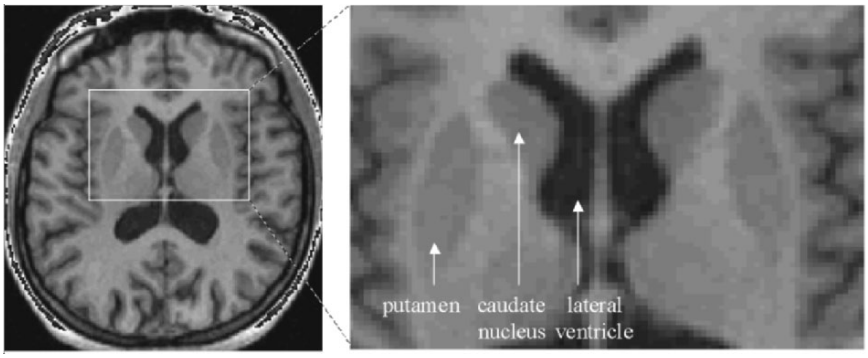


Figure 39. The lateral ventricle, caudate nucleus, and putamen shown in transversal brain MRI slice.

4.2. Simple Interacting Organisms

Simple interacting organisms were created to locate the lateral ventricles, caudate nuclei, and putamina in the left and right halves of transversal MR brain images (Figure 39). Since the ventricles are the most stable of the above structures, a ventricle organism is first released (Figure 40.1). It proceeds to locate the top of the ventricle (Figure 40.2) and its inner and outer (with respect to the brain) boundaries (Figure 40.3–5). Both ends of the ventricle organism are actively stretched to locate both the upper and the lower lobes of the ventricle (Figure 40.6). The ventricle organism then passes information about the shape and location of the segmented ventricle (Figure 40.7) to the caudate nucleus (CN) organism, which is initialized accordingly in a suitable position (Figure 40.8).

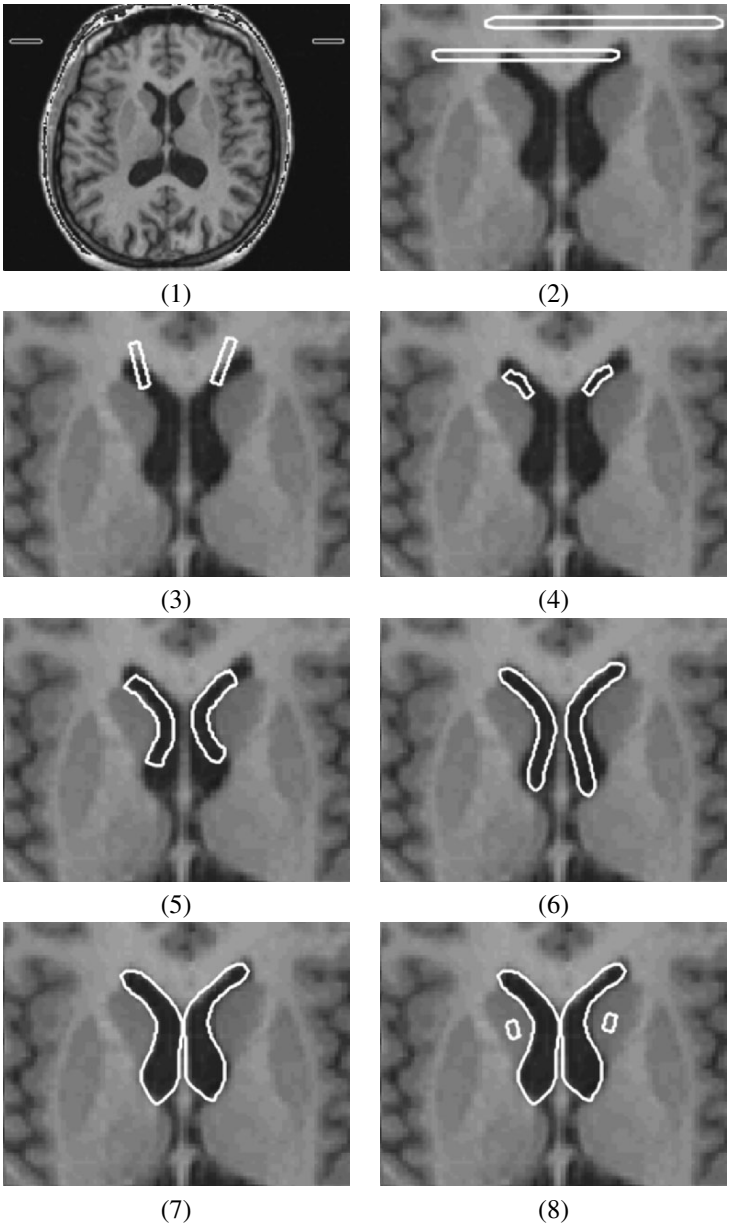


Figure 40. Deformable lateral ventricles (1–16), caudate nuclei (CN) (8–16), and putamina (11-16) organisms progressing through a sequence of behaviors to locate the corresponding structures in an MR brain image. Reprinted with permission from [85]. Copyright ©2002, Elsevier. (results continued on next page)

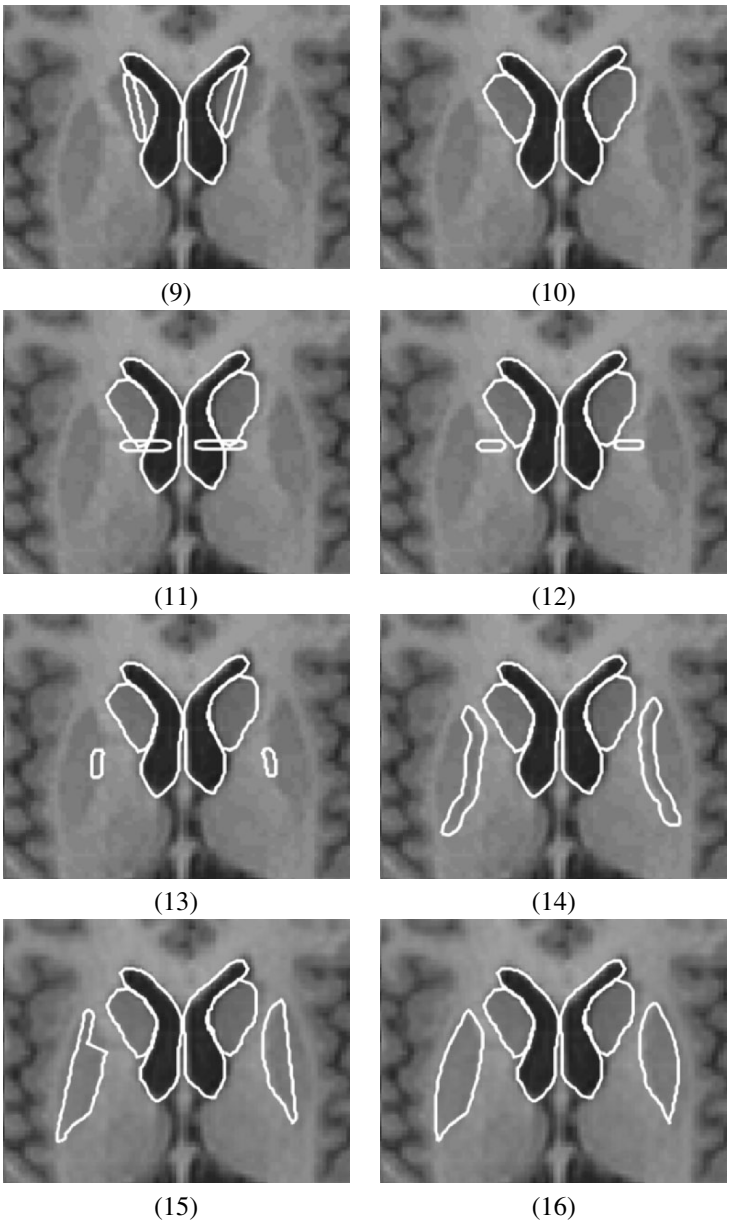


Figure 40. (continued). Results continued from previous page.

The CN organism segments the CN by stretching to locate its upper and lower limits (Figure 40.9) and thickening to latch onto its inner and outer boundaries (Figure 40.10). The CN organism passes information about the location of its lowest point (in the image) to the putamen organism, which is initialized accordingly (Figure 40.11). The putamen organism moves toward the putamen in the brain image (Figure 40.12) and then rotates and bends to latch onto the nearer putamen boundary (Figure 40.13). It then stretches and grows along the boundary until reaching the upper- and lower-most ends of the putamen (Figure 40.14), which identifies the medial axis of the putamen (Figure 40.15). Since the edges of the putamen boundary near the gray matter are usually weak, the organism activates an explicit search for an arc (parametrized only by one parameter controlling its curvature) that best fits the weak, sparse edge data in that region (Figure 40.16).

4.3. 2D Vessel Crawler

We also present the result of segmenting vessels in an angiogram (Figure 41) using a 2D artery crawler. Without proper constraints the vessel organism latches onto the wrong overlapping vessel (Figure 41a). However, adding additional sensors and high-level constraints enables the organism to distinguish between overlapping vessels and bifurcations (Figure 41b). When the latter is encountered, two new organisms are born from the original main branch organism, one for each branch (Figure 41c). Figure 32 demonstrates how this (overlap vs. bifurcate) decision strategy is implemented.

4.4. Physically Based Deformable CC Organism

In this section we exemplify the use of physics-based shape deformations (Section 3.3.3) within the deformable organisms framework yielding additional robustness by allowing intuitive real-time user guidance and interaction when necessary. The organism's behaviors and awareness of the different stages of the plan (Figure 1) enables it to not only segment the CC but also label anatomical regions (Figure 43a). Here we demonstrate the physics-based deformable organisms, with an underlying dynamic spring-mass mesh model, through the fully automatic segmentation and labeling of the CC in 2D midsagittal MRI slices. We also present further improvement of the segmentation results through minor, intuitive user interaction (Figure 43).

4.5. Caudate Nucleus and Lateral Ventricle Shape Modeling Using Medial Patches

We show two examples of medial patch-based deformations for fitting a 3D shape model to target brain structures. In the first example, we demonstrate the ability of performing intuitive and controlled manual 3D shape deformations to fit the medial patch representation to a lateral ventricle in a head MRI (Figure 44). We

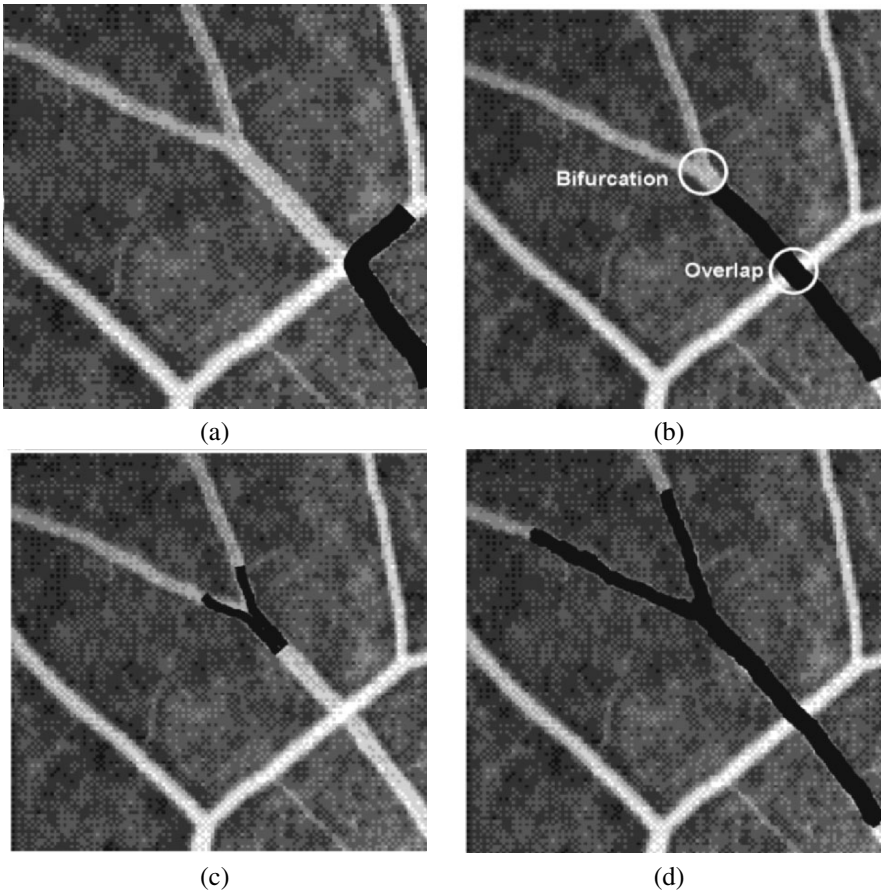


Figure 41. Segmenting vessels in an angiogram. (a) A deformable organism turning right and latching onto the wrong overlapping vessel. (b) High-level constraints enable the organism to differentiate between overlapping vessels and bifurcations. (c) Two new organisms are born upon identifying a bifurcation. (d) The segmented main vessel and its two branches. Reprinted with permission from [85]. Copyright ©2002, Elsevier.

then show an example of automatically fitting the 3D medial patch shape model to the binary image of a caudate nucleus from a 3D brain MRI (Figure 45). An initial medial sheet estimate is positioned at the plane spanned by the two main principal components of the locations of object points in the binary image. For each point in the medial sheet, rays are cast in both directions perpendicular to the medial sheet (along the third eigenvector) until they encounter an object boundary. The two boundary locations above and below the medial sheet are recorded. The nodes of the medial sheet are repositioned to be halfway between the top and

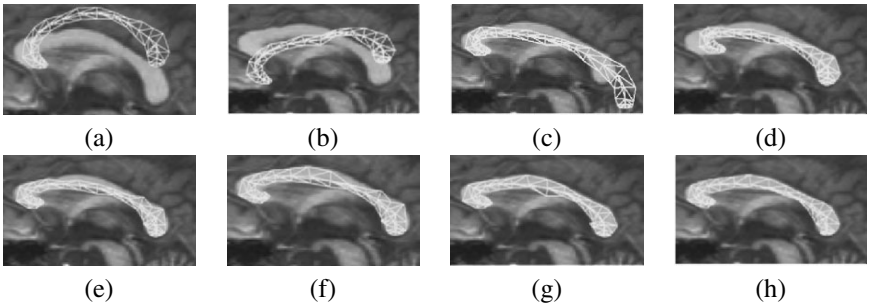


Figure 42. Progress of segmentation through its primary phases: (a) global model alignment, (b) model part alignment through (c) expansion and (d) contraction, (e) medial-axis alignment, (f) fitting to boundary, (g) detecting and (h) repairing fornix dip. Reprinted with permission from [80]. Copyright ©2005, SPIE. See attached CD for color version.

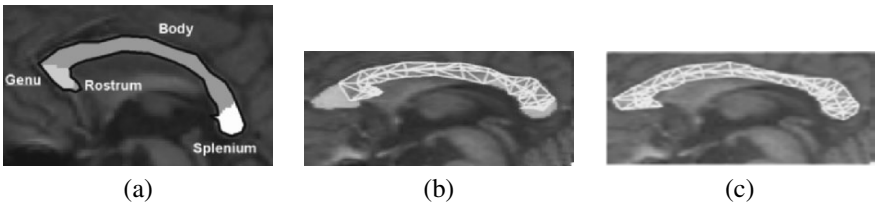


Figure 43. (a) Automatic labeling of important anatomical regions of the CC. (b) Before and (c) after intuitive manual intervention to improve the segmentation (red color corresponds to areas of erroneous segmentation). Reprinted with permission from [80]. Copyright ©2005, SPIE. See attached CD for color version.

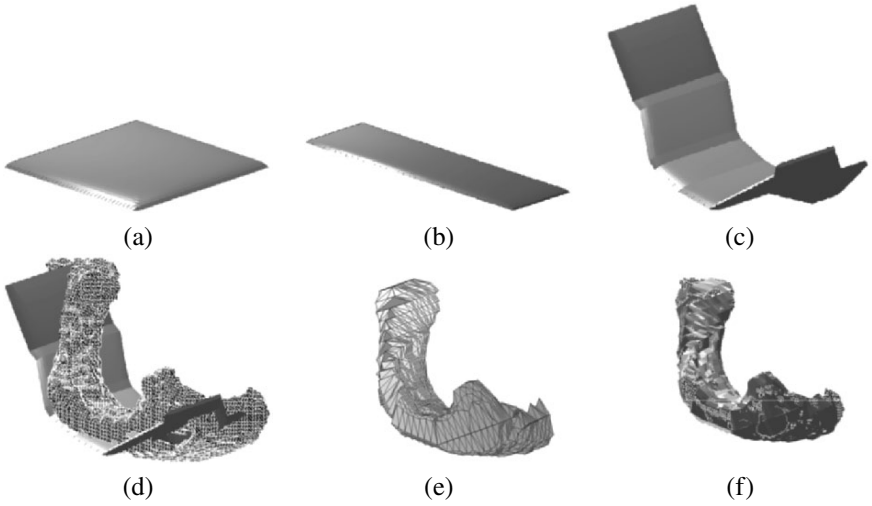


Figure 44. Fitting a 3D shape model to a lateral ventricle: (a) initial 3D model, (b) uniform shrinking along the x-axis, (c) bending deformations, (d) medial sheet bent into approximate bisecting position of target structure, (e) final 3D shape reconstructed from the medial sheets, and (f) overlay of 3D model on target structure.

bottom boundaries. Rays are now cast along vectors normal to the medial sheet until they encounter the boundaries. The procedure is iterated until the change in the locations of the medial nodes is too small.

4.6. 3D Vessel Crawler

In modern internal medicine, noninvasive imaging procedures are often crucial to the diagnosis of cardiovascular, pulmonary, renal, aortic, neurological, and abdominal diseases [82]. Common amongst the diagnosis of these diseases is the use of volumetric angiography to highlight branching vasculature. We present segmentation and analysis results of vasculature from volumetric magnetic resonance angiographic (MRA) data using the 3D vessel crawler. We ran the vessel crawler on an MRA image (Figure 46). The vessel crawler was initialized using a single seed point for each of the three root-most vessels. As shown, it was able to detect and track the vast majority of connected vessel segments. The crawler grows along vessels, latches onto their boundaries, detects bifurcations, spawns new child organisms, etc. All of these actions are described under the behavioral layer of the organism, and they rely upon both the physical and geometrical layers for implementation. For example, for crawling along the vessel the cognitive center gathers sensory input using the "vesselness" and "Hessian" sensory modules (Section 3.4), and elicits the acts of "growing" and then "fitting" to conform to the vascular walls (Section 3.5.2). In turn, each of these methods relies upon the physical and geometrical layers to carry out tasks, such as maintaining model stability through the application of stabilization springs.

In addition to their robust segmentation, deformable organisms are able to perform intuitive analysis, and labeling of the target structure. The 3D vessel crawler extracts clinically relevant features as it crawls though the vasculature, including the distance metric (DM), sum of angles metric (SOAM), and inflection count metric (ICM) [82]. In addition to those features the vessel crawlers are able to label branch points and vessel segments, determine branching angles, cross-sectional radius, vessel segment length and vessel volume, as well as highlight potential problem areas, the vascular regions they affect, and the shortest path to those locations from any target point (Figure 47).

5. SUMMARY

In this chapter we presented deformable organisms — an artificial life framework for medical image analysis. Deformable organisms extend the classical bottom–up, data-driven, deformable models to include additional top–down, knowledge-driven capabilities. This extension is realized by complementing the geometrical and physical layers of deformable models with behavioral and cognitive layers, as well as sensory modules.

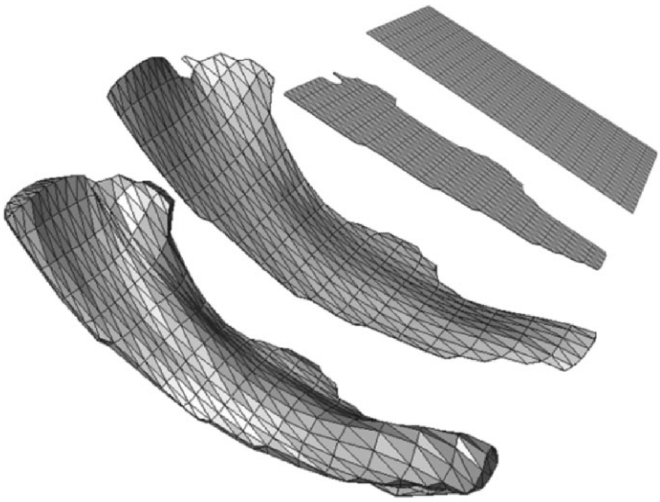


Figure 45. Caudate nucleus (CN) represented using a medial patch. Top to bottom: initial rectangular planar medial sheet, planar sheet cropped to match CN projection, curved medial sheet placed equidistant from the upper and lower CN boundaries. Thickness values are associated with each node in the medial sheet yielding a 3D surface of the CN.

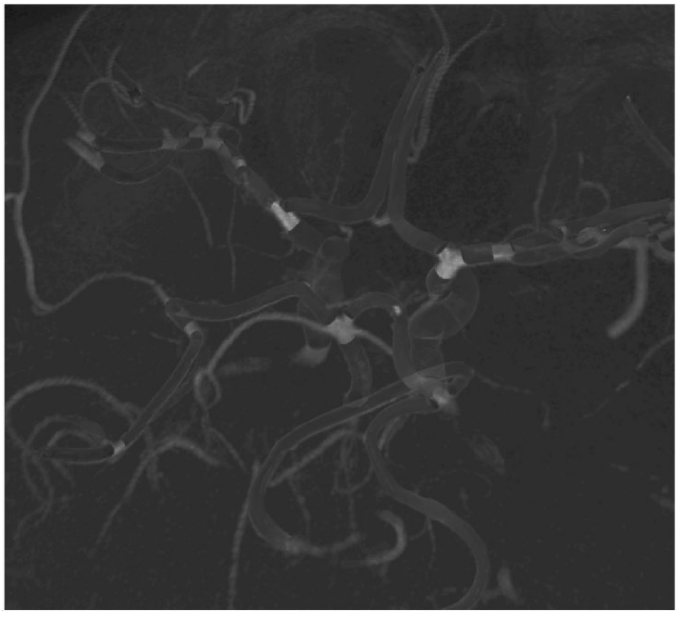


Figure 46. Maximum intensity projection rendering of an MRA showing the vessel crawler in orange. See attached CD for color version.

Figure 47. Three example representations of a vascular system: directed acyclic graph (left), a plot of the vessels with color corresponding to radial thickness (top), and a tree representing to the structure of a segmented vessel (bottom). See attached CD for color version.

The cognitive layer is the organism's brain, which is mainly responsible for decision-making based on sensory input, prior anatomical knowledge, a pre-stored segmentation plan, and interaction with other organisms. The cognitive layer controls the organism's perceptual capabilities by dynamically assigning on-board and off-board sensors. It also controls the sequence of behaviors that the organisms should adopt. The different behaviors in turn activate bodily deformations carried out through the lower geometrical and physical layers of the organism.

The need for top–down control of the shape deformations implies a requirement to develop shape representation techniques that respond to intuitive and controlled deformation commands (e.g., "move forward," "bend left"). To this end, we presented examples of pure geometrical and physically based shape representations that provide the desired deformation control capabilities. Specifically, we described the use of medial profiles/patches and 2D/3D deformable spring mass mesh models as examples of the geometric and physical layers of the deformable organism framework.

We presented examples of deformable organisms designed for different medical image analysis problems (e.g., segmentation of brain structures, analysis of vasculature) and highlighted their main behavioral routines, decision-making strategies, and sensory capabilities.

We believe the layered architecture of artificial life is a promising paradigm for medical image analysis, capable of incorporating state-of-the-art low-level image processing algorithms, high-level anatomical expert knowledge, and advanced planning and optimization strategies in a single modular design.

6. NOTES

1. See [28] as an example of previous work on, and motivation for, segmenting the corpus callosum.
2. Earlier mention of the use of AL in conjunction with segmentation appeared in [35, 36] However, as these methods rely on local, not global, decision-making, they strongly resemble traditional region-growing methods. For a survey of other applications of AL, see [37].

7. REFERENCES

1. Yoo TS. 2004. *Insight into images: principles and practice for segmentation*. Wellesley, MA: AK Peters Ltd.
2. Robb RA. 2000. *Biomedical imaging*. New York: Wiley-Liss.
3. Dhawan A. 2003. *Medical image analysis*. Wiley-IEEEE Press.
4. Bankman I. 2000. *Handbook of medical imaging: processing and analysis*. New York: Academic Press.
5. Sonka M, Fitzpatrick J. 2000. *Handbook of medical imaging*. Bellingham, WA: SPIE.
6. McInerney T, Kikinis R. 1998. An object-based volumetric deformable atlas for the improved localization of neuroanatomy in MR images. In *Proceedings of the first international conference on medical image computing and computer-assisted intervention (MICCAI'98)*. Lecture Notes in Computer Science, Vol. 1496, pp. 861–869. Berlin: Springer.
7. Shen D, Davatzikos C. 2000. An adaptive-focus deformable model using statistical and geometric information. *IEEE Trans Pattern Anal Machine Intell* **22**(8):906–913.
8. Tsotsos J, Mylopoulos J, Covvey H, Zucker S. 1980. A framework for visual motion understanding. *IEEE Trans Pattern Anal Machine Intell* **2**(6):563–573.
9. Draper BA, Hanson AR, Riseman EM. 1993. Learning blackboard-based scheduling algorithms for computer vision. *Int J Pattern Recognit Artif Intell* **7**(2):309–328.
10. Draper BA, Hanson AR, Riseman EM. 1996. Knowledge-directed vision: control, learning, and integration. *Proc IEEE Signals Symbols* **84**(11):1625–1637.
11. Crevier D, Lepage R. 1997. Knowledge-based image understanding systems: a survey. *Comput Vision Image Understand* **67**(2):161–185.
12. Strat T, Fischler M. 1991. Context-based vision: recognizing objects using information from both 2D and 3D imagery. *IEEE Trans Pattern Anal Machine Intell* **13**(10):1050–1065.
13. Poli R. 1996. Genetic programming for image analysis. In *Proceedings of the first international conference on genetic programming*, pp. 363–368. Ed JR Koza, DE Goldberg, DB Fogel, RL. Cambridge: MIT Press.
14. Martin MC. 2002. Genetic programming for real world robot vision. In *Proceedings of the international conference on intelligent robots and systems (IEEE/RSJ)*, pp. 67–72. Washington, DC: IEEE. Available at http://martincmartin.com/papers/GeneticProgramming ForRealWorldRobotVisionIROS2002Martin.pdf
15. Liu J, Tang Y. 1999. Adaptive image segmentation with distributed behavior-based agents. *IEEE Trans Pattern Anal Machine Intell* **21**(6):544–550.
16. Germond L, Dojat M, Taylor C, Garbay C. 2000. A cooperative framework for segmentation of MRI brain scans. *Artif Intell Med* **20**(1):77–93.
17. Boucher A, Doisy A, Ronot X, Garbay C. 1998. A society of goal-oriented agents for the analysis of living cells. *Artif Intell Med* **14**:183–199.
18. Rodin V, Harrouet F, Ballet P, Tisseau J. 1998. oRis: multiagents approach for image processing. In *Proceedings of the SPIE conference on parallel and distributed methods for image processing II*, Vol. 3452, 57–68. Ed S Hongchi, PC Coffield. Bellingham, WA: SPIE.

19. Bazen AM, van Otterlo M. 2001. A reinforcement learning agent for minutiae extraction from fingerprints. In *Proceedings of the Belgium–Netherlands Artificial Intelligence Conference (BNAIC'01)*, pp. 329–336. Ed B. Kröose, M de Rijke, G Schreiber, M van Someren. Washington, DC: IEEE Computer Society.
20. McInerney T, Terzopoulos D. 1996. Deformable models in medical image analysis: a survey. *Med Image Anal* **1**(2):91–108.
21. Montagnat J, Delingette H, Ayache N. 2001. A review of deformable surfaces: topology. *Image Vision Comput* **19**(14):1023–1040.
22. Osher S, Paragios N. 2003. *Geometric level set methods in imaging vision and graphics*. Berlin: Springer.
23. Caselles V, Kimmel R, Sapiro G. 1997. Geodesic active contours. *Int J Comput Vision* **22**(1):61–79.
24. Sethian JA. 1996. *Level set methods: evolving interfaces in geometry, computer vision and material sciences*. Cambridge: Cambridge UP.
25. Cootes TF, Cooper D, Taylor CJ, Graham J. 1995. Active shape models: their training and application. *Comput Vision Image Understand* **61**(1):38–59.
26. Cootes T, Beeston C, Edwards G, Taylor C. 1999. A unified framework for atlas matching using active appearance models. In *Proceedings of the 17th international conference on image processing in medical imaging (IPMI'99)*, pp. 322–333. Berlin: Springer.
27. Szekely G, Kelemen A, Brechbuehler Ch, Gerig G. 1996. Segmentation of 3D objects from MRI volume data using constrained elastic deformations of flexible Fourier surface models. *Med Image Anal* **1**(1):19–34.
28. Lundervold A, Duta N, Taxt T, Jain A. 1999. Model-guided segmentation of corpus callosum in MR images. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition*, Vol. 1, pp. 231–237. Washington, DC: IEEE Computer Society.
29. Cohen LD. 1991. On active contour models and balloons. *Comput Vision Graphics Image Process: Image understand* **53**(2):211–218.
30. Xu C, Prince J. 1998. Snakes, shapes, and gradient vector flow. *IEEE Trans Image Process* **7**(3):359–369.
31. Cootes TF, Edwards GJ, Taylor CJ. 2001. Active appearance models. *IEEE Trans Pattern Anal Machine Intell* **23**(1):681–685.
32. Leventon M, Grimson W, Faugeras O. 2000. Statistical shape influence in geodesic active contours. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition*, Vol. 1. pp. 316–323. Washington, DC: IEEE Computer Society.
33. Warfield SK, Kaus M, Jolesz FA, Kikinis R. 2000. Adaptive, template moderated, spatially varying statistical classification. *Med Image Anal* **4**(1):43–55.
34. Hamarneh G, Gustavsson T. 2000. Statistically constrained snake deformations. In *Proceedings of the international conference on systems, man, and cybernetics*, Vol. 3, pp. 1610–1615. Washington, DC: IEEE Computer Society.
35. Choi C, Wirth M, Jennings A. 1997. Segmentation: artificial life and watershed transform approach. In *Sixth international conference on image processing and its applications*, Vol. 1, pp. 371–375. Washington, DC: IEEE Computer Society.
36. Kagawa H, Kinouchi M, Hagiwara M. 1999. Image segmentation by artificial life approach using autonomous agents. In *Proceedings of the international joint conference on neural networks*, Vol. 6, pp. 4413–4418. Washington, DC: IEEE Computer Society.
37. Kim K-J, Cho S-B. 2006. A comprehensive overview of the applications of artificial life. *Artif Life* **12**(1):153–182.
38. Hamarneh G, McInerney T, Terzopoulos D. 2001. Deformable organisms for automatic medical image analysis. In *Proceedings of the fourth international conference on medical image computing and computer-assisted intervention (MICCAI'01)*. Lecture Notes in Computer Science, Vol. 2208, pp. 66–75. Berlin: Springer.
39. Terzopoulos D. 1999. Artificial life for computer graphics. *Commun ACM* **42**(8):32–42

40. Bookstein F. 1997. *Morphometric tools for landmark data: geometry and biology*. Cambridge: Cambridge UP.
41. Costa L, Cesar Jr R. 2000. *Shape analysis and classification: theory and practice*. Boca Raton, FL: CRC Press.
42. Dryden I, Mardia K. 1998. *Statistical shape analysis*. New York: John Wiley and Sons.
43. Lachaud J, Montanvert A. 1999. Deformable meshes with automated topology changes for coarse-to-fine three-dimensional surface extraction. *Med Image Anal* **3**(1):1–21.
44. Mandal C, Vemuri B, Qin H. 1998. A new dynamic FEM-based subdivision surface model for shape recovery and tracking in medical images. In *Proceedings of the first international conference on medical image computing and computer-assisted intervention (MICCAI'98)*. Lecture Notes in Computer Science, Vol. 1496, pp. 753–760. Berlin: Springer.
45. Miller J, Breen D, Lorensen W, O'Bara R, Wozny MJ. 1991. Geometrically deformed models: a method for extracting closed geometric models from volume data. In *Proceedings of the 18th annual conference on computer graphics and interactive techniques (SIGGRAPH'91)*, Vol. 25, pp. 217–226. New York: ACM Press.
46. Montagnat J, Delingette H. 1997. Volumetric medical image segmentation using shape constrained deformable models. In *Proceedings of the second international conference on computer vision, virtual reality and robotics in medicine (CVRMed-MRCAS'97)*. Lecture Notes in Computer Science, Vol. 1205, pp. 13–22. Berlin: Springer.
47. Barr A. 1984. Global and local deformations of solid primitives. In *Proceedings of the 11th annual conference on computer graphics and interactive techniques (SIGGRAPH'84)*, Vol. 18, pp. 21–30. New York: ACM Press.
48. Coquillart S. 1990. Extended free form deformations: a sculpting tool for 3d geometric modeling. In *Proceedings of the 17th annual conference on computer graphics and interactive techniques (SIGGRAPH'90)*, Vol. 24, pp. 187–196. New York: ACM Press.
49. Sederberg TW, Parry SR. 1986. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on computer graphics and interactive techniques (SIGGRAPH'86)*, Vol. 4, pp. 151–160. New York: ACM Press.
50. Singh K, Fiume E. 1998. Wires: a geometric deformation technique. In *Proceedings of the 25th annual conference on computer graphics and interactive techniques (SIGGRAPH'98)*, Vol. 99, No. 1, pp. 405–414. New York: ACM Press.
51. Terzopoulos D, Metaxas D. 1991. Dynamic 3D models with local and global deformations: deformable superquadrics. *IEEE Trans Pattern Anal Machine Intell* **13**(7):703–714.
52. Burel G, Henocq H. 1995. Three-dimensional invariants and their application to object recognition. *Signal Process* **45**(1):1–22.
53. Davatzikos C, Tao X, Shen D. 2003. Hierarchical active shape models: using the wavelet transform. *IEEE Trans Med Imaging* **22**(3)414–423.
54. McInerney T, Terzopoulos D. 1995. A dynamic finite-element surface model for segmentation and tracking in multidimensional medical images with application to cardiac 4D image analysis. *Comput Med Imaging Graphics* **19**(1):69–83.
55. Mortenson M. 1997. *Geometric modeling*. New York: Wiley.
56. Blum H. 1973. Biological shape and visual science. *Theor Biol* **38**:205–287.
57. Attali D, Montanvert A. 1997. Computing and simplifying 2D and 3D continuous skeletons. *Comput Vision Image Understand* **67**(3):261–273.
58. Borgefors G, Nystrom I, Baja GSD. 1999. Computing skeletons in three dimensions. *Pattern Recognit* **32**:1225–1236.
59. Bouix S, Dimitrov P, Phillips C, Siddiqi K. 2000. Physics-based skeletons. In *Proceedings of the vision interface conference*, pp. 23–30.
60. Dimitrov P, Damon J, Siddiqi K. 2003. Flux invariants for shape. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR'03)*, pp. 835–841. Washington, DC: IEEE Computer Society.

61. Fritsch D, Pizer S, Yu L, Johnson V, Chaney E. 1997. Segmentation of medical image objects using deformable shape loci. In *Proceedings of the 15th international conference on information processing in medical imaging (IPMI'97). Lecture notes in computer science*, Vol. 1230, pp. 127–140. Berlin: Springer.
62. Hamarneh G, McInerney T. 2003. Physics-based shape deformations for medical image analysis. In *Proceedings of the 2nd SPIE conference on image processing: algorithms and systems (SPIE-IST'03)*, Vol. 5014, pp. 354–362. Ed ER Dougherty, JT Astola, KO Egiazarian.
63. Hamarneh G, Abu-Gharbieh R, McInerney T. 2004. Medial profiles for modeling deformation and statistical analysis of shape and their use in medical image segmentation. *Int J Shape Model* **10**(2)187–209.
64. Leymarie F, Levine MD. 1992. Simulating the grassfire transform using an active contour model. *IEEE Trans Pattern Anal Machine Intell* **14**(1):56–75.
65. Pizer S, Fritsch D. 1999. Segmentation, registration, and measurement of shape variation via image object shape. *IEEE Trans Med Imaging* **18**(10):851–865.
66. Pizer S, Gerig G, Joshi S, Aylward SR. 2003. Multiscale medial shape-based analysis of image objects. *Proc IEEE* **91**(10):1670–1679.
67. Sebastian T, Klein P, Kimia B. 2001. Recognition of shapes by editing shock graphs. In *Proceedings of the international conference on computer vision (ICCV'01)*, pp. 755–762. Washington, DC: IEEE Computer Society.
68. Siddiqi K, Bouix S, Tannenbaum A, Zucker S. 2002. Hamilton-Jacobi skeletons. *Int J Comput Vision* **48**(3):215–231.
69. Duta N, Sonka M, Jain A. 1999. Learning shape models from examples using automatic shape clustering and Procrustean analysis. In *Proceedings of the 17th international conference on information processing in medical imaging (IPMI'99)*. Lecture Notes in Computer Science, Vol. 1613, pp. 370–375.
70. Yushkevich P, Joshi S, Pizer SM, Csernansky J, Wang L. 2003. Feature selection for shape-based classification of biological objects. In *Proceedings of the 17th international conference on information processing in medical imaging (IPMI'03)*. Lecture Notes in Computer Science, Vol. 2732, pp. 114–125.
71. Styner M, Gerig G, Lieberman J, Jones D, Weinberger D. 2003. Statistical shape analysis of neuroanatomical structures based on medial models. *Med Image Anal* **7**(3):207–220.
72. Lu C, Pizer S, Joshi S. 2003. A Markov random field approach to multiscale shape analysis. In *Proceedings of the conference on scale space methods in computer vision.* Lecture Notes in Computer Science, Vol. 2695, pp. 416–431. Available at http://midag.cs.unc.edu/pubs/papers/ScaleSpace03_Lu_shape.pdf.
73. Grenander U. 1963. *Probabilities on algebraic structures*. New York: Wiley.
74. Fletcher P, Conglin L, Pizer S, Joshi S. 2004. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Trans Med Imaging* **23**(8):995–1005.
75. Bill JR, Lodha SK. 1995. Sculpting polygonal models using virtual tools. In *Proceedings of the conference on the graphics interface*, pp. 272–279. New York: ACM Press.
76. O'Donnell T, Boult T, Fang X, Gupta A. 1994. The extruded generalized cylinder: a deformable model for object recovery. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR'94)*, pp. 174–181. Washington, DC: IEEE Computer Society.
77. Perona P, Malik J. 1990. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans Pattern Anal Machine Intell* **12**(7)629–639.
78. Weickert J. 1998. *Anisotropic diffusion in image processing*. ECMI Series. Stuttgart: Teubner.
79. Frangi AF, Niessen WJ, Vincken KL, Viergever MA. 1998. Multiscale vessel enhancement filtering. In *Proceedings of the first international conference on medical image computing and computer-assisted intervention (MICCAI'98)*. Lecture Notes in Computer Science, Vol. 1496, pp. 130–137. Berlin: Springer.

80. Hamarneh G, McIntosh C. 2005. Physics-based deformable organisms for medical image analysis. In *Proceedings of the SPIE conference on medical imaging: image processing*, Vol. 5747, pp. 326–335. Bellingham, WA: SPIE.
81. Kimme C, Ballard DH, Sklansky J. 1975. Finding circles by an array of accumulators. *Commun Assoc Comput Machinery* **18**:120–122.
82. Bullitt E, Gerig G, Aylward SR, Joshi SC, Smith K, Ewend M, Lin W. 2003. Vascular attributes and malignant brain tumors. In *Proceedings of the sixth international conference on medical image computing and computer-assisted intervention (MICCAI'03)*. Lecture Notes in Computer Science, Vol. 2878, pp. 671–679. Berlin: Springer.
83. Terzopoulos D. 1986. *On matching deformable models to images*. Technical Report 60, Schlumberger Palo Alto Research. Reprinted in *Topical Meeting MachineVision, Technical Digest Series* **12**:160–167 (1987).
84. Kass M, Witkin A, Terzopoulos D. 1987. Snakes: active contour models. *Int J Comput Vision* **1**(4):321–331.
85. McInerney T, Hamarneh G, Shenton M, Terzopoulos D. Deformable organisms for automatic medical image analysis. *J Med Image Anal* **6**(3):251–266.
86. C. McIntosh and G. Hamarneh, Vessel Crawlers: 3D Physically-based Deformable Organisms for Segmentation and Analysis of Tubular Structures in Medical Images. *IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 1084–1091.

13

PDE-BASED THREE DIMENSIONAL PATH PLANNING FOR VIRTUAL ENDOSCOPY

M. Sabry Hassouna and Aly A. Farag

Computer Vision & Image Processing Laboratory (CVIP)
University of Louisville, Louisville, Kentucky, USA

Robert Falk

Department of Medical Imaging, Jewish Hospital
Louisville, Kentucky, USA

Three-dimensional medial curves ($\mathcal{MC}$) are an essential component of any virtual endoscopy (VE) system, because they serve as flight paths for a virtual camera to navigate the human organ and to examine its internal views. In this chapter, we propose a novel framework for inferring stable continuous flight paths for tubular structures using partial differential equations (PDEs). The method works in two passes. In the first pass, the overall topology of the organ is analyzed and its important topological nodes identified. In the second pass, the organ's flight paths are computed by tracking them starting from each identified topological node. The proposed framework is robust, fully automatic, computationally efficient, and computes medial curves that are centered, connected, thin, and less sensitive to boundary noise. We have extensively validated the robustness of the proposed method both quantitatively and qualitatively against several synthetic 3D phantoms and clinical datasets.

Address all correspondence to: Dr. Aly A. Farag, Professor of Electrical and Computer Engineering, University of Louisville, CVIP Lab, Room 412, Lutz Hall, 2301 South 3rd Street, Louisville, KY 40208, USA. Phone: (502) 852-7510, Fax: (502) 852-1580. aafara01@louisville.edu.

1. INTRODUCTION

Unlike conventional medical visualization technology, which focuses mainly on the whole picture of a human organ with special interest in the organ's outer appearance, virtual endoscopy (VE) shifts attention to the inner appearance of an organ by integrating medical imaging with virtual reality. VE is a computer-based alternative to true fiber optic endoscopy (TE) for screening hollow organs. It has evolved rapidly over the past decade because of the great advances in manufacturing high-resolution helical spiral computed tomography (CT). VE has many advantages, such as being less invasive, more cost-effective, free of risks and side effects (e.g., perforation and infection), and easily tolerated by the patient. VE is not intended to replace TE, but rather to complement it by providing additional supportive information. For example, VE allows the visualization of neighboring structures outside the screened organ, and hence can assist in pathology localization, allows viewing in forward and reverse directions, visualizes areas that are hard to reach by TE, has the ability to pass high-grade stenoses, and finally is the only alternative offered to those patients who either refuse TE or are severely ill [1, 2]. In general, VE is ideal for screening purposes and surgical planning. It has been under development in many clinical areas, such as virtual colonoscopy, virtual bronchoscopy, virtual angioscopy, and others [3–10].

VE involves three major steps: organ segmentation, flight path generation, and rendering the internal views of an organ. The computation of 3D flight paths of volumetric objects is still a challenging process due to the complex nature of the anatomical structures.

2. PREVIOUS WORK

The 3D flight path generation methods can be categorized as follows: topological thinning methods, distance transform methods, and potential field methods.

2.1. Thinning Methods

The basic idea behind thinning methods is to iteratively peel off the object's boundary by identifying simple points whose removal does not alter its topology. To prevent over-thinning of the object, deletion of its endpoints must be avoided. Thinning methods usually extract $\mathcal{MC}$ by either checking the voxel's local neighborhood against a set of templates that prevents the deletion of topological nodes such as branch and endpoints [11, 12], or by pruning medial surfaces [5, 6] while preventing removal of junctions between surfaces.

Sequential detection and removal of simple points is a time-consuming process. To remedy this limitation, parallel and directional thinning methods have been introduced. In parallel thinning methods [5, 13–15], multiple voxels are detected and removed simultaneously per iteration using deletion templates that

are designed to preserve connectivity and topology of the objects. Unfortunately, these methods rely on local information to obtain skeleton voxels, and hence are sensitive to boundary noise and local distortion. On the other hand, in directional thinning methods [16–18] voxels are removed only from one particular direction in each pass. The methods use different numbers of directions and conditions to identify endpoints. As a consequence, the resultant $\mathcal{MC}$ may not be centered with respect to the object's boundary because it becomes sensitive to the order in which directions are processed.

Thinning methods produce connected $\mathcal{MC}$ that preserve object topology. However, under certain configurations of voxels it is possible to introduce an unwanted hole by deleting a voxel that does not generate a local connectivity violation. Although it is possible to detect introduction of a hole, the test requires exhaustive analysis of the local neighborhood surrounding the candidate voxel to be deleted, and hence is very time consuming. In addition, preserving endpoints leads to many undesired small branches.

2.2. Distance Transform Methods

The distance transform $D(\boldsymbol{x})$ computes at each voxel $\boldsymbol{x}$ its minimum distance from the object's boundary. $D(\boldsymbol{x})$ is normally a distance metric function such as a Manhattan distance, Chessboard distance, or Euclidean distance, which can be discretely approximated using a chamfer metric (3,4,5) [19], or continuously approximated using the fast marching methods [20]. The ridge voxels (singularities) of the distance field are locally centered with respect to the object's boundary; therefore, centeredness is guaranteed. Any distance transform method involves three steps: distance field generation, detection of ridge voxels, and reconnection of identified ridges into medial curves.

Gagvani et al. [21] provided a method to localize voxels with maximal spheres. A thickness parameter was employed to control the thickness of the generated skeletons. Although the method is fast and captures the topology of the object, it favors reconstruction over connectivity. Therefore, the generated skeleton forms a disconnected set of unordered voxels.

Zhou and Toga [3] proposed a voxel coding technique in which a discrete wavefront propagates the whole object starting from a manually selected reference point. The wave divides the object into clusters that are approximately normal to the $\mathcal{MC}$. Initially, $\mathcal{MC}$ are extracted as trajectories and then centered using the Medial Point Replacement (MPR) method. MPR defines the cluster center as the voxel of maximum distance from the boundary. The algorithm guarantees connected paths. However, as the complexity of the object increases, the clusters are no longer normal to the $\mathcal{MC}$, and hence centeredness is not guaranteed, especially near branching nodes.

Bitter et al. [4] proposed a penalized-distance algorithm to extract $\mathcal{MC}$, which is a modification of a previous work [22, 23]. The algorithm first uses a distance

field and the Dijkstra algorithm [24] to locate an initial $\mathcal{MC}$, which is then refined iteratively by discarding its redundant voxels. The penalty factor is specified heuristically by the user for each dataset, preventing the algorithm from automation. In addition, the method requires modification to handle complex tree structures.

Paik et al. [25] suggested a different approach for extracting $\mathcal{MC}$, where a geodesic path is found on the surface of a tubular structure and is later centered by applying an iterative thinning procedure. The algorithm is sensitive to boundary noise, difficult to apply to complex tubular structures, and requires user interaction.

Siddiqi et al. [26] extracted $\mathcal{MC}$ in two steps. Initially, medial surfaces were computed by combining a thinning algorithm with the average outward flux measure to distinguish medial voxels from others. Then they applied an ordered thinning method [27] based on the Euclidean distance field to extract $\mathcal{MC}$. They presented very good results for vascular trees. The method requires a postprocessing step to clean unwanted branches caused by thinning as well as to convert the computed skeleton into a graph for diagnoses and navigation.

Telea and Vilanova [28] resampled the volumetric object in three directions—x, y, and z—which results in three separate stacks of cross-sections for the same object. Then, they apply a level set-based centerline extraction method to compute the medial points of each 2D cross-section in each volume independently. Finally, the resulting sets of medial points are merged (intersected) to yield the final skeleton. The algorithm involves a number of heuristics with no theoretical justification. Also, the convergence of their method is not guaranteed because the actual topology of a 3D object cannot be interpreted from its 2D cross-sections in different directions.

The work by Deschamps et al. [29] is the most closely related work to the one presented in this chapter. The user selects the endpoints of a branch of interest, and then a fast marching front propagates from the starting point with a speed determined by a scalar potential that depends upon location in the medium until it reaches the endpoint. Minimal paths are extracted using gradient descent methods. They presented very nice results for single tube structures such as the colon. However, for tree structures their method generates trajectories near branching nodes.

2.3. Potential Field Methods

In these type of methods, the Euclidean distance field is replaced with another continuous function that is computed over the domain of the object, and then the medial curve voxels are identified as the field extremum points.

Chaung et al. [30] computed the Newtonian potential field over the object domain. Then they derived an analytical form for the gradient of the potential field, which is then used to extract $\mathcal{MC}$ by tracing them along the lines of field force starting from each vertex of the object surface until a minimum is reached (medial points). Since the method is limited only to polygonal data, each vertex

in the object is assumed to start a new $\mathcal{MC}$. The resulting skeleton is usually disconnected and requires postprocessing for reconnection. Although generalizing the method to handle non-polygonal data is possible, computation of the potential function becomes very expensive because the potential value at an internal voxel of the object is a function of all the voxels of the boundary. Also, for a noisy boundary the method will generate a large number of unwanted branches since no method is presented to identify the starting points of each $\mathcal{MC}$.

Cornea et al. [31] extended [30] to handle non-polygonal data, they computed the critical points and low divergence points of the field force as well as the high-curvature boundary points to identify the starting points of $\mathcal{MC}$. Again, computation of the potential field is still very expensive. The method takes a half an hour for computing the potential of 200^3 objects. Also, identification of critical field points may result in either an over- or underestimate of the true numbers, and hence inaccurate identification of the exact topological nodes of the organ.

Ma et al. [32] used radial basis functions (RBFs) to build a new potential field, and then applied a gradient descent algorithm to locate local extremes in RBF (branching and end nodes). Constructing an RBF to preserve the shape property of an arbitrary 3D model is still worth more consideration.

Wu et al. [33] regarded the 3D model faces as charged planes. Seed points with negative charges at the model vertices are initiated, which are then pushed toward local minimum positions by electric static force. The local minimum positions are connected to complete the skeleton. This method is very computationally expensive and requires user interaction to remove undesired connection.

3. LIMITATION OF EXISTING METHODS

Existing 3D flight path extraction techniques suffer from at least one of the following limitations: $\mathcal{MC}$ are extracted from medial surfaces by different pruning methods, and so accuracy of the former is a function of extraction of the latter; manual interaction is required to select the starting point of each medial curve, which is computationally expensive; heuristics are required to handle branching nodes, dedicated to a specific application, and hence there is a lack of generality and a sensitivity to boundary noise.

4. THE PROPOSED MEDIAL CURVE EXTRACTION FRAMEWORK

Consider the *minimum-cost path problem* that finds the path $C(s) : [0, \infty) \longrightarrow \mathbb{R}^n$ that minimizes cumulative travel cost from starting point A to some destination B in $\mathbb{R}^n$. If the cost U is only a function of the location $\boldsymbol{x}$ in the given domain, then the cost function is called isotropic, and the minimum cumulative cost at $\boldsymbol{x}$

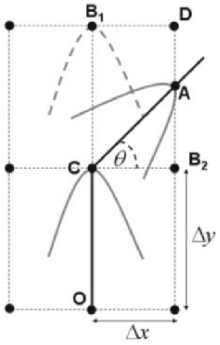


Figure 1. The dark line is a medial curve $\mathcal{MC}$, while light gray lines are wavefronts. Lattice voxels are represented by solid dots.

is defined as

$$T(\boldsymbol{x}) = \min_{C_{A\boldsymbol{x}}} \int_0^L U(C(s))ds, \tag{1}$$

where $C_{A\boldsymbol{x}}$ is the set of all paths linking A to $\boldsymbol{x}$. The path length is L, and the starting and ending points are $C(0) = A$ and $C(L) = \boldsymbol{x}$, respectively. The path that gives minimum integral is the minimum cost path. In geometrical optics, it has been proved that the solution of Eq. (1) is eikonal equation Eq. (2):

$$|\nabla T(\boldsymbol{x})|F = 1, \tag{2}$$

where $T(\boldsymbol{x})$ is the minimum arrival time of a wave as it crosses a point $\boldsymbol{x}$, and the speed is given by the reciprocal of the cost function:

$$F(\boldsymbol{x}) = 1/U(\boldsymbol{x}). \tag{3}$$

In this work we derive a new speed function such that the minimum cost path between two medial points is a medial curve.

4.1. Derivation of Speed Function

For clarity, let's first derive the speed function for a 2D grid lattice and then generalize it to a 3D lattice. Let P_S be a medial voxel that belongs to the medial curves of the object. Assume that P_S is a point source that transmits wavefronts at speed $F(\boldsymbol{x})$ inside the object. The goal is to find the speed $F(x)$ that makes the wavefront faster only at medial voxels such that $\mathcal{MC}$ intersect the propagating fronts at those voxels of maximum positive curvatures.

Consider the medial curve $\mathcal{MC}$ whose medial voxels are O, C, and A, as shown in Figure 1. Let B_i be non-medial voxels. Assume that the front reaches C, A, and B_i at time t_0, $t_0 + \Delta t_A$, and $t_0 + \Delta t_{B_i}$, respectively. The goal is to let the wave reach A before reaching B_i even if $d(C, A) > d(C, B_i)$, where $d(.,.)$ is the Euclidean distance. If the goal is not satisfied, then $t_{B_i} < t_A$ and the front that reaches B_i will have a large positive curvature, which means that B_i is wrongly classified as a medial voxel.

$$t_A < t_{B_i} \tag{4}$$

$$\frac{d(C,A)}{F(A)} < \frac{d(C,B_i)}{F(B_i)}. \tag{5}$$

Let $F(\boldsymbol{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^+$ be given by

$$F(\boldsymbol{x}) = g(\lambda(\boldsymbol{x})), \tag{6}$$

where $\lambda(\boldsymbol{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^+$ is a medialness function that assigns medial voxels higher weight than non-medial ones. Let

$$\lambda(A) = h \tag{7}$$

$$\lambda(B_i) = h - \delta, \tag{8}$$

where δ is the difference between λ of two neighboring voxels $\boldsymbol{x}$ and $\boldsymbol{y}$, where $\lambda(\boldsymbol{x}) > \lambda(\boldsymbol{y})$. Let Δx and Δy be the grid spacing in the x and y directions, respectively, and θ the angle between $\mathcal{MC}$ and the horizontal, as shown in Figure 1. For $0 \leq \theta \leq tan^{-1}(\Delta y/\Delta x)$,

$$d(C, A) = \frac{\Delta x}{\cos(\theta)}, \tag{9}$$

$$d(C, B_i) = \min(\Delta x, \Delta y), \tag{10}$$

and hence Eq. (4) yields

$$\frac{\Delta x}{\min(\Delta x, \Delta y)\ \cos(\theta)} < \frac{g(h)}{g(h-\delta)}. \tag{11}$$

In the *worst case* when $\mathcal{MC}$ makes θ_{max}, the wavefront will travel a longer distance to reach A than to reach B_i:

$$\cos(\theta_{max}) = \frac{\Delta x}{\sqrt{\Delta^2 x + \Delta^2 y}}. \tag{12}$$

Note that $\theta_{max} = 45$ when $\Delta x = \Delta y$. Substituting Eq. (12) into Eq. (11),

$$\frac{\sqrt{\Delta^2 x + \Delta^2 y}}{\min(\Delta x, \Delta y)} < \frac{g(h)}{g(h-\delta)}. \tag{13}$$

Let

$$r = \frac{\sqrt{\Delta^2 x + \Delta^2 y}}{\min(\Delta x, \Delta y)}; \tag{14}$$

then

$$rg(h-\delta) - g(h) < 0. \tag{15}$$

Applying Taylor series expansion to $g(h-\delta)$ and substituting in Eq. (15),

$$r\left(g(h) - \delta\frac{dg(h)}{dh} + \epsilon(\delta)\right) - g(h) < 0, \tag{16}$$

where

$$\epsilon(\delta) = \sum_{k=2}^{\infty}(-1)^k \frac{\delta^k}{k!}\frac{d^k g}{dh^k}. \tag{17}$$

Assume for now that $\epsilon(\delta) \to 0$ is very small, such that it can be ignored without affecting inequality (16). Later we will find the condition that satisfies the assumption

$$r\left(g(h) - \delta\frac{dg(h)}{dh}\right) - g(h) < 0. \tag{18}$$

By rearranging Eq. (18),

$$\frac{dg(h)}{g(h)} > \frac{r-1}{r\delta}dh. \tag{19}$$

Let

$$\alpha_1 = \frac{r-1}{r\delta} = \frac{1}{\delta}\left(1 - \frac{\min(\Delta x, \Delta y)}{\sqrt{\Delta^2 x + \Delta^2 y}}\right) > 0. \tag{20}$$

By integrating both sides of Eq. (19) for a given $\boldsymbol{x}$, we get

$$\int_{\lambda_{\min}}^{\lambda(\boldsymbol{x})} \frac{dg(h)}{g(h)} > \int_{\lambda_{\min}}^{\lambda(\boldsymbol{x})} \alpha_1 dh \tag{21}$$

$$\ln g(\lambda(\boldsymbol{x})) - \ln g(\lambda_{\min}) > \alpha_1(\lambda(\boldsymbol{x}) - \lambda_{\min}) \tag{22}$$

$$\ln F(\boldsymbol{x}) > \alpha_1\lambda(\boldsymbol{x}) - (\alpha_1\lambda_{\min} - \ln F_{\min}), \tag{23}$$

where $F_{\min} = g(\lambda_{\min})$ is the minimum speed value. Let

$$\zeta = \alpha_1\lambda_{\min} - \ln F_{\min}. \tag{24}$$

Then,

$$F(\boldsymbol{x}) > \exp(\alpha_1\lambda(\boldsymbol{x}) - \zeta) = \exp(-\zeta)\exp(\alpha_1\lambda(\boldsymbol{x})). \tag{25}$$

There are an infinite number of speed functions that satisfy Eq. (25), among which we pick the one that allows us to neglect the effect of $\epsilon(\delta)$ without altering inequality Eq. (16).

For some $\alpha_2 > 0$, one possible choice for $F(\boldsymbol{x})$ is

$$F(\boldsymbol{x}) = \exp(\alpha_2 \lambda(\boldsymbol{x})) \exp(\alpha_1 \lambda(\boldsymbol{x})), \tag{26}$$

which requires,

$$\begin{aligned} \exp(\alpha_2 \lambda(\boldsymbol{x})) &> \exp(-\zeta), & (27)\\ \alpha_2 \lambda(\boldsymbol{x}) &> \ln F_{\min} - \alpha_1 \lambda_{\min}. & (28) \end{aligned}$$

By restricting the minimum value of the speed to unity, the equation always holds. Therefore, Eq. (26) reduces to

$$F(\boldsymbol{x}) = \exp(\alpha \lambda(\boldsymbol{x})), \tag{29}$$

where $\alpha = \alpha_1 + \alpha_2$.

Now, let's find the condition that satisfies $\epsilon(\delta) \to 0$. By substituting the proposed speed function, Eq. (29), into Eq. (16),

$$r\left(\exp(\alpha h) - \delta\alpha \exp(\alpha h) + \frac{\delta^2\alpha^2}{2!}\exp(\alpha h) - \cdots\right) - \exp(\alpha h) < 0 \tag{30}$$

$$r \exp(\alpha h)\left(1 - \delta\alpha + \frac{\delta^2\alpha^2}{2!} - \cdots\right) - \exp(\alpha h) < 0. \tag{31}$$

Since

$$\exp(-\delta\alpha) = 1 - \delta\alpha + \frac{\delta^2\alpha^2}{2!} - \cdots; \tag{32}$$

then,

$$\begin{aligned} \exp(\alpha h)(r\exp(-\delta\alpha) - 1) &< 0 & (33)\\ \alpha &> \frac{1}{\delta}\ln(r) & (34)\\ \alpha &> \frac{1}{\delta}\ln\left(\frac{\sqrt{\Delta^2 x + \Delta^2 y}}{\min(\Delta x, \Delta y)}\right), & (35) \end{aligned}$$

which is the necessary condition to satisfy Eq. (18). Under the proposed speed model, Eq. (29), all medial voxels are moving faster than non-medial ones. Since the cost function $U(\boldsymbol{x})$ is the reciprocal of the speed, then the voxels of a $\mathcal{MC}$ are the voxels with minimal cost value. Therefore, $\mathcal{MC}$ is a minimal cost path.

For a 3D lattice, derivation of the speed is straightforward. The value of α is given by

$$\alpha > \frac{1}{\delta}\ln\left(\frac{\sqrt{\Delta^2 x + \Delta^2 y + \Delta^2 z}}{\min(\Delta x, \Delta y, \Delta z)}\right). \tag{36}$$

4.2. Single $\mathcal{MC}$ Extraction

In the isotropic fast marching method, the fastest traveling is always along the direction perpendicular to the wavefront [34]. Since the propagating fronts are level sets, then the direction of the gradient at each medial voxel is normal to the front. Recall that the front is faster at medial voxels, so that $\mathcal{MC}$ can be found by backtracking along the gradient of $T(\boldsymbol{x})$. If we backtrack along $-\nabla T(\boldsymbol{x})$ a distance h starting from medial voxel P_n, we then reach the next voxel, P_{n+1}. This recursive process can be described by the following ordinary differential equation (ODE):

$$\frac{dC(s)}{ds} = -\frac{\nabla T}{|\nabla T|} \quad \text{given} \quad C(0) = P_0, \tag{37}$$

where $C(s)$ traces out the medial curve. Let A and B be the starting and ending voxels of a $\mathcal{MC}$; then tracking continues from $C(0) = B$ until A is found, as shown in Figure 2. The point A is guaranteed to be found since the field is monotonically increasing from A to B:

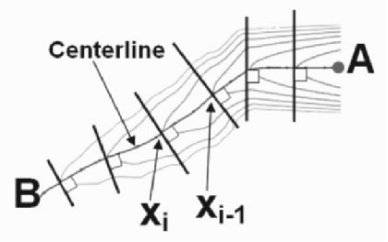


Figure 2. A medial curve $\mathcal{MC}$ intersects the propagating fronts at those voxels of maximum positive curvatures. See attached CD for color version.

4.3. Numerical Integration Methods for Extracting an $\mathcal{MC}$

Many differential equations cannot be solved analytically, in which case we have to find a good approximation to their solution. The ODE given by Eq. (37) can be solved using different integration methods [35], including the Euler method, Heun's method, and the Runge-Kutta methods. These methods are ordered according to their accuracy.

4.3.1. Euler Method

The derivative dC/ds is replaced by the finite-difference approximation

$$\frac{dC}{ds} = \frac{C(s+h) - C(s)}{h}, \tag{38}$$

which yields the following formula:

$$C(s+h) = C(s) - h\frac{\nabla T}{|\nabla T|}. \tag{39}$$

This formula is usually applied in the following way. We choose a step size h, and construct the sequence s_0, $s_0 + h$, $s_0 + 2h$, etc. Let C_n be the numerical estimate of the exact solution $C(s_n)$, and then the solution can be computed by the following recursive scheme:

$$C_{n+1} = C_n - h\frac{\nabla T}{|\nabla T|} \quad \text{given} \quad C(0) = B. \tag{40}$$

The Euler method is fast but less accurate. Its total accumulated error is on the order of $O(h)$.

4.3.2. Heun's Method

The Heun method initially uses Euler's method as a predictor, $\bar{C}_{n+1}$, and then finds solution C_{n+1} by correcting the predictor as follows:

$$\begin{aligned} f(C_n) &= -\frac{\nabla T}{|\nabla T|} \\ \bar{C}_{n+1} &= C_n + hf(C_n) \\ C_{n+1} &= C_n + \frac{h}{2}(f(C_n) + f(\bar{C}_{n+1})). \end{aligned} \tag{41}$$

The Heun method is slower but more accurate than the Euler method. The total accumulated error is on the order of $O(h^2)$.

4.3.3. Runge-Kutta Methods

The Runge-Kutta methods were developed around 1900 by the mathematicians C. Runge and M.W. Kutta. The next value C_{n+1} is determined by the present value C_n plus the product of the size of interval h and an estimated slope. A fourth-order method Runge-Kutta is given by

$$C_{n+1} = C_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \tag{42}$$

where

$$\begin{aligned} k_1 &= f(C_n) \\ k_2 &= f(C_n + \frac{h}{2}k_1), \\ k_3 &= f(C_n + \frac{h}{2}k_2), \\ k_4 &= f(C_n + hk_3). \end{aligned} \tag{43}$$

The slope is a weighted average of slopes k_1 at the beginning of the interval, k_2 and k_3 at the midpoint of the interval, and k_4 at the end of the interval. When the four slopes are averaged, more weight is given to the slopes at the midpoint. The Runge-Kutta method is a fourth-order method, meaning that the total accumulated error is on the order of $O(h^4)$.

The second-order Runge-Kutta method is given by

$$C_{n+1} = C_n + hk_2. \tag{44}$$

The total accumulated error is on the order of $O(h^3)$. For the Heun and Runge-Kutta methods, the gradient is evaluated at non-lattice voxels, which require bilinear/trilinear interpolation for 2D/3D spaces.

Although the fourth-order Runge-Kutta method is more accurate than others, the gain in accuracy is sometimes lost by interpolation at non-lattice voxels. Therefore, we will use the second-order Runge-Kutta method because it strikes a balance between accuracy and efficiency.

4.4. Medialness Function $\lambda(\boldsymbol{x})$

The medialness function $\lambda(\boldsymbol{x}) : \mathbb{R}^n \rightarrow \mathbb{R}^+$ is a scalar function that distinguishes medial voxels by assigning them higher weights than non-medial ones. It also must be monotonically increasing from the boundary of the object toward its center. In this work, we proposed the following medialness:

$$\lambda(\boldsymbol{x}) = D(\boldsymbol{x}). \tag{45}$$

This medialness assigns each object's voxel $\boldsymbol{x}$ its minimum distance from the boundary Γ as given by Eq. (46). $D(\boldsymbol{x})$ can be discretely approximated using chamfer metric (3,4,5) [19], or continuously approximated using the fast marching methods [20], which is more accurate than discrete approximation:

$$D(\boldsymbol{x}) = \min_{\mathbf{y}} \{d(\boldsymbol{x}, \mathbf{y}) | \mathbf{y} \in \Gamma\}. \tag{46}$$

If a wave is propagating from the object's boundary with unit speed, the arrival time solution of the eikonal equation is directly proportional to $D(\boldsymbol{x})$. In this chapter, $D(\boldsymbol{x})$ is computed by solving Eq. (47) using the proposed multi-stenciled fast marching method, (see Chapter 9)

$$|\nabla D(\boldsymbol{x})| = 1. \tag{47}$$

We have previously defined δ as the difference between the λ of two neighboring voxels, $\boldsymbol{x}$ and $\boldsymbol{y}$, where $\lambda(\boldsymbol{x}) > \lambda(\boldsymbol{y})$. Since the Euclidean distance between two neighboring voxels in 2D space is either Δx, Δy, or $\sqrt{\Delta^2 x + \Delta^2 y}$, then to handle all cases of voxel configurations,

$$\delta = \min(\Delta x, \Delta y). \tag{48}$$

Similarly, in 3D

$$\delta = \min(\Delta x, \Delta y, \Delta z). \tag{49}$$

This medialness is suitable for extracting centerlines of arbitrary 2D shapes as well as $\mathcal{MC}$ of 3D tubular objects, whose cross-section is nearly or perfectly circular.

4.5. Multiple $\mathcal{MC}$ Extraction

In order to extract the entire $\mathcal{MC}$ of an object, we have to first identify its important topological nodes from which its $\mathcal{MC}$ originate and then apply the proposed single $\mathcal{MC}$ extraction procedure, which starts from each topological node until P_S is reached or intersects a previously extracted $\mathcal{MC}$ to prevent overlapped paths.

4.5.1. Identification of Topological Nodes

If the object could be represented by a graph, then its topological nodes such as extreme, branching, and merging could be identified easily.

In this chapter, we propose an automatic method for identifying the topological nodes of an object by converting it into a graph. Initially, we compute the Euclidean distance field $D(\boldsymbol{x})$. We then select automatically one medial voxel that belongs to one of the object's $\mathcal{MC}$ and consider it a point source P_S that transmits a moderate speed wavefront. The front motion is governed by the eikonal equation, whose solution is a new distance field $D_1(\boldsymbol{x})$. The speed of the front is proportional to $D(\boldsymbol{x})$, as given by (50):

$$F(\boldsymbol{x}) = \exp\Big(\beta\lambda(\boldsymbol{x})\Big). \tag{50}$$

The parameter β is a speeding parameter to be estimated later. P_S is selected as the voxel with global maximum in $D(\boldsymbol{x})$:

$$P_S = \underset{\boldsymbol{x}}{\operatorname{argmax}}\ D(\boldsymbol{x}). \tag{51}$$

Let $\hat{D}_1(\boldsymbol{x})$ be the discretized version of the floating distance field $D_1(\boldsymbol{x})$:

$$\hat{D}_1(\boldsymbol{x}) = round\Big(D_1(\boldsymbol{x})\Big). \tag{52}$$

$\hat{D}_1$ converts the object with voxels as its basic elements into an object with clusters as its new basic elements (Cluster Graph). Each cluster consists of connected voxels with the same $\hat{D}_1$ value. Therefore, we can expect more than one cluster with the same $\hat{D}_1$ value, if they are not adjacent. Two voxels are said to be connected if they share a face, an edge, or a vertex *26-connected*). Two clusters c_1 and c_2 are adjacent if a voxel in c_1 shares a face with a voxel in c_2 (*6-connected*).

In the cluster graph, each cluster is represented by a node and adjacent clusters by links. The root of the graph is the cluster containing P_S with zero cluster value, followed by clusters with increasing $\hat{D}_1$ value. The cluster graph contains three types of clusters: Extreme clusters ($Xcluster$), which exists at the tail of the graph; Branching clusters ($Bcluster$), which has at least two adjacent clusters with the same $\hat{D}_1$ value but greater than the value of $Bcluster$; and Merging clusters ($Mcluster$), which has at least two adjacent clusters (*Successors*) with the same $\hat{D}_1$ value but lower than the value of $Mcluster$. Merging clusters exist only if the object contains holes (loops).

The medial voxel of a cluster is computed by searching the cluster for the voxel with maximum $D(\boldsymbol{x})$. Extreme and merging nodes are the medial voxels of the associated clusters. Figure 3a shows the cluster graph of a tree structure with one loop, where X, M, and S represent extreme, merging, and successor clusters, respectively.

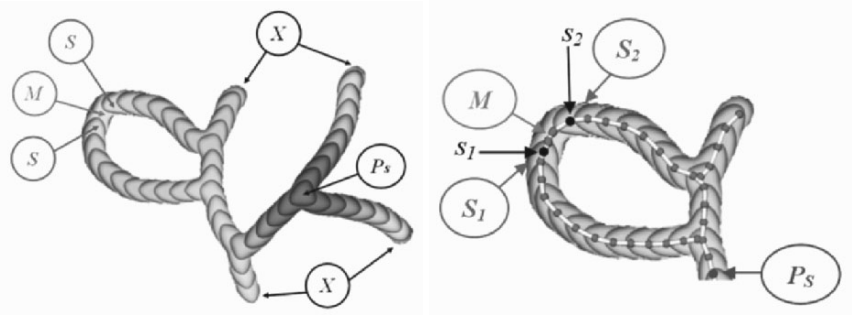
4.5.2. Analytical Derivation of β

For clarity, let's first estimate the value of β for a 2D lattice and then generalize it to a 3D lattice. Consider the $\mathcal{MC}$ of an arbitrary 2D shape whose endpoints are A and B, as shown in Figure 4a. It consists of N points and $N-1$ line segments. Let A be a source point P_S that transmits a moderate speed wave Eq. (50). Let T be the total travel time from A to B along the $\mathcal{MC}$:

$$T = \sum_{i=1}^{N-1} \Delta t_i, \tag{53}$$

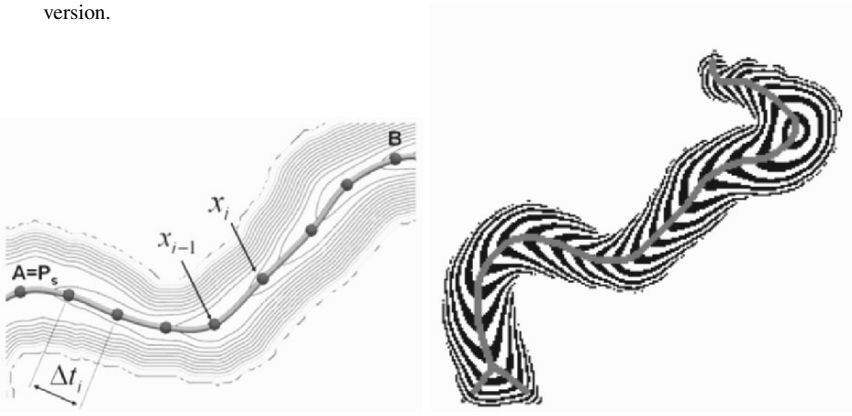
where Δt_i and $d(.,.)$ are the travel time and Euclidean distance between two medial neighbor voxels, respectively:

$$\Delta t_i = \frac{d(\boldsymbol{x}_{i-1}, \boldsymbol{x}_i)}{F(\boldsymbol{x}_i)}. \tag{54}$$



(a) (b)

Figure 3. (a) Cluster Graph. (b) Medial curves around a loop. See attached CD for color version.



(a) (b)

Figure 4. (a) The $\mathcal{MC}$ of an arbitrary shape consists of N medial voxels. (b) Cluster graph of the same shape. See attached CD for color version.

By restricting the value Δt_i to be greater than a certain value τ, where $0 < \tau < 1$, then

$$\tau \leq \Delta t_i, \tag{55}$$

$$\tau \leq \frac{d(\boldsymbol{x}_{i-1}, \boldsymbol{x}_i)}{\exp(\beta\lambda(\boldsymbol{x}_i))}, \tag{56}$$

$$\beta \leq \frac{1}{\lambda(\boldsymbol{x})} \ln\left(\frac{d(\boldsymbol{x}_{i-1}, \boldsymbol{x}_i)}{\tau}\right). \tag{57}$$

The worst-case scenario for the right-hand side of Eq. (57) occurs when $\lambda(\boldsymbol{x}) = \lambda_{max}$ and

$$d(\boldsymbol{x}_{i-1}, \boldsymbol{x}_i) = \min(\Delta x, \Delta y). \tag{58}$$

Let β_c be the critical value of β; then

$$\beta_c = \frac{1}{\lambda_{max}} \ln\left(\frac{\min(\Delta x, \Delta y)}{\tau}\right). \tag{59}$$

The parameter τ is inversely proportional to the maximum number of voxels V_m that the cluster can have along the $\mathcal{MC}$ passing through it. For example, for $\tau = 0.2$, the arrival times $t_i - 2\tau, t_i - \tau, t_i, t_i + \tau, t_i + 2\tau$ will be rounded to t_i, which will result in a cluster with 5 voxels along the $\mathcal{MC}$ passing through it, as shown in Figure 4b.

Setting $V_m = 1$ corresponds to $\tau = 1$. Under isotropic voxels, $\Delta x = \Delta y = \Delta z = 1$, $\beta = 0$, and then $F = 1$. As a consequence, the front is not fast at the middle of the shape and hence the cluster graph fails to capture correctly the topology of the shape, especially its large curvature parts such as loops. Therefore, V_m must be greater than 1, which corresponds to $\tau < 1$.

For a 3D lattice,

$$\beta_c = \frac{1}{\lambda_{max}} \ln\left(\frac{\min(\Delta x, \Delta y, \Delta z)}{\tau}\right). \tag{60}$$

4.5.3. $\mathcal{MC}$ of Loops

Our framework extracts one $\mathcal{MC}$ for each tunnel or cavity the object may have. Each loop in the object is associated with one merging cluster M of the cluster graph. For illustration, let M have only two successors S_1 and S_2, as shown in Figure 3b. In order to extract the $\mathcal{MC}$ of this loop, three steps are required. In the first step, we compute the medial voxel s_1 of S_1 and consider the entire set of voxels of both M and S_2 as part of the object's background (construct holes) such that there is a unique $\mathcal{MC}$ from s_1 to P_S. Finally, we propagate a fast wave from P_S until s_1 is reached and then extract the $\mathcal{MC}$ between them. In the second step, we extract the $\mathcal{MC}$ between s_2 and P_S in a similar fashion to the first step, except that we consider the entire voxels of both M and S_1 as part of the object's background and those of S_2 as part of the object's foreground. In the third step, we propagate a fast wave from s_1 until s_2 is reached and then extract the $\mathcal{MC}$ between them. The same concept can be generalized for a merging cluster with any number of successors.

4.6. The Medical Curve Extraction Algorithm

The proposed medial curve $\mathcal{MC}$ extraction framework can be summarized as follows: (1) construct the Euclidean distance field $D(\boldsymbol{x})$; (2) find the point source

P_S; (3) propagate a moderate speed wave from P_S and solve for the new distance field $D_1(\boldsymbol{x})$; (4) discretize $D_1(\boldsymbol{x})$ to obtain $\hat{D}_1(\boldsymbol{x})$ and then construct the cluster graph; (5) identify the extreme and merging nodes; (6) construct a new distance field from P_S by propagating a fast speed wave and solve for the new distance field $D_2(\boldsymbol{x})$; (7) if the object contains loops, extract their $\mathcal{MC}$; and, finally, (8) extract those $\mathcal{MC}$ that originate from extreme nodes and end with either a P_S or on a previously extracted path. The pseudocode of the proposed framework is presented in Algorithm 1.

Algorithm 1 Proposed $\mathcal{MC}$ Extraction Framework

$D(\boldsymbol{x})$ = ComputeEuclideanDistance(*UnitSpeed*)
P_S = FindSourcePoint()
$D_1(\boldsymbol{x})$ = ComputeDistanceFromSource(P_S,*ModerateSpeed*)
CG = CreateClusterGraph()
$[M_1, M_2, \cdots, M_k]$ = GetMergingClusters(CG)
{Handle Holes}
for $i = 1$ to K **do**
  $S = [S_1, S_2, \cdots, S_l]$ = GetSuccessors(M_i)
  AddToBackground(M_i)
  for $j = 1$ to l **do**
    AddToBackground(S/S_j)
    $D_2(\boldsymbol{x})$ = ComputeDistanceFromSource(P_S,*HighSpeed*)
    P_j = GetMedialVoxel(S_j)
    C = ExtractMedialCurve(P_j)
    AddToForeground(S/S_j)
  end for
  AddToForeground(M_i)
  $D_2(\boldsymbol{x})$ = ComputeDistanceFromSource(P_1,*HighSpeed*)
  for $j = 2$ to l **do**
    C = ExtractMedialCurve(P_j)
  end for
end for
{Handle Branches}
$[X_1, X_2, \cdots, X_N]$ = GetExtremeClusters(CG)
$D_2(\boldsymbol{x})$ = ComputeDistanceFromSource(P_S, *HighSpeed*)
for $j = 1$ to N **do**
  P_j = GetMedialVoxel(X_j)
  C = ExtractMedialCurve(P_j)
end for

4.7. Complexity Analysis

During extraction of all the medial curves of an object, we use the proposed multi-stencils fast marching method in computing several distance fields: the Euclidean distance field $D(\boldsymbol{x})$, the new distance field due to moderate speed propagation $D_1(\boldsymbol{x})$, and the new distance field due to high-speed propagation $D_2(\boldsymbol{x})$. We have shown that the proposed multi-stencils fast marching method has a complexity of $O(n)$ by implementing the narrow band as an untidy priority queue [36], where n is the number of object voxels. Therefore, the complexity of the proposed extraction framework is $O(3n)$.

5. VALIDATION AND SENSITIVITY ANALYSIS

In this section, we aim to study the accuracy of the analytical parameter estimation of the proposed speed models, validating the accuracy of the computed $\mathcal{MC}$ by the proposed framework, and finally studying its sensitivity to boundary noise.

5.1. 3D Synthetic Shapes

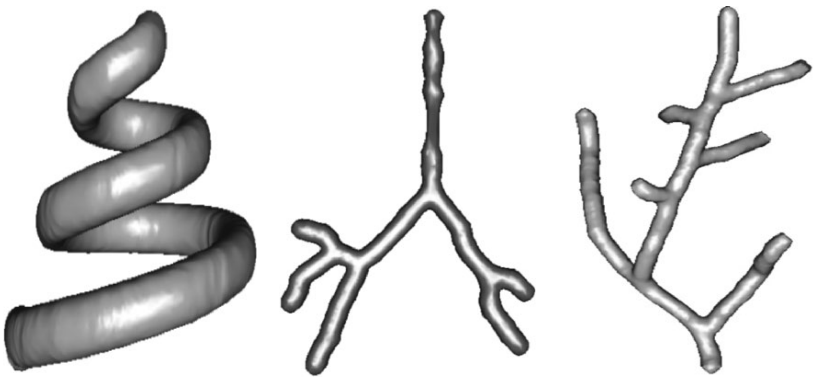
We have prepared 3D synthetic shapes with known ground truth $\mathcal{MC}$. The ground truths are generated analytically and then voxelized. Each synthetic shape is created by translating a sphere of fixed or varying radius along its ground truth path. In Figure 5 we show 3D synthetic shapes with different complexity. The phantoms are designed to measure the performance of the method when the anatomical structures have the following geometrical or topological properties: (1) high curvature and torsion (e.g., blood vessels), (2) sudden change in the organ cross-section (e.g., colon or aneurysm in vessels), and (3) several branching nodes (e.g., blood vessels and tracheobronchial trees).

5.2. Accuracy of the Analytical Estimation of α

The parameter α controls the centeredness property of the computed $\mathcal{MC}$. In the previous section, we derived a closed form analytical solution for α. The critical value of α is given by

$$\alpha_c = \frac{1}{\delta} \ln \left(\frac{\sqrt{\Delta^2 x + \Delta^2 y + \Delta^2 z}}{\min(\Delta x, \Delta y, \Delta z)} \right). \tag{61}$$

Bad estimation of α may lead to undesired properties in the computed skeletons. For example, if $\alpha \ll \alpha_c$, the computed paths are trajectories rather than centered $\mathcal{MC}$. On the other hand, if $\alpha \gg \alpha_c$, the front speed will be very high and hence the arrival time nearly zero. Since when extracting a $\mathcal{MC}$ we stop if we reach the voxel with zero travel time, the extraction process may halt at the first extracted



(a) (b) (c)

Figure 5. 3D synthetic shapes of different complexity: (a) spiral, (b) simple tree (in-plane), and (c) complex tree.

voxel, which leads to disconnected $\mathcal{MC}$. Since $\delta = \min(\Delta x, \Delta y, \Delta z)$, then for isotropic voxel size ($\Delta x = \Delta y = \Delta z = 1.0$), the critical value of α_c is 0.549.

In this experiment, we study the accuracy of the analytical estimation of α by manually changing the value of α from 0 to 1 in steps of 0.1. In each step, we compute the L_1 error between the computed $\mathcal{MC}$ and the ground truth for several 3D synthetic shapes. We then plot the L_1 error versus α and determine the range of α (R_α) that corresponds to the steady-state range of the minimum L_1 error, as shown in Figure 6. Finally, we check if the estimated α_c is within that range. In Table 1 we show the computed L_1 error under different values of α of various synthetic shapes: spiral (Figure 5a), simple tree (Figure 5b), and complex tree (Figure 5c). It is clear from Figure 6 that the estimated α_c is always within R_α. Also, there is a wide stable range for α that is $\alpha_c \leq \alpha \leq 1.0$. In our experiments we automated the proposed framework by fixing $\alpha = 0.7$.

5.3. Effect of Changing β on the Generation of Cluster Graph

The parameter β controls generation of the cluster graph. In the previous section we derived a closed-form analytical solution for β. The critical value of β is given by

$$\beta_c = \frac{1}{\lambda_{max}} \ln\left(\frac{\min(\Delta x, \Delta y, \Delta z)}{\tau}\right). \tag{62}$$

Table 1. Computed L_1 Error of Various Synthetic Shapes under Different Values of α

α	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1.0**
L_1 (Spiral)	0.451	0.335	0.328	0.327	0.326	0.324	0.322	0.322	0.317	0.317
L_1 (Simple Tree)	0.388	0.247	0.186	0.176	0.179	0.175	0.173	0.172	0.175	0.174
L_1 (Complex Tree)	0.535	0.405	0.381	0.371	0.365	0.361	0.361	0.359	0.357	0.359

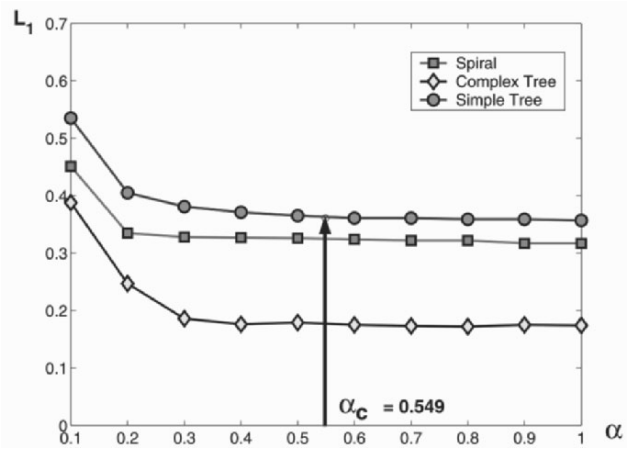
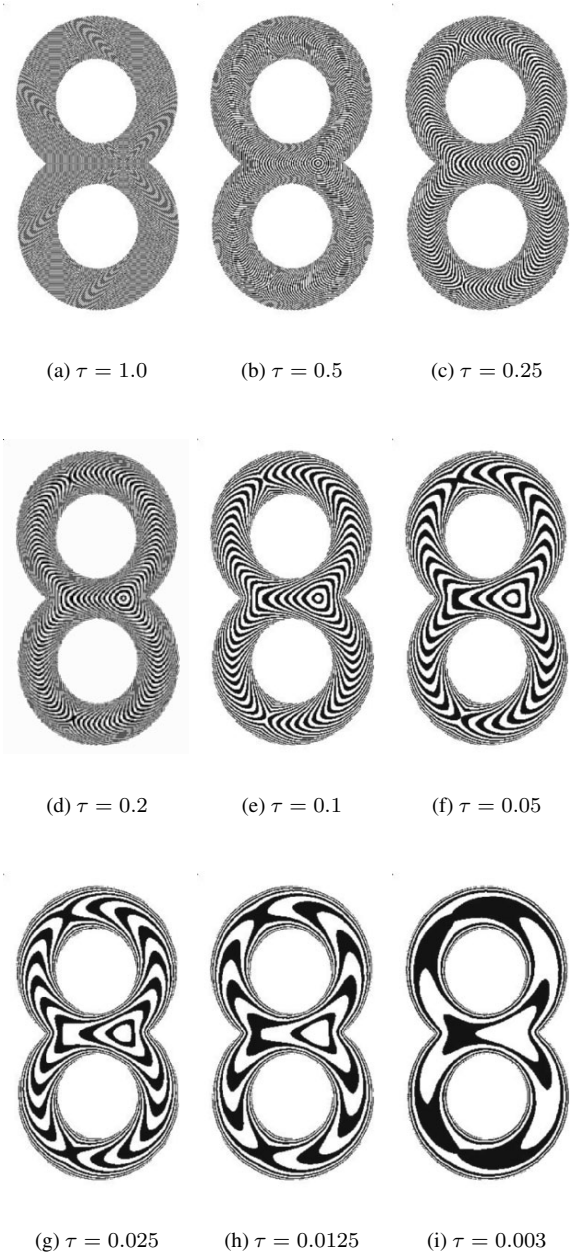


Figure 6. Computed L_1 error of various synthetic shapes under different values of α. See attached CD for color version.

The parameter τ is inversely proportional to the maximum number of voxels V_m that the cluster has along the $\mathcal{MC}$ passing through it. We have shown that $0 < \tau < 1$. In this section we study the effect of changing β on generation of the cluster graph.

From Figures 7 and 8, which show different cross-sections in the cluster graph of 3D synthetic shapes (double donuts and a tree with a hole) under different values of τ, we can infer that there is a wide stable range $[0.05, 0.2]$ for β to generate a good cluster graph that is capable of correctly describing the topology of the shape. If $\tau < 0.05$, the tails of end clusters span multiple neighbor clusters; therefore, the medial voxel of the cluster may be wrongly determined. On the other hand, if $\tau > 0.2$, the cluster graph does not capture the large curvature parts of the shape, especially if the shape contains both branches and loops. In our experiments we automated the proposed framework by fixing $\tau = 0.1$.

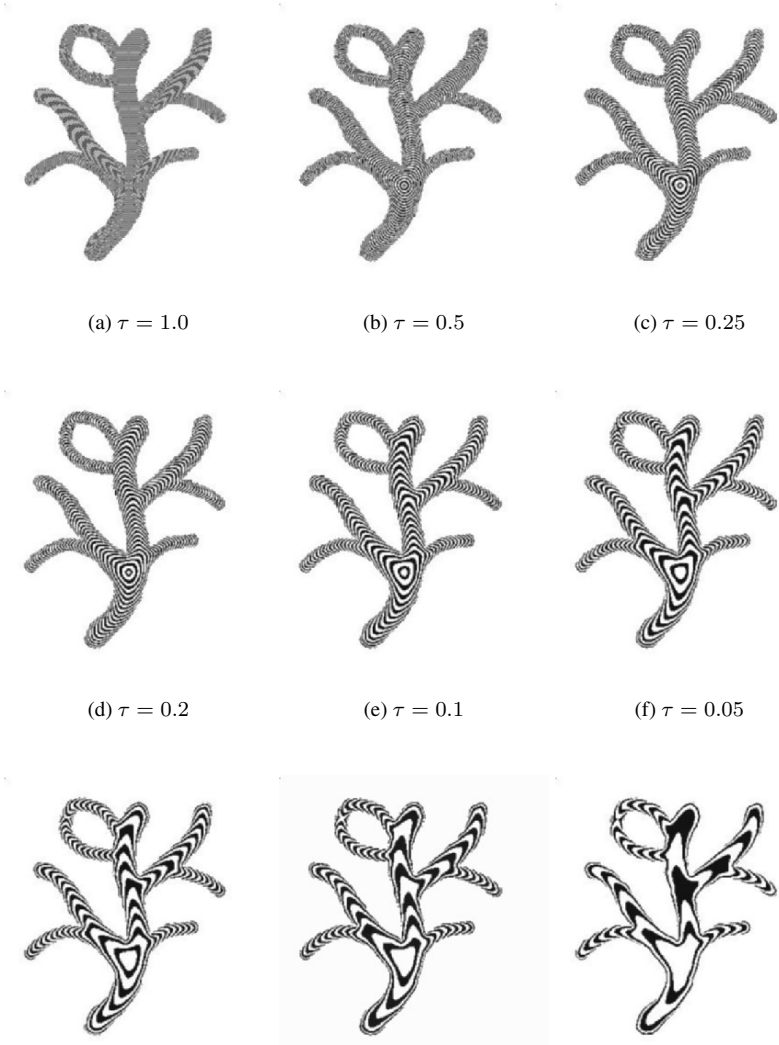


(a) $\tau = 1.0$ (b) $\tau = 0.5$ (c) $\tau = 0.25$

(d) $\tau = 0.2$ (e) $\tau = 0.1$ (f) $\tau = 0.05$

(g) $\tau = 0.025$ (h) $\tau = 0.0125$ (i) $\tau = 0.003$

Figure 7. Cross-sections in the cluster graph of a 3D synthetic shape (double donuts) under different values of τ.



(a) $\tau = 1.0$ (b) $\tau = 0.5$ (c) $\tau = 0.25$

(d) $\tau = 0.2$ (e) $\tau = 0.1$ (f) $\tau = 0.05$

(g) $\tau = 0.025$ (h) $\tau = 0.0125$ (i) $\tau = 0.003$

Figure 8. Cross-sections in the cluster graph of a 3D synthetic shape (Tree) under different values of τ.

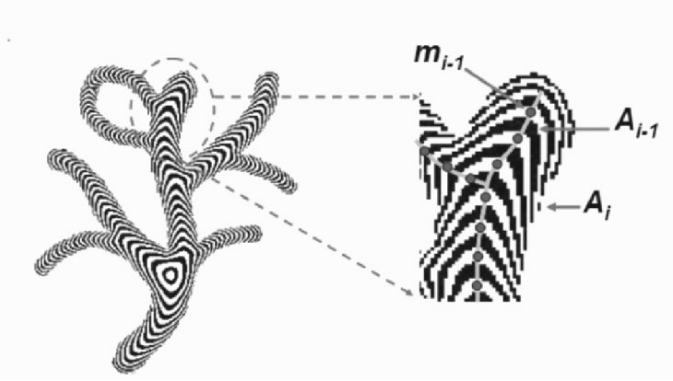


Figure 9. Each extreme node is identified not from its corresponding extreme cluster A_i but from its successor neighbor A_{i-1}. See attached CD for color version.

5.4. Accuracy of Centeredness and Sensitivity to Boundary Noise

For branching objects, each $\mathcal{MC}$ must start from an extreme node. It is known that any perturbation of the object's boundary creates an extra undesired $\mathcal{MC}$. Therefore, for noisy objects, a large number of $\mathcal{MC}$ are expected to be generated. The proposed framework overcomes this problem by identifying each extreme node not from its corresponding extreme cluster but from its successor neighbor cluster. For example, in Figure 9 the boundary noise has formed the extreme cluster A_i, whose successor cluster is A_{i-1}. The medial voxel of the extreme cluster is chosen to be m_{i-1} of the successor cluster A_{i-1}, which is already part of the original $\mathcal{MC}$ of the object; therefore, m_i will not start a new $\mathcal{MC}$ because it does not belong to a salient part of the object.

In this experiment we study the sensitivity of the proposed framework against boundary noise by corrupting the object's boundary voxels with an additive noise of different noise levels to simulate either an acquisition or segmentation error. If the boundary is corrupted by an $N_l\%$ noise level, then $N_l\%$ of the boundary's voxels are randomly corrupted by an additive noise. If we set $N_l = 0$, then the centeredness property of the computed $\mathcal{MC}$ can be studied as well. For each noise level, a quantitative analysis was carried out by computing the average error L_1 in mm, maximum error L_∞ in mm, and standard deviation σ between the ground truth and computed $\mathcal{MC}$ for both noise-free and noisy synthetic shapes.

In Figures 10 and 11 we show synthetic shapes with a smooth surface (0% noise level), a rough surface (100% noise level), $\mathcal{MC}$ of smooth surface, and $\mathcal{MC}$ of rough surface. The quantitative results are presented in Tables 2 and 3.

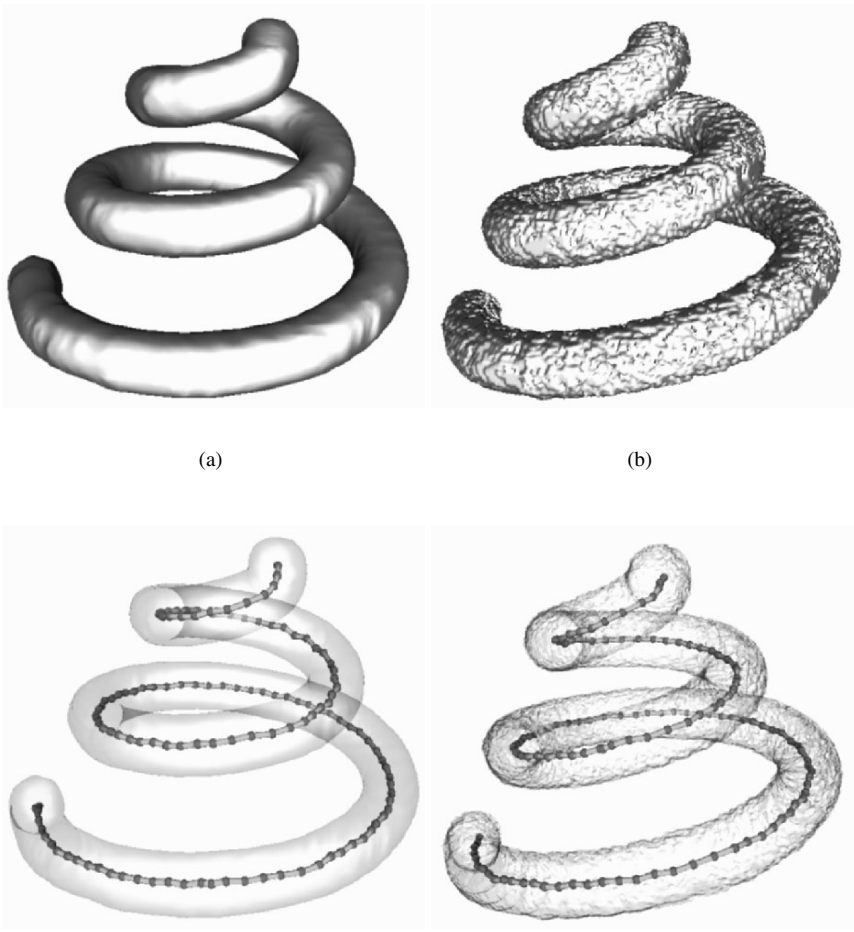
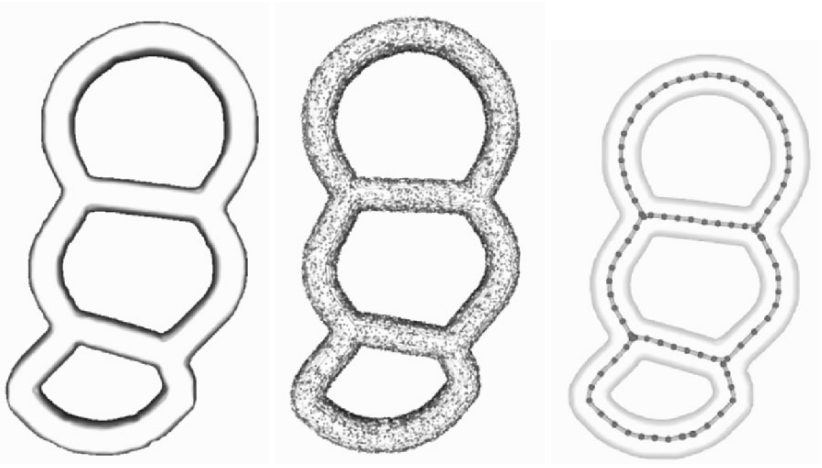


Figure 10. Spiral synthetic shape: (a) smooth surface (0% noise level), (b) rough surface (100% noise level), (c) $\mathcal{MC}$ of smooth surface, (d) $\mathcal{MC}$ of rough surface. See attached CD for color version.



(a) (b) (c)

Figure 11. Three-connected donuts. Synthetic shape: (a) smooth surface (0% noise level), (b) rough surface (100% noise level), (c) $\mathcal{MC}$ of smooth surface. See attached CD for color version.

For noise-free synthetic shapes, the L_1 and L_∞ errors never exceeded 0.37 and 1.73 mm (e.g., one voxel), respectively. In the presence of severe noise levels, L_1 and L_∞ errors never exceeded 0.47 and 2.23 mm (e.g., two voxels), respectively, which is quite acceptable for flight paths; therefore, the proposed method has low sensitivity to boundary noise.

It is worth noticing that under some noise levels some error measures decrease rather than increase because the profile of noise become symmetrically distributed around the object, and hence the centeredness property of the computed $\mathcal{MC}$ is not altered.

6. RESULTS

We have also validated the proposed method qualitatively against several clinical datasets as shown in Figure 12. Notice the complexity of the clinical datasets and the accuracy of the computed $\mathcal{MC}$ especially around loops and near branching and merging nodes.

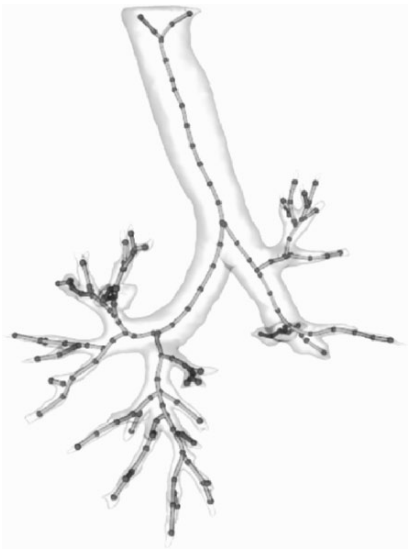
We have implemented the proposed method using C++ on a single 400-Mhz SGI infinite reality supercomputer. The volume sizes and running times in seconds of the tested datasets are listed in Table 4.

(a) Vessels

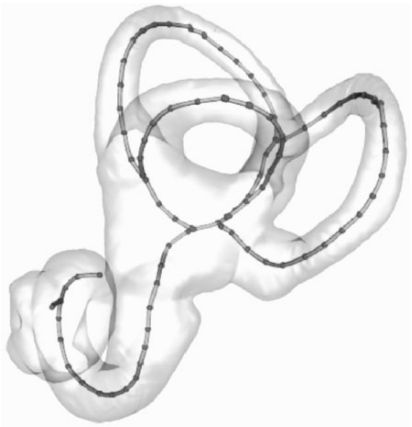
(b) Aorta-1

(c) Spinal

(d) Aorta-2



(e) Trachea



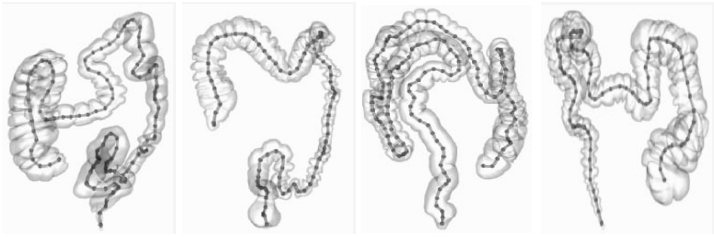
(f) Middle Ear

Figure 12. Computed $\mathcal{MC}$ of clinical datasets. See attached CD for color version.

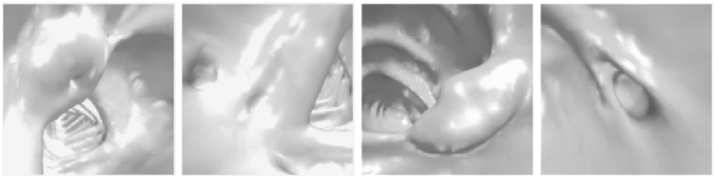
6.1. Case Study: Virtual Colonoscopy

The goal of this case study is to bring about one of the important goals of virtual colonoscopy (VC) as a diagnostic tool. The ideal scenario is that the segmented colon maintain the intricate details of the real colon, and that the virtual camera project views as detailed as those shown in real colonoscopy. If that is achieved, then analysis of projected views can be used for automatic colon scanning against abnormalities, and hence early detection of colon cancer using VC would be a strong possibility. Our research group is currently pursuing this goal.

In order to illustrate the potential of this research in colonoscopy, the proposed framework has been tested on several CT datasets acquired using a Siemens Sensation CT scanner. The dataset volume is $512 \times 512 \times 580$ with a voxel size of $0.74 \times 0.74 \times 0.75$. All patients have undergone standard cleansing preparations prior to scan. In Figure 13e–l we show different polyps captured by a virtual camera for different colons that are shown in Figure 13a–d. We have rendered the results using the visualization toolkit (vtk).



(a) Colon 1 (b) Colon 2 (c) Colon 3 (d) Colon 4



(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 13. Virtual colonoscopy: (a–d) computed $\mathcal{MC}$ for different colon datasets; (e–l) polyp views captured by the virtual camera. See attached CD for color version.

Table 2. L_1, L_∞, and σ Error Measures of the Computed $\mathcal{MC}$ of a Synthetic Spiral Shape

Error Measure / Noise Level N_l	**0%**	**20%**	**40%**	**60%**	**80%**	**100%**
L_1	0.3743	0.3990	0.3500	0.4200	0.3829	0.4646
L_∞	1.7320	1.7320	1.4140	2.2300	1.414	1.4140
σ	0.2518	0.2671	0.2500	0.2800	0.2596	0.2916

Table 3. L_1, L_∞, and σ Error Measures of the Computed $\mathcal{MC}$ of a 3-Connected Donut's Shape

Error Measure / Noise Level N_l	**0%**	**20%**	**40%**	**60%**	**80%**	**100%**
L_1	0.2123	0.2900	0.2372	0.4409	0.2670	0.4700
L_∞	1.0000	1.4140	1.4140	2.0000	2.0000	2.2300
σ	0.1672	0.2127	0.1871	0.2741	0.2580	0.2760

Table 4. Sizes and Running Times of Different Clinical Datasets Using the Proposed Method

Anatomical Organ	**Actual Volume**	**Time in Sec.**
Vessels	$512 \times 512 \times 93$	180
Trachea	$145 \times 248 \times 198$	70
Aorta-1	$84 \times 145 \times 365$	40
Aorta-2	$130 \times 259 \times 97$	55
Spinal	$253 \times 49 \times 55$	13
Middle Ear	$162 \times 241 \times 125$	114

7. CONCLUSION AND FUTURE WORK

In this chapter we have presented a robust, fully automatic, and fast method for computing flight paths of tubular structures for virtual endoscopy applications. The computed flight paths enjoy several advantageous features, such as being centered, connected, thin, and less sensitive to noise. Unlike previous methods, our technique can handle complex anatomical structures with an arbitrary number of loops, can extract only part of the skeleton given the starting and ending voxels of the medial curve to be computed, and, finally, does not require voxels to be of isotropic size because it takes voxel data spacing into account. The robustness of the proposed method is demonstrated by correctly extracting all the $\mathcal{MC}$ of the tested clinical datasets as well as successful validation against synthetic phantoms of different complexity. In the future we intend to implement the proposed method in the graphical processing unit (GPU) of a computer graphics card to reduce computational time.

8. REFERENCES

1. Baert AL, Sartor K. 2001. *Virtual endoscopy and related 3D techniques*. Berlin: Springer.
2. Buthiau D, Khayat D. 2003. *Virtual endoscopy*. Berlin: Springer.
3. Y Zhou, Toga AW. 1999. Efficient skeletonization of volumetric objects. *IEEE Trans Visualiz Comput Graphics* **5**(3):196–209.
4. Bitter I, Kaufman AE, Sato M. 2001. Penalized-distance volumetric skeleton algorithm. *IEEE Trans Visualiz Comput Graphics* **7**(3):195–206.
5. Ma CM, Sonka M. 1996. A fully parallel 3d thinning algorithm and its applications. *Comput Vision Image Understand* **64**:420–433.
6. Svensson S, Nyström I, Sanniti di Baja G. 2002. Curve skeletonization of surface-like objects in 3d images guided by voxel classification. *Pattern Recognit Lett*, **23**(12):1419–1426.
7. Deschamps T. 2001. *Curve and shape extraction with minimal path and level-sets techniques: applications to 3D medical imaging*. PhD dissertation, Université Paris, IX Dauphine.
8. Bouix S, Siddiqi K, Tannenbaum A. 2003. Flux driven fly throughs. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition*, pp. 449–454. Washington, DC: IEEE Computer Society.
9. Hassouna MS, Farag AA. 2005. PDE-based three-dimensional path planning for virtual endoscopy. In: *Information processing in medical imaging: 19th international conference, IPMI 2005*, pp. 529–540. Lecture Notes in Computer Science, Vol. 3565. Berlin: Springer.
10. Hassouna MS, Farag AA, Falk R. 2005. Differential fly-throughs (DFT): a general framework for computing flight paths. In *Medical image computing and computer-assisted intervention: MICCAI 2005: 8th international conference*, pp. 26–29. Berlin: Springer.
11. Ma CM. 1995. A fully parallel thinning algorithm for generating medial faces. *Pattern Recognit Lett* **16**:83–87.
12. Tsao YF, Fu KS. 1981. A parallel thinning algorithm for 3d pictures. *Comput Graphics Image Process* **17**:315–331.

13. Saha PK, Majumder DD. 1997. Topology and shape preserving parallel thinning for 3d digital images: a new approach. In *Proceedings of the 9th international conference on image analysis and processing*, Vol. 1, pp. 575–581. Lecture Notes in Computer Science, Vol. 1310. Berlin: Springer.

14. Palagyi K, Kuba A. 1997. A parallel 12-subiteration 3d thinning algorithm to extract medial lines. In *Proceedings of the 7th international conference on computer analysis of images and patterns*, pp. 400–407. Lecture Notes in Computer Science, Vol. 1296. Berlin: Springer.

15. Lohou C, Bertrand G. 2004. A 3d 12-subiteration thinning algorithm based on p-simple points. *Discr Appl Math* **139**(1–3):171–195.

16. Manzanera A, Bernard TM, Prêteux F, Longuet B. 1999. A unified mathematical framework for a compact and fully parallel n-d skeletonization procedure. *Proc SPIE*, **3811**:57–68.

17. Palagyi K, Kuba A. 1999. Directional 3d thinning using 8 subiterations. In *Proceedings of the 8th international conference on discrete geometry for computer imagery (DCGI '99)*. Lecture Notes in Computer Science, Vol. 1568, pp. 325–336. Berlin: Springer.

18. Gong W, Bertrand G. 1990. A simple parallel 3d thinning algorithm. In *Proceedings of 10th International Conference on Pattern Recognition, 1990*, pp. 188–190. Washington, DC: IEEE.

19. Borgefors G. 1986. Distance transformations in digital images. *Comput Vision Graphics Image Process* **34**:344–371.

20. Adalsteinsson D, Sethian J. 1995. A fast level set method for propagating interfaces. *J Comput Phys* **118**:269–277.

21. Gagvani N, Silver D. 1999. Parameter-controlled volume thinning. *Graphical Models Image Process* **61**(3):149–164.

22. Bitter I, Sato M, Bender M, McDonnell KT, Kaufman A, Wan M. 2000. Ceasar: a smooth, accurate and robust centerline extraction algorithm. In *Proceedings of the Visualization '00 conference*, pp. 45–52. Washington, DC: IEEE Computer Society.

23. Sato M, Bitter I, Bender MA, Kaufman AE, Nakajima M. 2000. Teasar: Tree-structure-extraction algorithm for accurate and robust skeletons. In *Proceedings of the 8th Pacific conference on computer graphics and applications (PG '00)*, p. 281. Washington, DC: IEEE Computer Society.

24. Dijkstra EW. 1959. A note on two problems in connexion with graphs. *Num Math* **1**:269–271.

25. Paik DS, Beaulieu CF, Jeffrey RB, Rubin GD, Napel S. 1998. Automated path planning for virtual endoscopy. *Med Phys* **25**(5):629–637.

26. Dimitrov P, Damon JN, Siddiqi K. 2003. Flux invariants for shape. In *Proceedings of 2003 IEEE computer society conference on computer vision and pattern recognition (CVPR 2003)*, pp. 835–841. Washington, DC: IEEE Computer Society.

27. Pudney C. 1998. Distance-ordered homotopic thinning: a skeletonization algorithm for 3d digital images. *Comput Vision Image Understand* **72**(3):404–413.

28. Telea A, Vilanova A. 2003. A robust level-set algorithm for centerline extraction. In *Proceedings of the symposium on visualization (VisSym 2003)*, pp. 185–194. Aire-la-Ville, Switzerland: Eurographics Association.

29. Deschamps T, Cohen LD. 2001. Fast extraction of minimal paths in 3d images and applications to virtual endoscopy. *Med Image Anal* **5**(4): 281–299.

30. Chuang J-H, Tsai C-H, Ko M-C. 2000. Skeletonization of three-dimensional object using generalized potential field. *IEEE Trans Pattern Anal Machine Intell* **22**(11):1241–1251.

31. Yuan X, Balasubramanian R, Cornea ND, Silver D. 2005. Computing hierarchical curve-skeletons of 3d objects. *The Visual Computer*. **21**(11):945–955.

32. Ma W-C, Wu F-C, Ouhyoung M. 2003. Skeleton extraction of 3d objects with radial basis functions. In *Proceedings of the 2003 international conference on shape modeling and applications (SMI 2003)*, pp. 207–215, 295. Washington, DC: IEEE Computer Society.

33. Wu F-C, Ma W-C, Liou P-C, Laing R-H, Ouhyoung M. 2003. Skeleton extraction of 3d objects with visible repulsive force. In *Proceedings of the Computer Graphics Workshop 2003* (Hua-Lien, Taiwan). Available online: http://www.lems.brown.edu/vision/people/leymarie/Refs/CompGraphics/Shape/Skel.html.

34. Bellman R, Kalaba R. 1965. *Dynamic programming and modern control theory.* London: London Mathematical Society Monographs.

35. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. 1992. *Numerical recipes in C: the art of scientific computing*. New York: Cambridge UP.

36. Yatziv L, Bartesaghi A, Sapiro G. 2006. A fast o(n) implementation of the fast marching algorithm. *J Comput Phys* **212(2)**:393–399.

14

OBJECT TRACKING IN IMAGE SEQUENCE COMBINING HAUSDORFF DISTANCE, NON-EXTENSIVE ENTROPY IN LEVEL SET FORMULATION

Paulo S. Rodrigues and Gilson A. Giraldi

National Laboratory for Scientific Computing
Petropolis, Brazil

Jasjit S. Suri

Biomedical Research Institute, Idaho State University, Pocatello, Idaho, USA

Ruey-Feng Chang

Department of Computer Science and Information Engineering, National Chung Cheng University
Chiayi, Taiwan

The task of Object Recognition is one of the most important problems in computational vision systems. Generally, to meet this problem, specific approaches are applied in two phases: detecting the region of interest and tracking them around a frame sequence. For applications such as medical images or general object tracking, current methodologies generally are machine dependent or have high speckle noise sensitivity. In a sequence of ultrasound images some data are often missing if the approach does not use lesion-tracking methods. Also, the segmentation process is highly sensitive to noise and changes in illumination. In this chapter we propose a four-step method to handle the first phase of the problem. The idea is to track the region of interest using the information found in

Address all correspondence to: Paulo Sérgio Rodrigues, Laboratório Nacional de Computação Científica, Av. Getulio Vargas, 333, Quitandinha, Petropolis, Brazil, CEP: 25651-075. Phone: +55 24 2233-6088, Fax: +55 24 2231-5595. pssr@lncc.br.

the previous slice to search for the ROI in the current one. In each image (frame) we accomplish segmentation with Tsallis entropy, which is a new kind of entropy for non-extensive systems. Then, employing the Hausdorff distance we match candidate regions against the ROI in the previous image. In the final step, we use the ROI curve to compute a narrow band that is an input for an embedded function in a level set formulation, smoothing the final shape. We have tested our method with three classes of images: a general indoor office, a Columbia database, and low-SNR ultrasound images of breast lesions, including benign and malignant tumors, and have compared our proposed method with the Optical Flow Approach.

1. INTRODUCTION

In the field of Computational Vision, detecting and tracking a region or object of interest from a scenario represented by a frame sequence of real images is a known problem. Even though several approaches have been proposed, this problem is still a challenge where the solutions are closely related to the desired goals, as well as to some features of the data to be analyzed. We may split these goals between two separate groups: one related to the type of object to be recognized, and the other related to the velocity of recognition. On the other hand, the features of the data to be analyzed are also partitioned between two groups: one related to the quality of the images and other related to the type of sequence used.

Regarding the type of object to be recognized, it may have a irregular and flexible (or rigid) shape, as we see in many medical images. Also, during scene evolution we may see random transformations such as in scale, rotation, translation, color, and texture, and topologic changes such as split and merge. With regard to the objectives regarding velocity of recognition, the solution may require a real-time response as well as high volume and dimension of images. Additionally, we may find solutions to treat occlusion problems.

On the other hand, regarding features that influence problem solution, image quality is one that requires more attention. As examples, noise distribution, low contrast, and resolution are among the more difficult challenges. Another set of features is the type of data sequence used, where we may have sequences with high or low changes in illumination, which so far affect the segmentation algorithms, as well as the formulation parameters. We may also have objects or regions in the sequence that are not of interest, or even several objects or regions to be recognized, a moving background, and large variations in view or zoom.

These four groups of objectives and features demand specific solutions for recognition and/or tracking of objects or regions in a specific sequence of images. Examples of such images include those from digital video of medical images, like those produced by ultrasound devices. There is a high incidence of breast cancer among women throughout the world, and it has become one of the major causes of death among women. Mammographic examination has been an important ally in detection of this condition. It is well known that, although high-resolution

3D scanners have been available on the market for some time, more than 90% of ultrasound exams are accomplished using traditional 2D scanners. These devices generally offer a low signal-to-noise ratio (SNR) as well as low contrast and resolution, while Digital Image Processing promises to enhance the quality and usefulness of images. On the other hand, the shear volume of exams and the time spent on image collection and evaluation require intense attention and from an experienced medical team. Many investigators have found that more than 60% of masses referred for breast biopsy on the basis of mammographic findings are actually benign [1–3]. Although biopsy is still necessary and the definitive exam, 60% is a high rate of false positives.

One way to help reduce the rate of false positives is through automatic breast lesion detection based on features such as tumor shape, texture, and color. Most benign tumors have an elliptical boundary and uniform texture, while most malignant tumors have an irregular or spread-out boundary and texture. The feature detection aspects of Computational Vision with a sequence of ultrasound images are highly advantageous, mainly with 2D ultrasound images of low resolution and low SNR.

From the point of view of Computational Vision, automatic diagnosis of breast lesions can be divided in two phases: (1) detection of the region of interest (ROI), and (2) classification of the ROI based on medical classification schemes as a benign or malignant breast lesion. Since phase 2 is highly dependent on phase 1, the algorithms for shape, texture, and color detection should have high robustness. The first phase is a step that demands much research time. The approaches found in the literature are highly machine dependent and noise sensitive. This is generally due to the fact that detection of an ROI is accomplished on an independent slice, and the information from one frame to the next is not taken into account, so that some features are missing. One way to reduce this dependence and take advantage of the data from the previous slice is employment of a tracking algorithm.

One typical and important application that requires specific solutions is tracking a lesion area in a breast ultrasound image sequence (see, e.g., Figure 1) where

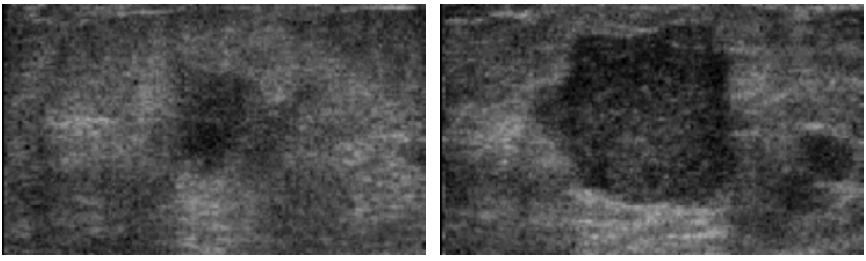


Figure 1. Example of two desired ultrasound images of a breast lesion for which our proposed methodology was developed.

images have a low signal-to-noise ratio (SNR) and a high time step between consecutive frames, and there is no need for real-time computation.

Such an application fits some of the objectives and features described above. Our objectives include tracking objects with highly deformable shape that during sequence evolution suffer transformations in translation, rotation, scale, split, or merge. In addition, there is no need to manage occlusion or background shift or more than one object in the scene. Other conditions include the following: no need for real-time computation, a low SNR, and a high time step between consecutive images, which produces difficulties when traditional techniques of object correspondence are used. On the other hand, the low SNR presents problems in terms of parameter setup that demand a robust segmentation algorithm.

One natural approach to handling these problems is to establish up a ground truth region in the first frame and compute a correspondence problem between consecutive images. This setup may be achieved through manual segmentation or by using an automatic approach. In a simple way, the general problem can be stated as a problem of correspondence between two regions. Generally, however, a segmentation process is required before achieving correspondence. As the objectives and features become more and more complex, the segmentation algorithm needs to be increasingly robust and the whole process becomes more and more dependent on initial definition of parameters (e.g., spatial filters and morphological operations).

In this context, segmentation methods that employ entropy applied to the object tracking problem have been broadly investigated. Traditional Shannon entropy is based on the achievement of a threshold between background and foreground for the purpose of maximization of image information. However, these methods are strongly dependent on image contrast and noise. In 1988, Constantino Tsallis [4, 5] presented the concept of non-extensive entropy for systems with long microscopic interactions and memory and fractal behavior. Such entropy expands the well-known Boltzman-Gibbs and Shannon entropy for extensive systems. The main advantage of this new theory is that it is based on a unique parameter, called q, which may be handled according to system non-extensiveness. The authors of [6] presented a first proposal for image segmentation using non-extensive entropy. In our work, we follow this proposal for region tracking in a breast lesion ultrasound image sequence. The task at hand is to establish a correspondence between regions of interest (objects in scenes).

In our proposed methodology we used the general idea of making a match between the region achieved in frame i (taken as a ground truth) and a candidate region achieved in frame $i+1$. We employ Tsallis entropy to find a candidate region. We do so because it generates good results for images with a low SNR, because of its simplicity of implementation, and because we have only one parameter to manage (q), which permits writing efficient algorithms. Each candidate region is smoothed, embedding the region boundary in a level set formulation with a narrow band. This has dramatically lowered computational load, as processing is carried

out only in the boundary area, called the "narrow band". In addition, this yields more accurate and smoother results. Matching is carried out through the use of the Hausdorff distance, which is employed due to its simplicity of implementation and the possibility of comparison between curves of different length.

The Hausdorff distance is a well-known method of matching two shapes not having the same number of points [7, 8]. However, it has high sensitivity to speckle noise and such other image artifacts as spurious and small regions. The main contribution of our present chapter is a methodology for tracking a general region of interest that combines the Hausdorff distance, a level set formulation, and Tsallis non-extensive entropy in order to reduce Hausdorff noise sensitivity as well as avoiding the increased parameter dependence that generally occurs with other methodologies.

The chapter is organized as follows. In Section 2 we briefly discuss the principal works related to our approach. We give some background on the main approaches we have used in Section 3. The proposed Hausdorff-Tsallis level set algorithm is described in Section 4, while our experimental results are reported in the following section. Section 6 presents a discussion of our results, and we present our conclusions in Section 7.

2. RELATED WORK

Recently, Xu [7] and collaborators proposed a method for detection of moving objects that employed the Hausdorff distance and segmentation based on watershed approaches. Their method involves three steps: first, the target objects are extracted with the use of only the first two frames. This initial extraction is achieved through a split and merge of coherent regions over a quadratic tree structure.

With the extracted object taken as a model, their second step uses the Hausdorff distance to track objects of interest in the remainder frames. The object found in the last frame is used as a new model to find the object in the current frame. The model is then updated at each frame during the tracking task. This approach has the advantage of allowing the system to be immune from possible error propagation.

The final step uses a watershed algorithm, not for segmentation as usual, but for upper bounding the object boundary. This strategy avoids the influence of external regions, as well as noise, in object segmentation.

A major disadvantage of this approach, as outlined by the authors, is the fact that the proposed method fails to separate regions of low contrast. It is also sensitive to changes in illumination.

Another approach that uses the Hausdorff distance for object tracking is that proposed by Paragios and Deriche [8], which minimizes the interframe difference density function, approximated using a statistical model.

Kim and Hwang [9] presented a method for detecting moving objects in a video sequence. Their work employs the edge map difference between consecutive

frames to extract moving objects. In [10], Luo and collaborators extended the work of Kim and Hwang [9] by attaching a Dynamic Bayesian Network (DBN) for interpretation of the object of interest. When the object of interest is extracted from the background, it is split into four quadrants (labeled I, II, III, and IV), and several attributes are extracted under the proposed DBN. This method was applied to a sports video database and achieved up to 86% accuracy.

Gao and collaborators [11] investigated detection of human activity through tracking of moving regions. The tracking process is achieved by segmenting dominant moving regions inside the ROI from where spatial features are obtained. These features are used to describe the related activities. An algorithm, called *Weighted Sequential Projection* (WSR), is then applied to achieve consistent movements, and to place them on a list of candidate regions. The idea underlying WSR is the calculus of the correlation of two vectors for consecutive regions of consecutive frames. The consistent regions are found based on the comparison between these two vectors.

The process of identifying human activities is achieved in three steps: first, an individual person is found by attaching segmented regions frame to frame; second, the face is found using the algorithm derived by Sheidermann [12]; finally, the hands are found through analysis of movements. Movements of the hands in relation to the head are analyzed to detect, for example, a human dining.

The paper presented by Lim and Kriegman [13] offers a tracking process for people walking around. The process uses two types of models for the tracking algorithm: one off-line, called the shape model, and one on-line, called the appearance model. The off-line model is based on learning of several generic human poses; the on-line model is based on the individual visual features that are being tracked. The combination of these two models composes the final tracking algorithm. In addition, during the off-line step, the background is learned trough the Mahalanobis distance. Since this is a statistical distance, it can be updated on-line.

The general idea combines both the shape and the appearance models as follows: at the on-line step the shape model is used to capture moving people. The appearance model is then used to fine tune the candidate regions.

The recent paper by Yeasin and collaborators [14] presents a framework for tracking several objects in a frame sequence whose principal contribution is a strategy for reducing the problem of the conflict of trajectories, which happens when two individual objects cross each other. In this case, a simple algorithm has no way of distinguishing both objects. The strategy of Yeasin and collaborators [14] is to use *Multiple Hypotheses Tracking* (MHT) algorithm combined with a *Path Coherent Function* (PCF) to handle the conflict of trajectories. Since MHT suffers from real-time implementation due to the exponential growth in the number of hypotheses, the PCF is implemented to correct real-time object position, reducing the number of hypotheses. This results is enhancement of the MHT algorithm. An efficient implementation of the MTH algorithm can also be found in the work of Cox and Hingorani [15].

Rodrigues and Araújo [16] studied object recognition based on traditional morphological segmentation. The contribution of this work is a new algorithm, based on region segmentation, that extracts spatial features and placing them into a directed graph. This methodology was applied for Content-Based Image Retrieval.

Recently, we introduced work [17, 18] on using a Bayesian Network for parameter estimation in a framework for object recognition in an augmented reality environment. We used feature extraction to generate several candidate regions in a frame. The proposed Bayesian Network was then applied to select the target object from among all candidates.

All works presented in this section (except for [16]) are from 2004 and discuss other related approaches. They also handle the tracking problem. Despite application for tracking people or general objects, they may use learned models (off-line) or statistical models (on-line) based on some kind of segmentation. Therefore, the segmentation process is crucial for tracking procedures due to changes in illumination. However, approaches to handle this problem remain an open problem and demand further investigation. Aside from these issues, the segmentation problem applied for a tracking problem using Tsallis non-extensive entropy has not investigated yet. That is the main contribution of the present work.

Computer-aided diagnostic systems for breast lesion identification and classification have been the object of study for many years. Several semiautomatic as well as completely automatic methods have been proposed. Semiautomatic methods include those based on texture information, while completely automatic methods include those based on such morphologic information as roundness, length, and area. The main problems one faces are machine dependency and noise sensitivity. Both 2D and 3D scanning devices are faced with these problems.

Chen et al. [1] presented work on classification of breast lesions (benign or malignant tumors) on sonograms using a Feed-Forward Neural Network (FFNN). The authors argue that the main difficulties of such an application is the morphological constraints of the lesion (which depend on the feature extraction approach) and the region information (which depends on device acquisition and the specific setup). In order to be device independent, their proposed method avoids region information, being based only on seven morphological features, which in turn are used as inputs to the FFNN. However, it is highly sensitive to speckle noise.

To avoid dependency on the feature extraction algorithms, Chen et al. [19] stated a semiautomatic method where a physician first might trace the lesion boundary area, and then a neural network classifies between benign and malignant tumors based on information from the physicians. However, their proposed approach is machine dependent and will run only on devices manufactured in the United States.

Garra et al. [20] analyzed 72 images from an ultrasound image sequence of several tumors. They used fractal analysis and a co-occurrence matrix for texture information and evaluated the value of such information in distinguishing between benign and malignant breast lesions. One drawback is that texture is dependent on machine setup.

Even when a non-automatic method is proposed, such as by Sawaki et al. [21], the morphological features are fundamental to breast lesion classification. They proposed a CAD system using fuzzy inference for breast sonography and evaluated the performance of their system. To accomplish the diagnostics, the radiologists needed to trace the lesion boundary, which is also machine dependent and noise sensitive.

Chang et al. [22, 23] presented work which used a learning vector-quantized model with 24 autocorrelation texture features to classify tumors as benign or malignant.

Drukker et al. [24] used radial gradient index (RGI) filtering, an average radial gradient (ARG), round-robin analysis, and a Bayesian Neural Network to automatically detect breast lesion ultrasound images.

Horsch et al. [25] presented a CAD method for breast lesion classification on ultrasound based on four features related to lesion shape, margin, texture, and posterior acoustic behavior.

Chen et al. [26] employed texture parameters of a region of interest and a decision-tree model to classify breast lesion tumors as benign and malignant. They also proposed [27] a method using morphological features extracted with a level set formulation of breast lesion tumors, and used a support vector machine to classify benign and malignant cancers.

Kuo et al. [28] used texture analysis and data mining with a decision tree model to classify breast lesions with different US systems.

3. BACKGROUND

We now present a review of the three main theories underlying our proposed approach for region tracking of breast lesions in an ultrasound image sequence: the Hausdorff distance, the level set theory, and Tsallis non-extensive entropy.

3.1. The Hausdorff Distance

The Hausdorff distance is a measure between two sets, not necessarily with the same dimensions. It measures how a subset of points of set A is near to a subset of distinct set B [29], [30], and has been successfully used in many applications [30–32].

Despite the possibility of matching two data sets of different dimensions, other advantages of the Hausdorff distance are its simple implementation and low computational load. However, it runs in quadratic polynomial time, which may yield slow application if this problem is not managed. To overcome this, we can reduce the total number of points in the sets to be processed, thus reducing its resolution. This can result in quick implementation (see [32]). The main drawback of the Hausdorff distance is that it may render applications sensitive to noise and outliers.

Another disadvantage of the Hausdorff distance is its sensitivity to rotation and deformation. in Addition, if a point is a noise or outlier, the resulting of Hausdorff distance will be very large, even if all other points match perfectly. In the presence of rotations, for example, different Hausdorff distance results may be obtained.

Care must thus be taken to avoid the influence of noise and outliers, increased computational time, and the effects of rotation. And if such care is taken, the Hausdorff distance can produce good results in applications that require the matching of two sets of different dimensions, such as a model and a target region in a tracking application for an ultrasound image sequence of breast lesion anomalies.

Formally, given two point sets A and B, the Hausdorff distance between them is defined as

$$H(A, B) = max(h(A, B), h(B, A)), \tag{1}$$

where

$$h(A, B) = max_{a \in A} min_{b \in B} \|a - b\|. \tag{2}$$

$\| \bullet \|$ represents some underlying norm defined in the space of the two point sets, which is generally required to be an L_p norm, usually the L_2 or Euclidian norm. The function $h(A, B)$ is called the direct Hausdorff distance from A to B.

Intuitively, if $h(A, B) = d$, each point in A must be within a distance d of some point in B.

The maximum value for the Hausdorff distance of two regions A and B in an image with dimension $M \times N$ is half of its diagonal, $\eta = \sqrt{M^2 + N^2}/2$. Therefore, to obtain the Hausdorff distance between 0 and 1, we normalize Eq. (1) by η, and define the following extensible Hausdorff distance:

$$H_e(A, B) = 1 - \frac{H(A, B)}{\eta}. \tag{3}$$

In our application, we propose a region tracker that matches a Model Region against a Target Region in subsequent frames by minimizing the Hausdorff distance [29], [7].

3.2. Tsallis Entropy

Entropy is an idea born under classical thermodynamics, not as something intuitive, but as mainly quantitative, defined by an equation. However, we may say that it is a concept that is associated with the order of irreversible processes. Physically, it may be associated with the amount of disorder in a physical system. Shannon has redefined the concept of Boltzmann-Gibbs entropy as an uncertainty measure associated with the content of information in a system. In this theory, the entropy of a discrete source is often obtained from a probability distribution $P = (p_1, \ldots, p_k)$, $0 \leq p_i \leq 1$, and $\sum_i p_i = 1$, where p_i is the probability of finding the system in state i. In this context, Shannon entropy (BGS) may be

described as

$$S = -\sum_i p_i \ln(p_i). \tag{4}$$

Generally speaking, systems that have BGS-type statistics are called extensive systems. Such systems have an additive property, defined as follows. Let X and Y be two random variables, with probability density functions $P = (p_1, \ldots, p_n)$ and $Q = (q_1, \ldots, q_m)$, respectively. If X and Y are independent, under the context of Probability Theory, the entropy of the composed distribution will verify the so-called additivity rule:

$$S(\{p_i q_j\}_{i,j}) = S(\{p_i\}_i) + S(\{q_j\}_j). \tag{5}$$

This traditional form of entropy, called Boltzmann-Gibbs-Shannon (BGS) entropy is well known and for years has achieved relative success in explaining several phenomena *if* the effective microscopic interactions are *short*-ranged (i.e., close spatial connections), *and* the effective spatial microscopic memory is *short*-ranged (i.e., *close* time connections), *and* the boundary conditions are *non-(multi)fractal*. Roughly speaking, the standard formalism is applicable whenever (and probably only whenever) the relevant space-time (hence the relevant phase space) is non-(multi)fractal. If this is not the case, some kind of extension appears to become necessary [4]. In complete analogy with Newtonian mechanics, it is an approximation (and an increasingly bad one) when the involved velocities approach the speed of light or the masses are as small as, say, the mass of an electron, and standard statistical mechanics do not apply when the above requirements (short-range microscopic interactions, short-range microscopic memory, and (multi)fractal boundary conditions) are not the case. However, recent developments based on the concept of non-extensive entropy, also called Tsallis entropy, have generated new interest in the study of Shannon entropy for Information Theory [5, 6, 33]. And this interest appears mainly due to the similarities between the functions of Shannon and Boltzmann-Gibbs entropy. Tsallis entropy (or q-entropy) is a new proposal for generalization of traditional Boltzmann-Gibbs entropy applied to non-extensive physical systems.

The non-extensive characteristic of Tsallis entropy has been applied through inclusion of parameter q, which generates several mathematical properties, such as non-negativity, concavity, equiprobability, q-additivity, and q-axiomality of a group. These characteristics give q-entropy flexibility in explaining several physical systems. On the other hand, this new kind of entropy does fail to explain traditional physical systems, as it is a generalization.

However, a generalization of a theory may suppose violation of one of its postulates. In the case of the generalized entropy proposed by Tsallis, the additive property described by Eq. (5) is violated in the form of Eq. (6), which applies if the system has a non-extensive characteristic. In this case, Tsallis statistics are useful and q-additivity better describes the composed system. In our case,

the experimental results (Section 5) show that it is better to consider our systems non-extensive:

$$\begin{aligned} S_q(\{p_i q_j\}_{i,j}) = S_q(\{p_i\}_i) + S_q(\{q_j\}_j) \\ +(1-q)\cdot S_q(\{p_i\}_i)\cdot S_q(\{q_j\}_j). \end{aligned} \tag{6}$$

In this equation, term $(1-q)$ represents the degree of non-extensiveness: in the limit $q \to 1$, $S(X+Y)$ meets the BGS entropy. The extensive character also can be seen in the general equation of entropy proposed by Tsallis:

$$S_q(p_1, \ldots p_k) = \frac{1 - \sum_{i=1}^{k} (p_i)^q}{q-1}, \tag{7}$$

where k is the total number of possibilities of the system, and real number q is the entropic index that characterizes the degree of non-extensiveness. As state above, in the limit $q \to 1$, Eq. (7) meets the traditional BGS entropy defined by Eq. (4).

Considering $S_q \geq 0$ in the pseudo-additive formalism of Eq. (6), the following classification for entropic systems is defined:

- Subextensive entropy $(q > 1)$
  $S_q(X+Y) > S_q(X) + S_q(Y)$
- Extensive entropy $(q = 1)$
  $S_q(X+Y) = S_q(X) + S_q(Y)$
- Superextensive entropy $(q < 1)$
  $S_q(X+Y) < S_q(X) + S_q(Y)$

Taking into account the similarities between the formalisms of Shannon and Boltzmann-Gibbs entropy, it is interesting to investigate the possibility of generalization of Shannon entropy to the case of information theory, as was recently shown by Yamano [34]. This generalization may be extended to image classification systems by applying Tsallis entropy, which has nonadditive information contents.

We propose here to segment an image using q-entropy to search the ROI, as will be explained in Sections 3.2 and 4. The motivations to employ q-entropy are: (1) managing only a simple parameter q opens up the possibility of simply applying several segmentations and later choosing the one that generates the best results; (2) as suggested in [6], the mammographic images and possibly several other medical images include non-extensive behavior; and (3) it is simple to make the implementation easy and it also ensures low computational overload.

3.2.1. Entropy Segmentation

Entropy is a well-known strategy for image segmentation that has been successfully used [35–38] for years since Pun [39] showed how to segment an image

maximizing Shannon entropy (Eq. (4)). In this chapter we use Tsallis entropy to generate candidate regions in each image from the sequence. The manner in which we use this entropy is the one proposed by Tsallis [4, 5] and applied by Albuquerque et al. [6] on mammographic images.

If we consider two random variables, X and Y, as stated above, and letting P be the probability density function of the background and Q the probability density function of the foreground (ROI), with constraints $\sum_{i=1}^{t}(\frac{p_i}{N}) = 1$ and $\sum_{i=t+1}^{L}(\frac{p_i}{M}) = 1$, and the assumption that t is a threshold, M is the total amount of background pixels, N is the total amount of foreground pixels, and the discrete levels are equally spaced between 1 and L in gray-scale level, the entropy of the composed system, considering the systems as non-extensive, according to Eq. (6), we have

$$\begin{aligned} S_q(X+Y) = S_q(X) + S_q(Y) \\ +(1-q)\cdot S_q(X)\cdot S_q(Y), \end{aligned} \tag{8}$$

where

$$S_q(X) = \frac{1-\sum_{i=1}^{t}(\frac{p_i}{N})^q}{q-1} \tag{9}$$

and

$$S_q(Y) = \frac{1-\sum_{i=t+1}^{L}(\frac{p_i}{M})^q}{q-1}. \tag{10}$$

We maximize the information measured between the two classes (ROI and background). When $S_q(X+Y)$ is maximized, luminance level t is considered to be the optimum threshold value. This can be achieved with a comparatively cheap computational effort:

$$t_{opt} = \text{argmax}[S_q^X(t) + S_q^Y(t) + (1-q)\cdot S_q^X(t)\cdot S_q^Y(t)]. \tag{11}$$

Therefore, it is interesting to investigate the images under the light of the non-extensive entropy. Although it will be difficult to discuss the non-extensive features (long-range and long-memory interactions and fractal behavior) for images in general, we can quantitatively justify the use of q-entropy for image segmentation, as the q parameter introduces a flexibility in the entropic formulation, and this suggests an investigation of segmentation under q variation. Albuquerque et al. [6] have studied q for mammographic images and achieved good results for $0 \leq q \leq 1$, which suggests a non-extensiveness for systems composed of such images.

In our work, we suggest the use of q-entropy to designate candidate regions of interest. The general idea is to generate some perturbation around the q value (used to generate the ROI in slice i) to achieve an ROI in slice $i+1$. Then a level set formulation is used to smooth the boundary and matching is accomplished through the Hausdorff distance. A more detailed algorithm will be set forth in Section 4.

3.3. The Level Set formulation

A level set method, also called an "implicit snake," is a numerical technique developed first by Osher and Sethian [40] to track the evolution of interfaces. These interfaces can develop sharp corners, break apart, and merge. The technique has a wide range of applications, including fluid mechanics, combustion, manufacture of computer chips, computer animation, snowflake structure, and soap bubble shape, and image processing, including medical images.

Informally speaking, let a boundary image interface separating one region of interest (inside) from another (outside), and a speed function F that gives the speed of each point on the boundary. This speed can depend on several physical effects, including temperature, if the environment is icy (ROI) inside the water. In this case, the boundary can shrink as the ice melts, or grow as it freezes. Speed function F then depends on the adopted model. However, the original level set formulation proposed by Osher and Sethian [40] takes a curve and builds it into a surface. The curve is evaluated into the surface that intersects the xy plane exactly where the curve sits. The surface is called the level set function because it accepts as input any point in the plane and feeds back its height as output. The curve that intersects the surface is called the zero level set, because it is the collection of all points at height zero. A complete level set study can be found in [41].

The advantages of level set formulation in image segmentation include how it handles naturally complex topology, how it permits flexible models representing an ROI boundary, how robust it can be to noise, and its computational efficiency.

In our work, we use a level set formulation to smooth the ROI before a matching phase (see Section 4 for a better explanation of the matching process). To enhance computational overhead, we compute the level set only inside a narrow band, and the initial curve is given by the previous segmentation with Tsallis entropy.

More formally, the main idea of the 2D level set method is to represent the deformable surface (or curve) as a level set $\{x \in \Re^2 | G(x) = 0\}$ of an embedding function:

$$G : \Re^2 \times \Re^+ \to \Re, \tag{12}$$

such that the deformable surface (also called the *front* in this formulation) at $t = 0$ is given by a surface S:

$$S\,(t = 0) = \left\{x \in \Re^2 | G\,(x, t = 0) = 0\right\}. \tag{13}$$

The next step is to find an Eulerian formulation for the front evolution. Following Sethian [42], let us suppose that the front evolves in the normal direction with velocity $\overrightarrow{F}$, which may be a function of the curvature, normal direction, etc.

We need an equation for the evolution of $G\,(x, t)$, considering that surface S is the level set given by

$$S\,(t) = \left\{x \in \Re^2 | G\,(x, t) = 0\right\}. \tag{14}$$

Let us take a point $x(t)$, $t \in \Re^+$ of propagating front S. From its implicit definition, given the above, we have

$$G(x(t), t) = 0. \tag{15}$$

We can now use the Chain Rule to compute the time derivative of this expression:

$$G_t + F|\nabla G| = 0, \tag{16}$$

where $F = \left\| \overrightarrow{F} \right\|$ is called the *speed function*. An initial condition $G(x, t = 0)$ is required. A straightforward (and expensive) technique to define this function is to compute a signed-distance function as follows:

$$G(x, t = 0) = \pm d, \tag{17}$$

where d is the distance from x to the surface $S(x, t = 0)$ and the signal indicates if the point is interior (–) or exterior (+) to the initial front.

Finite-difference schemes based on a uniform grid can be used to solve Eq. (16). The same entropy condition of T-Surfaces (*once a grid node is burnt, it stays burnt*) is incorporated in order to drive the model to the desired solution (in fact, T-Surfaces was inspired by the level sets model [43]).

In this higher-dimensional formulation, topological changes can be efficiently implemented. Numerical schemes are stable and the model is general in the sense that the same formulation holds for 2D and 3D, as well as for merge and splits. Besides, the surface geometry is easily computed. For example, the front normal and curvature are given by

$$\overrightarrow{n} = \nabla G(x, t), \quad K = \nabla \cdot \left(\frac{\nabla G(x, t)}{\|\nabla G(x, t)\|} \right), \tag{18}$$

respectively, where the gradient and divergent ($\nabla \cdot$) are computed with respect to x.

Initialization of the model through Eq. (17) is computationally expensive and not efficient if we have more than one front to initialize [44].

The *narrow-band* technique is much more appropriate for this case. The key idea of this technique comes from the observation that the front can be moved by updating the level set function at a small set of points in the neighborhood of the zero set instead of updating it at all points on the domain (see [42,45] for details).

To implement this scheme we need to preset a distance Δd to define the narrow band. The front can move inside the narrow band until it collides with the narrow-band frontiers. Then function G should be reinitialized by treating the current zero set configuration as the initial one.

This method can also be made cheaper by observing that the grid points that do not belong to the narrow band can be treated as sign holders [42], in other words, points belonging inside the narrow band are positive, and negative otherwise.

To clarify ideas, let us consider Figure 2a, which shows the level set bounding the search space, and Figure 2b, which pictures a bidimensional surface where the zero level set is comprised of the contours just presented.

(a) (b)

Figure 2. (a) Level set bounding the search space. (b) Initial function with zero level set as the contour presented.

The surface evolves such that the contour gets closer to the edge. In order to accomplish this goal, we must define a suitable speed function and an efficient numerical approach. For simplicity, we consider the one-dimensional version of the problem depicted in Figure 2. In this case, the evolution equation can be written as

$$G_t + \frac{\partial G}{\partial x} F = 0. \tag{19}$$

The main point is to design speed function F so as to obtain the desired result, which can be accomplished if $G_t > 0$. For instance, if we set the sign of F opposite to the one of G_x, we get $G_t > 0$:

$$G_t = -\frac{\partial G}{\partial x} F. \tag{20}$$

Hence, the desired behavior can be obtained by the distribution of the sign of F shown in Figure 3.

Once our application focus is shape recovery in image I, we must choose a suitable speed function F as well as a convenient stopping term S to be added to the right-hand side of Eq. (19). Among the possibilities [46], the following ones have been found to be suitable in our case:

$$F = \frac{1 + \alpha k}{1 + |\nabla I|^2}, \tag{21}$$

$$S = \beta \nabla I \cdot \nabla G, \tag{22}$$

where β is a scale parameter. Therefore, we are going to deal with the following level sets model:

$$G_t = \left(\frac{1 + \alpha k}{1 + |\nabla I|^2} \right) |\nabla G| + \beta \nabla P \cdot \nabla G, \tag{23}$$

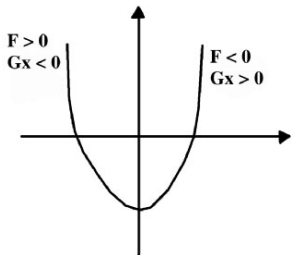


Figure 3. Sign of the speed function.

where $P = -|\nabla I|^2$. After initialization, the front evolves following Eq. (23). Next, we develop the numerical elements for the level set implemented herein.

3.3.1. Numerical Methods

Expression (23) can be written in general form as

$$G_t + H(G_x, G_y) = 0, \tag{24}$$

where H may be a non-convex *Hamiltonian*. Therefore, following [45], we can use the first-order numerical scheme given by

$$G_{i,j}^{n+1} = G_{i,j}^{n} + \triangle t \cdot F \parallel \nabla G_{i,j}^{n} \parallel, \tag{25}$$

where

$$\nabla G_{i,j}^{n} = [(\frac{G_{i+1,j}^{n} - G_{i-1,j}^{n}}{2 \cdot \triangle x}), (\frac{G_{i,j+1}^{n} - G_{i,j-1}^{n}}{2 \cdot \triangle y})], \tag{26}$$

and

$$\parallel \nabla G_{i,j}^{n} \parallel = \sqrt{(\frac{G_{i+1,j}^{n} - G_{i-1,j}^{n}}{2 \cdot \triangle x})^2 + (\frac{G_{i,j+1}^{n} - G_{i,j-1}^{n}}{2 \cdot \triangle y})^2}. \tag{27}$$

In Eqs. (25)–(27), n stands for the evolution in time n, and $\triangle t$ is the time step defined by the user.

3.3.2. Initialization

Discontinuities in image intensity are significant and important when it comes to shape recovery problems. In fact, such discontinuities of image I can be modeled as step functions, like

$$s\,(x, y) = \sum_{i=1}^{n} a_i S_i\,(x, y)\,, \tag{28}$$

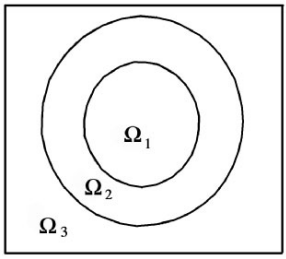


Figure 4. Support of step function given by Eq. (28) with $a_i = 1$, $a_2 = -1$ and $a_3 = 1$.

where S_i is a step function defined on a closed subset Ω_i of the image domain, which means that

$$\begin{aligned} S_i(x,y) &= 1 \quad if \quad (x,y) \in \Omega_i, \\ S_i(x,y) &= 0 \quad otherwise. \end{aligned} \tag{29}$$

Therefore, an image I is not a differentiable function in the usual sense. The usual way to address this problem is to work with a coarser-scale version of the image obtained by convolving this signal with a Gaussian kernel:

$$I_g(x,y) = K_\sigma(x,y) * I(x,y), \tag{30}$$

where $K_\sigma(x,y)$ is a Gaussian function with standard deviation σ. Therefore, we must replace function I by I_g in Eq. (23). This point is interesting for our context due to initialization of the level sets approach. A very simple way to do that would be through step functions like Eq. (28). For instance, Figure 4 shows such an example, based on the sign distribution of Figure 2a, in which we decompose the image domain in three closed subsets: Ω_1, Ω_2, and Ω_3. The step function would be given by Eq. (28) with $a_i = 1$, $a_2 = -1$, and $a_3 = 1$.

Such a proposal is computationally efficient but cannot be accepted due to discontinuities in function s. From a functional analysis viewpoint, we should use a differentiable function to initialize the method. We can address this problem following the same idea behind Eq. (30); that is, we can define a step function through Eq. (28), and then take a convoluted version of it obtained by replacing image I by function $s(x,y)$ in Eq. (30). Such a smoothed function will be suitable for our purposes, as we shall show in Section 5. If we add boundary conditions to Eq. (23), we get a parabolic problem that may have complex behaviors from a numerical viewpoint.

3.3.3. Narrow Band and Termination Condition

As usual with level sets approaches, we are using the narrow-band method, so that only the values of G within a tube placed around the front are updated. This simplified approach can drop the complexity of the algorithm from $O\left(N^2\right)$, in three dimensions, to $O\left(Mn\right)$, where m is the number of cells in the width of the narrow band [45]. When the front moves near the edge of the tube boundary, the computation is stopped and a new tube is built with the zero level set at the center.

To construct the narrow band we define two polygons. Each polygon is associated with an element of the curve that approximates the zero level set of the previous iteration. Each edge and vertex defines two polygons, for the inside and outside parts of the narrow band. Edge polygons are quadrilaterals, build by a shift of the edge in the direction of the normal vector, by a displacement defined by the width of the narrow band. Similarly, at each vertex we build triangles, using the normal vectors of each adjacent edge and with a height defined also by the width of the narrow band. This approach is similar to the procedure used by Maunch [47] and Peikert [48] to calculate the Euclidean Distance Function using a scan process. Our scan process is done by the GPU of a video card, in the rasterization pipeline phase. The result of the render process is returned to the application as a texture, where the cells in the narrow band are associated with the pixels generated by each polygon rasterization.

4. PROPOSED HAUSDORFF-TSALLIS LEVEL SET ALGORITHM

In this section we present our proposed framework, which combines the Hausdorff distance, non-extensive entropy, and a level set formulation for tracking regions representing a breast lesion in a frame sequence of ultrasound images.

The general idea for tracking adopted for this work, which is well known, is to use a region achieved in image i of a sequence to search a corresponding region in image $i+1$. We then propose a framework combining the Hausdorff Distance, Tsallis entropy, and a level set formulation to match region i and a candidate region.

As stated in Section 3.2, Tsallis entropy is a new trend in image segmentation that is simple and can be well adapted to real-time applications. In turn, the Hausdorff distance is a well-known similarity measure between two curves with such advantages as simplicity of implementation and low computational time (see Section 3).

In our notation, a frame has index i and specific segmentation index j. Assuming that we have several segmentations on each image, $I_{i,j}$ is the jth segmentation of the ith frame. By using *set notation*,

$$I_s = \{I_{i,1}, I_{i,2}, \ldots, I_{i,m}\} \tag{31}$$

are the m segmented outputs of frame I_i. Each $I_{i,j}$ may contain several segmented regions — among foreground and background ones —, called Target Regions

(TRs). Then, $r_{i,k} \in I_{i,j}$ is the kth TR of segmented image $I_{i,j}$. For the sake of explanation, we consider that segmented frame $I_{i,j}$ has only one TR of interest (the foreground), called r_i; the remaining ones belong to the background. Finally, a model for tracking a region is denoted by $\mathcal{MR}$.

By the way, there is a need for MO initialization in the first image; however, there are several approaches to doing so, which is generally a step before the tracking task. We assume that model region $\mathcal{MR}$ was already defined in the first frame. However, some investigators [9, 11, 15, 17–18] offer other techniques for such initialization.

Our proposed HTLS tracking algorithm is the following: let $\mathcal{MR}_{i-1}$ be the model region achieved in image $i-1$. For image I_i we compute I_s given in Eq. (31) with the Tsallis entropy according to Eq. (11), where each $I_{i,j}$ is computed with a different value of q. Each $I_{i,j}$ has a candidate region r_j. We extract r_j from $I_{i,j}$ by applying mathematical morphology according to [49].

The region given by the morphological operators is the input for the level set framework discussed in Section 3.3. This framework, with the added advantages of the narrow band, has the net effect of smoothing the contour region, approaching each contour pixel independently to the real edge.

Finally, each r_j is matched against $\mathcal{MR}_{i-1}$. $\mathcal{MR}_i$ is that region that minimizes Eq. (2).

To drop m (the total number of segmentations over each image I_i), we choose a set of values of q around that used in the previous image. Experimentally, we have seen that $m \simeq 3$ is sufficient to achieve good results; the proposed method then does not generate computational overhead due to applying segmentation m times to each image. On the other hand, this approach allows the proposed method to be robust to noise and changes in illumination conditions, as Tsallis entropy for segmentation slightly adapts to new conditions from one image to another throughout the sequence. This is the key idea of our work: the parameter setup for the first slice is automatically updated along the entire image sequence. There is then no need for user intervention as context, noise, or illumination change. A schematic view of the proposed HTLS algorithm can be seen in Figure 5.

5. EXPERIMENTAL RESULTS

The proposed method was experimentally tested over three classes of image sequences and compared to the Optical Flow Approach. The first class includes real office scenes, where we can see a telephone around other objects on a table with a white wall. The camera moves, generating a completely moving scenario. This is the same as moving the object across a moving background. An example of this sequence is shown in Figure 6.

In this class of sequence images we can note a complex background as the camera moves. There are several objects in the scene: the telephone (ROI) has

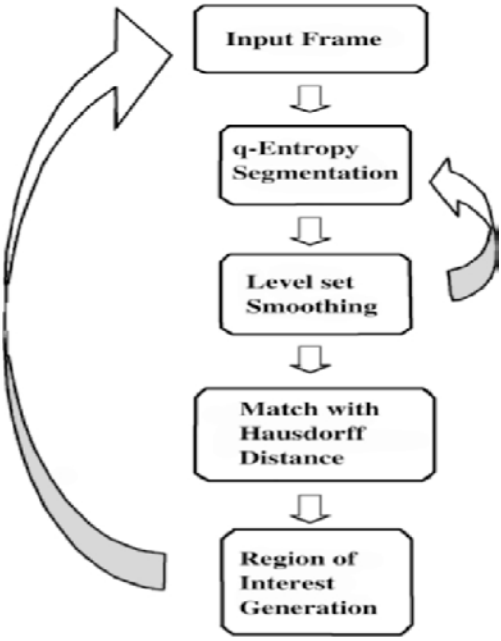


Figure 5. Schematic view of the proposed HTLS algorithm. The initial user-defined ROI is not shown.

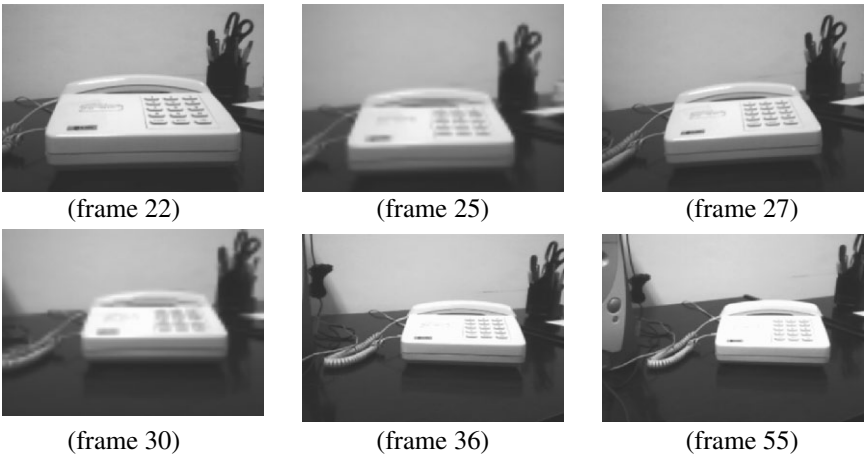


Figure 6. Sequence of images from an office table with slight camera movement, changes in illumination, and a heterogeneous background. See attached CD for color version.

basically the same color as the wall, and the entire background moves together, which generates difficulties for tracking.

A second class of images is from the Columbia database [50], which contains 7200 color images of isolated objects (100 objects taken at 5 degrees incremented in pose = 72 shots per object). This database is suitable for evaluating the performance of our system for object tracking. As we have several object viewers with zoom, this is equivalent to having a unique object moving over a homogeneous background, which simplifies segmentation. An example of some object sequence from this database can be seen in Figure 7.

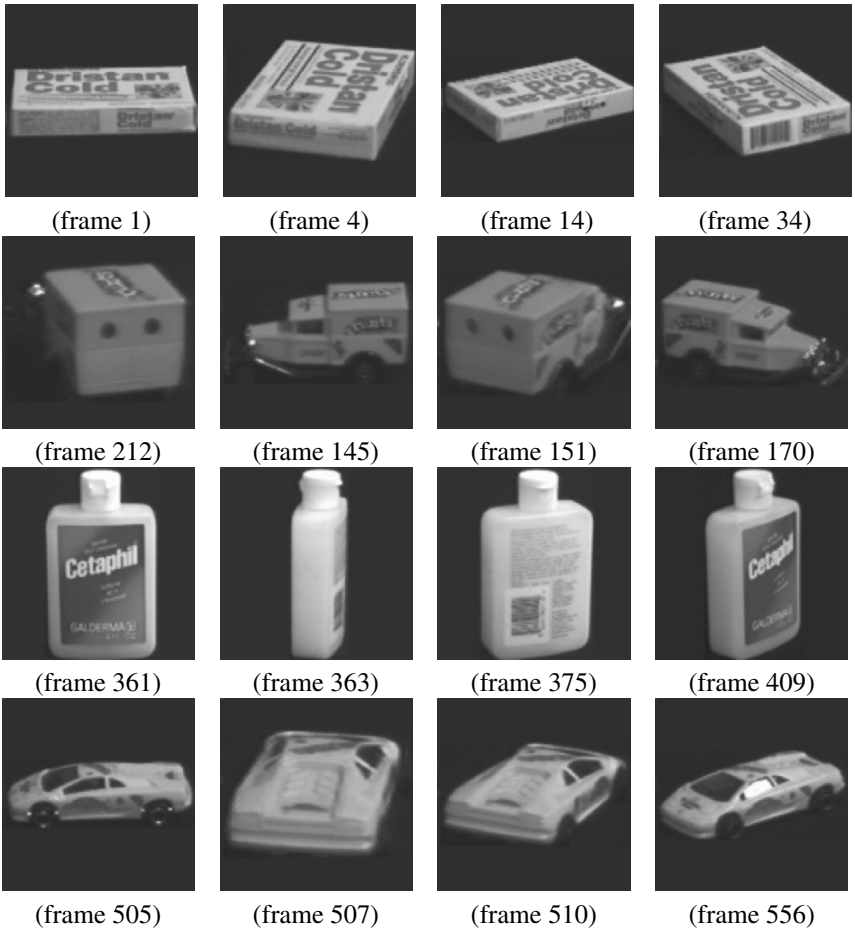


Figure 7. Four sequences of images from the Columbia database. Each row is an object sequence. This database has several object viewers and zoom, which is suitable for evaluating the performance of our proposed algorithm.

The third and last class of images we used in our experiments includes ultrasound images. There were 140 ultrasound images with low SNR that also were individually affected by speckle noise. We used 80 images of benign tumors and 60 of malignant tumors, divided into 7 sequences. Each sequence corresponded to a different case and had 20 images. We carried out our experiment only on those images where a tumor appeared. In the first slice of each sequence, we manually traced the ROI. The main reason we chose this kind of image was to show the performance of our algorithm under conditions of speckle noise. Although this sequence of images was taken from real clinical exams of cancerous breast lesions (which included several malignant and benign images), the purpose of this work is not to classify the achieved region of interest as malignant or benign. The images were thus not classified by any expert, such as a breast lesion tumor radiologist. Rather, we assumed that the ROI in each image is the darker region around the center of each image, without any care as to whether it of the correct shape for a breast tumor. The advantages of testing our proposed algorithm with this class of images is that we could trace the darker central region (ROI) throughout a low-SNR context; in addition, the lesions have marked topological changes over the evolution that are suitable for testing our proposed methodology. An example of a sequence of images from this class is shown in Figure 8.

For each of the above classes we carried out experiments employing our proposed algorithm with and without the level set formulation and compared its performance against the Optical Flow Approach (OFA). The reason we chose the optical approach is that it is a well-known and often-used method for tracking moving objects since being proposed [51].Since then, several variations have been proposed for a wide range of applications. We match the automatically generated ROI against ground truth images manually segmented, and measure the curve distances with the Polyline Distance Measure (PDM), which has become a standard approach for matching two curves since it was proposed for use with medical images [52, 53]. In the following, we will give in brief explanation about the OFA and PDM approaches.

5.1. Optical Flow Approach (OFA)

We now briefly describe the Optical Flow Approach (OFA) to calculating optical flow between two images, as we have implemented it, for comparison with our proposed algorithm.

Most work devoted to motion estimation use the term "optical flow." Optical flow is defined as an apparent motion of image brightness. Let $I(x, y, t)$ be the image brightness that changes in time to provide an image sequence. Two main assumptions can be made:

- Brightness $I(x, y, t)$ smoothly depends on coordinates x and y in the greater part of the image.

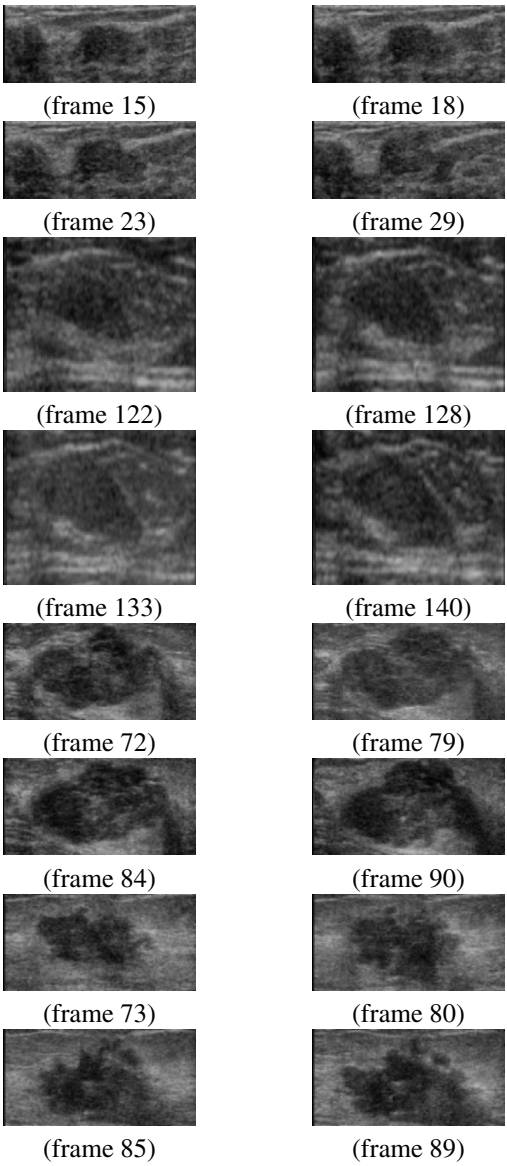


Figure 8. Four sequences of images from the Columbia database. Each row is an object sequence. This database has several object viewers and zoom, which is suitable for evaluating the performance of our proposed algorithm.

- The brightness of every point of a moving or static object does not change in time.

Let some object in the image, or some point of an object, move; after time dt the object displacement is (d_x, d_y). Using a Taylor series for brightness $I(x, y, t)$ gives the following:

$$I(x + d_x, y + d_y, t + d_t) = I(x, y, t) + \frac{\partial I d_y}{\partial x} + \frac{\partial I d_y}{\partial y} + \frac{\partial I d_y}{\partial t} + ..., \quad (32)$$

where "..." are higher-order terms.

Then, according to Assumption 2,

$$I(x + d_x, y + d_y, t + d_t) = I(x, y, t), \quad (33)$$

and

$$\frac{\partial I d_y}{\partial x} + \frac{\partial I d_y}{\partial y} + \frac{\partial I d_y}{\partial t} + ... = 0. \quad (34)$$

Dividing Eq. (33) by d_t and defining

$$\frac{\partial d_x}{\partial d_t} = u, \frac{\partial d_x}{\partial d_t} = v \quad (35)$$

gives

$$\frac{\partial I}{\partial t} = \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v, \quad (36)$$

usually called the *optical flow constraint equation*, where u and v are components of the optical flow field in x and y coordinates, respectively. Since Eq. (36) has more than one solution, more than one constraint is required.

5.2. Polyline Distance Measure (PDM)

In order to compare a computed-extracted boundary with the ideal boundary, a quantitative error measure based on the average polyline distance of each point was developed by Suri et al. [52–55]. The polyline distance is defined as the closest distance from each estimated boundary point to the ideal/ground-truth ROI boundary. The closest distance of each estimated boundary point can be the perpendicular distance (shortest Euclidian distance) to one skin-line, or it can be one of the end boundary points joining the points of the closest interval. In the following paragraphs, we will derive the PDM mathematically.

Let B_1 be the first boundary, and B_2 the second boundary. Let the Cartesian coordinates of point A on B_1 be (x_0, y_0). Let there be two successive boundary points, B and C, given by coordinates (x_1, y_1) and (x_2, y_2) on B_2. Let λ be the

free parameter for the equation of the line joining points B and C. Then, line interval BC between B and C is given as

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \lambda \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \end{pmatrix} \tag{37}$$

where (x, y) is the coordinate of a point on the line and $\lambda \in [0, 1]$.

Now, let μ be the parameter of the distance orthogonal to the line interval BC. Thus, the line segment between (x_0, y_0) and (x, y) is perpendicular to the line interval BC. Therefore, we can express (x_0, y_0) similar to Eq. 37:

$$\begin{aligned} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} &= \begin{pmatrix} x \\ y \end{pmatrix} + \mu \begin{pmatrix} -(y_2 - y_1) \\ x_2 - x_1 \end{pmatrix} \\ &= \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \lambda \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \end{pmatrix} + \mu \begin{pmatrix} -(y_2 - y_1) \\ x_2 - x_1 \end{pmatrix} \end{aligned} \tag{38}$$

Solving the above equation using the related determinants, unknown parameters λ and μ are given as

$$\lambda = \frac{(y_2 - y_1)(y_0 - y_1) + (x_2 - x_1)(x_0 - x_1)}{(x_2 - x_1)^2 + (y_2 - Y_1)^2} \tag{39}$$

and

$$\mu = \frac{(y_2 - y_1)(x_1 - x_2) + (x_2 - x_1)(y_0 - y_1)}{(x_2 - x_1)^2 + (y_2 - Y_1)^2}. \tag{40}$$

Let the two distance measures, d_1 and d_2, between A and B_1 and B/C on B_2 be defined as Euclidian distances:

$$\left\{ \begin{array}{l} d_1 = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2} \\ d_2 = \sqrt{(x_0 - x_2)^2 + (y_0 - y_2)^2} \end{array} \right\}. \tag{41}$$

The polyline distance, d_{poly}, is then defined as

$$d_{\text{poly}}(A, BC) = \left\{ \begin{array}{c} min\{d_1, d_2\}; \lambda < 0, \lambda > 1 \\ |d^{\top}|; 0 \leq \lambda \leq \end{array} \right\}, \tag{42}$$

where

$$|d^{\top}| = |\mu||BC| = \frac{(y_2 - y_1)(x_1 - x_0) + (x_2 - x_1)(y_0 - y_1)}{\sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)}}. \tag{43}$$

A quantitative error measure between the ideal boundary and the computer-estimated boundary could then be defined using the polyline distance described in

Eq. (42). The measure is defined as the average polyline distance of all boundary points of the estimated and ground-truth breast boundaries. We will denote the measure as $d_{\text{poly}}^{\text{Error}}$, which is derived as follows:

$$d_b(A, B_2) = min_{S \in sides B_2} d(A, S), \tag{44}$$

$$d_{vb}(B_1, B_2) = \sum_{A \in vertices B_1} d_b(A, B_2), \tag{45}$$

$$d_{\text{poly}}^{\text{Error}} = \frac{d_{vb}(B_1, B_2) + d_{vb}(B_2, B_1)}{\sharp vertices \in B_1 + \sharp vertices \in B_2}. \tag{46}$$

5.3. Experiments

Initially, we will demonstrate a performance comparison between the proposed algorithm with and without the level set framework regarding optical flow for the image sequence of the first class. We are considering a manmade ground truth, such as in Figure 9.

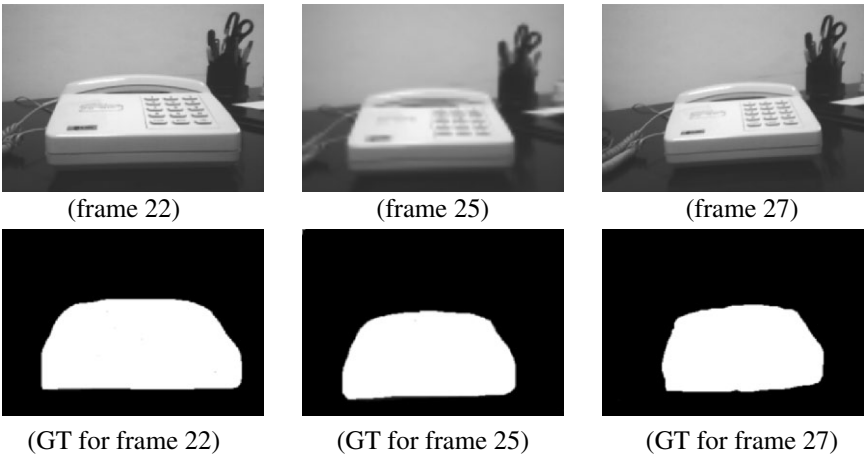


Figure 9. Three images (top row) from the first class and their corresponding manually traced ground truth (bottom row). GT = ground truth. See attached CD for color version.

For each of the images of the given sequence, the GT was manually traced for comparison with the automatic results. The similarity measure between the GT and an automatic generated curve was taken using Eq. (45). The performance was taken in evaluating a sequence of 30 frames (around 1 second each), is shown in Figure 10, where the normalized error (Nerr) is taken as $\text{Nerr} = 1 - \frac{1}{\eta}\text{Err}$, and Err is calculated by Eq. (45) and $\eta = \sqrt{M^2 + N^2}/2$ is the normalization factor taken

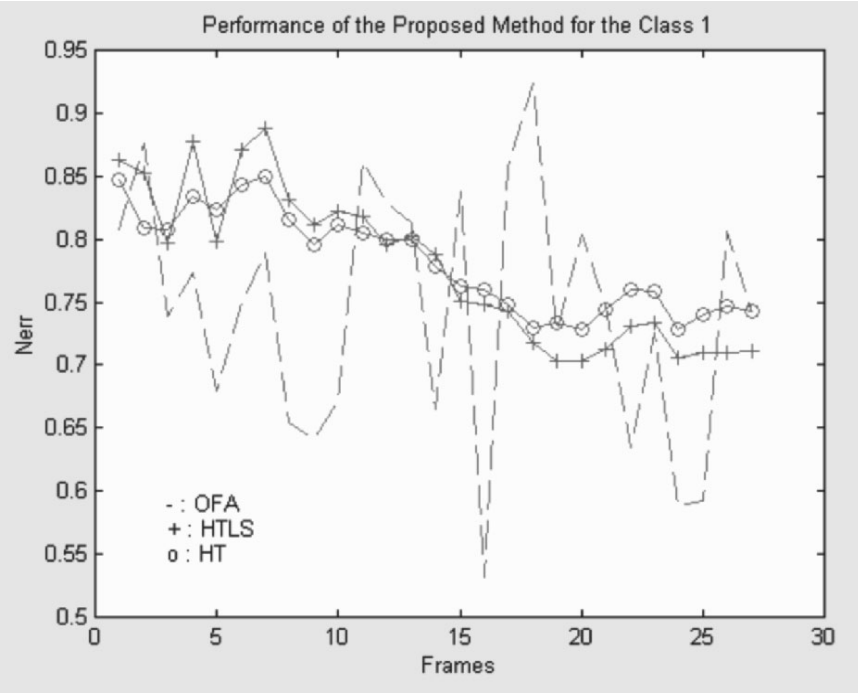


Figure 10. Performance evaluation of the HTLS algorithm as a function of frames for the first class sequence. "o" represents the HTLS performance; "+" represents HT algorithm performance; "–" represents OFA algorithm performance. See attached CD for color version.

as the higher possible error value, which depends on the corresponding image dimensions M and N. In Figure 10, note the similar performance of our method with and without the level set formulation. There is only a slight advantage for the HTLS algorithm. In this class of images, with a moving object background, the HTLS algorithm outperforms the optical flow approach. This is due to the fact that OFA does not work well when the background moves. However, the HTLS algorithm also has difficulties in tracking the ROI. This is because Tsallis segmentation can only deal with two regions in a scene — the background and the foreground — and in the class 1 images we have a wall with a similar gray-scale level to that of the ROI.

Another important observation is that the HTLS algorithm performance algorithm decreases with time. This is due to error propagation. A solution to this disadvantage is to periodically restart the system at different points. On the other hand, the optical flow approach is unstable. The same behavior we can note in Fig-

ure 11, which is with class 2 images. However, we can see that all the algorithms demonstrate good performance since the images are synthetic with a homogeneous background.

The performance evaluation for the class 2 sequence is shown in Figure 11, the same as in Figure 10.

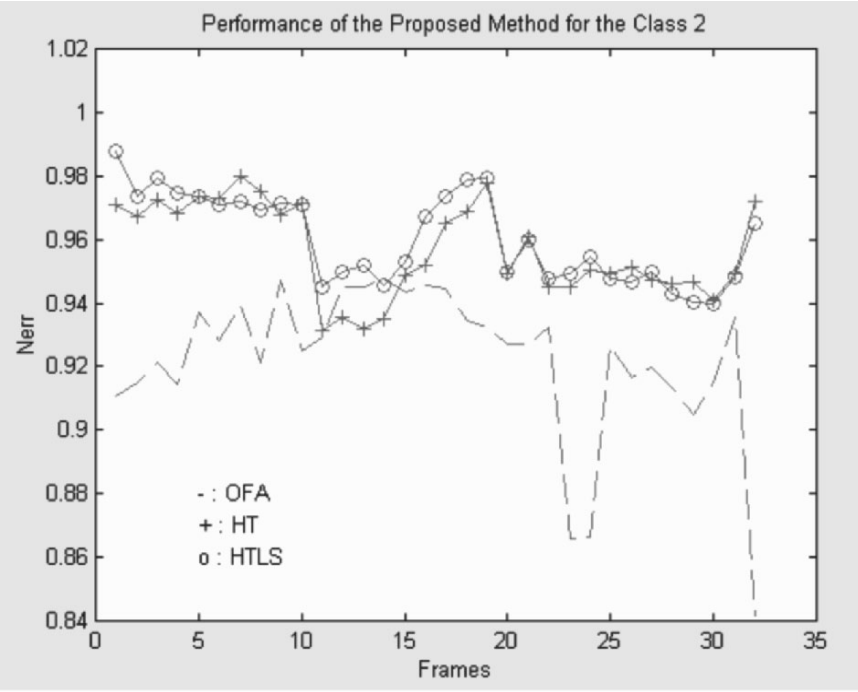


Figure 11. Performance of the HTLS algorithm as a function of frames for the class 2 sequence. "o" represents HTLS performance; "+" represents HT performance; "–" represents OFA performance. See attached CD for color version.

Figure 12 shows the performance curves for the class 3 images of benign tumor ultrasound images, and Figure 13 shows the same for malignant tumor ultrasound images.

The best performance of the HTLS algorithm over the optical flow approach can be seen when there are low-SNR images in the sequence. The OFA does not work well in this case. On the other hand, the HTLS algorithm (with and without the level set framework) can show good performance with both benign and malignant tumor images.

Before presenting examples of specific results with ultrasound images, we will show the usefulness of Tsallis entropy in Figure 14, where we can see an image

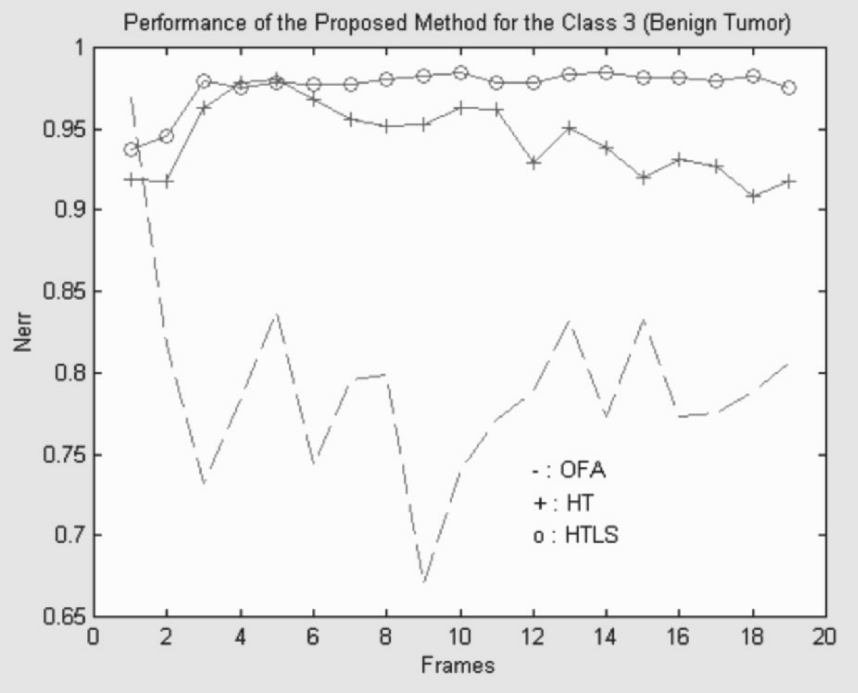


Figure 12. Performance of the HTLS algorithm for class 3 images (benign tumor). "o" represents HTLS performance; "+" represents HT performance; "–" represents OFA performance. See attached CD for color version.

from the sequence of a benign tumor and several segmentations for $0 < q < 1$ (subextensive systems). Figure 14a is the original image, and the lesion is the darker area traced manually as a white boundary around the image center. Below each segmented image is the q value employed. The same happens with Figure 15. The q values were varied from a value near 0 to a value close to 1. In this case, for values greater than 1 we did not achieve satisfactory results. In these two figures, even so, the better results are those for q values near 1. It is important to find an ideal q value, which may yield as crisp a boundary as possible. In the case of Figure 14, when q approaches 1, the ROIs seem to become well defined. The inverse seems to happen in Figure15, where better segmentation of the ROI seems to occur when $q = 0.00001$. However, in both figures it is not possible to point out the exact tumor boundary. An algorithm to define a boundary must take into account not only the boundary traced in the first slice but also an adaptive q value.

To reinforce this idea, if the same q value is used for all images of a sequence, the results may not be satisfactory, as is clearly shown in Figure 16.

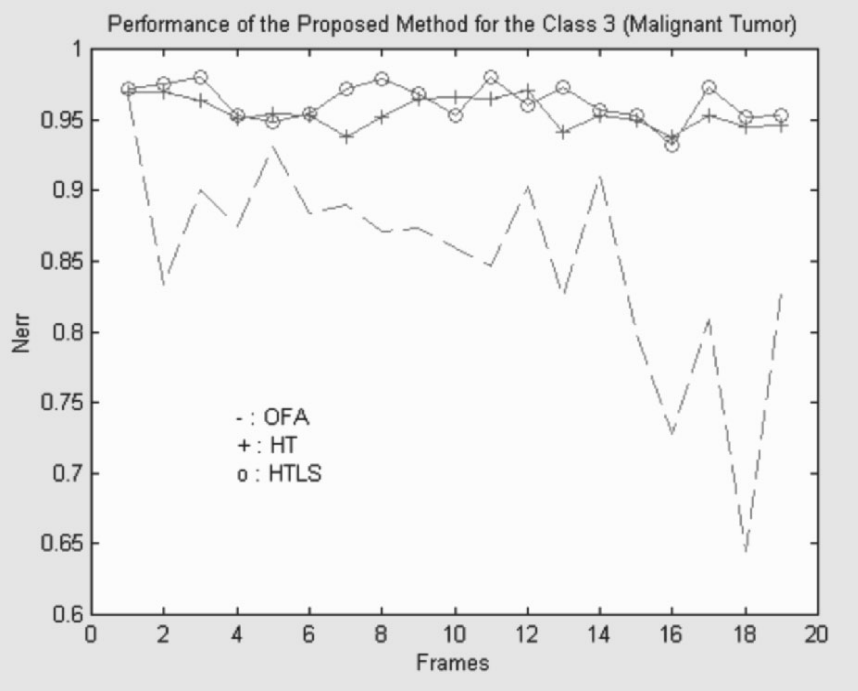


Figure 13. Performance of the HTLS algorithm with class 3 images (malignant tumor). "o" represents HTLS performance; "+" represents HT performance; "–" represents OFA performance. See attached CD for color version.

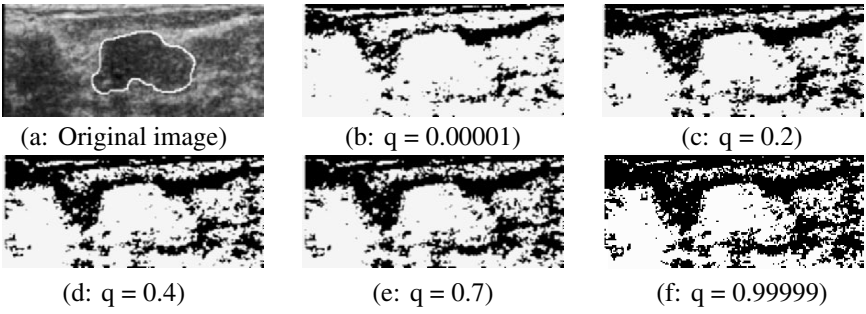


Figure 14. Mammographic image and five segmentations with different q values: (a) original image; (b)–(f) segmentation for $q = (0.00001, 0.2, 0.4, 0.7, 0.9999)$. Intuitively, in this case, as q approaches 1, the images seem to become well segmented.

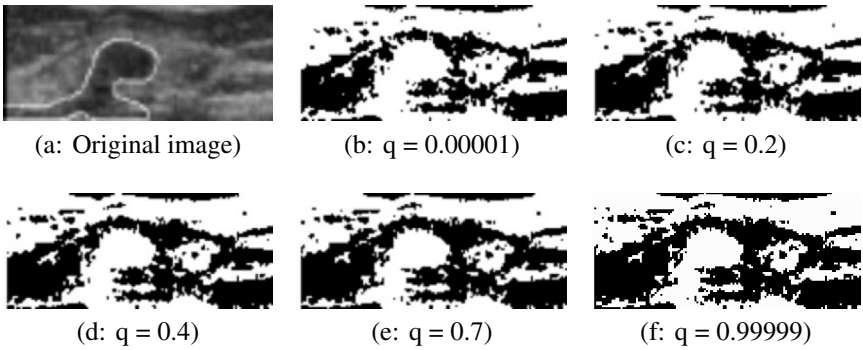


Figure 15. Mammographic image and 5 segmentation with different q values: (a) original image; (b)–(f) segmentation for $q = (0.00001, 0.2, 0.4, 0.9999)$. In this case, as q approaches 0, the images seem to become well segmented.

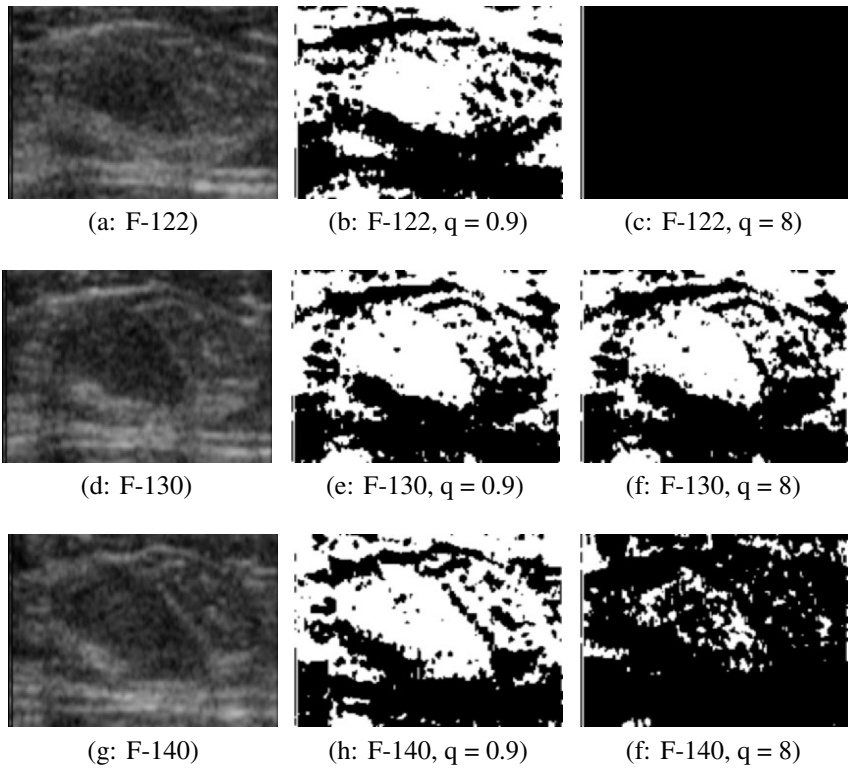


Figure 16. Frame sequence (frames 122, 130, and 140) of ultrasound images. Each row corresponds to a frame and two segmentations for $q = 0.9999$ and $q = 8$. For different frames we need different q values to achieve a good segmentation.

In this figure, the first column shows three original images (F-122, F-130, and F-140); the second column shows the respective segmentation for the same $q = 0.9$ (superextensive system) and the third column shows segmentation for $q = 8$ (subextensive system). We can note that, for frame 122, $q = 0.9$ generates a better segmentation than does $q = 8$ (where it is not possible to find any region). If we set the same $q = 0.9$ value for the remaining frames, when frame 130 arrives, we can see that $q = 8$ is more adequate. The inverse occurs again when frame 140 arrives, where $q = 0.9$ is better for segmentation than $q = 8$. These conditions occur due to the noise distribution, and therefore depend on image acquisition as well as on the selected target region.

Using non-extensive entropy on mammographic images may have advantages due to the flexibility of finding a good segmentation as well as the simplicity in managing only the q parameter, which is not possible with the other entropic theories (such as BGS) or those that depend on setting up several parameters, mainly when tracking a sequence of images. The disadvantage is that reaching an ideal value of q may not be an easy task. q is a value that, given a value of t, maximizes Eq. (8). One way to handle this problem is as follows. Given a t value that maximizes Eq. (8), we can have:

$$\frac{\partial S(X+Y)}{\partial q} = 0. \tag{47}$$

Therefore, an ideal q value may be found with an iterative procedure, such as the Newton-Raphsor method:

$$q^{n+1} = q^n - \frac{\frac{\partial S(X+Y)}{\partial q}}{\frac{\partial^2 S(X+Y)}{\partial q^2}}, \tag{48}$$

where n stands for the nth iteration.

However, in practical terms, and considering the problem at hand, we can save time by taking the value of q achieved in slice i and creating a small perturbation over it on slice $i+1$, testing each match between ROI_i and candidate region r_{i+1}, according to the algorithm presented in Section 4. The new q is that corresponding to the r_{i+1} that best matches ROI_i.

Following this idea, for $q = (0.0001, 0.1, 0.2, \ldots, 1.0)$, Figure 17 shows the results of applying the proposed method on a mammographic image sequence of 20 slices (only 8 are shown). The left-hand column shows some images from the original sequence, while the right-hand column shows their corresponding found ROI. Below each image on the right are the achieved q values.

Figure 18 shows the experiment for a sequence image of a malignant tumor. The main characteristic of this anomaly is its irregular shape (a benign tumor normally has an oval or ellipsoid shape). In the left-hand column are 8 of the original images from the sequence, and on the right are their corresponding traced

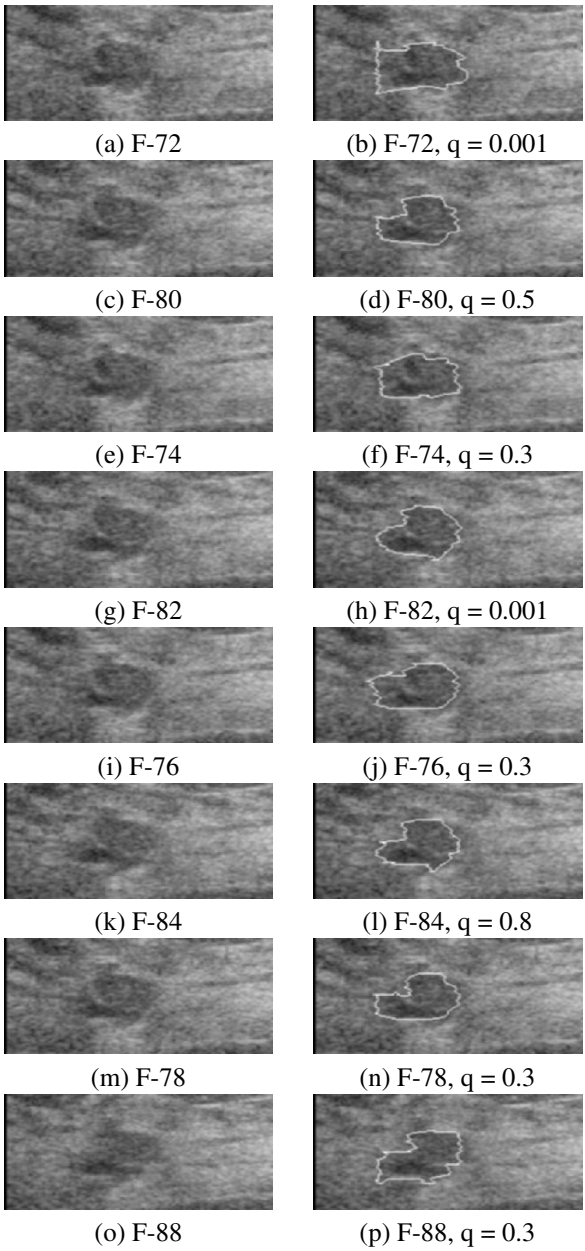


Figure 17. Eight images from a frame sequence of a benign tumor. The left column includes the original images; the right column presents their corresponding achieved boundary with their respective best q values (indicated below each image).

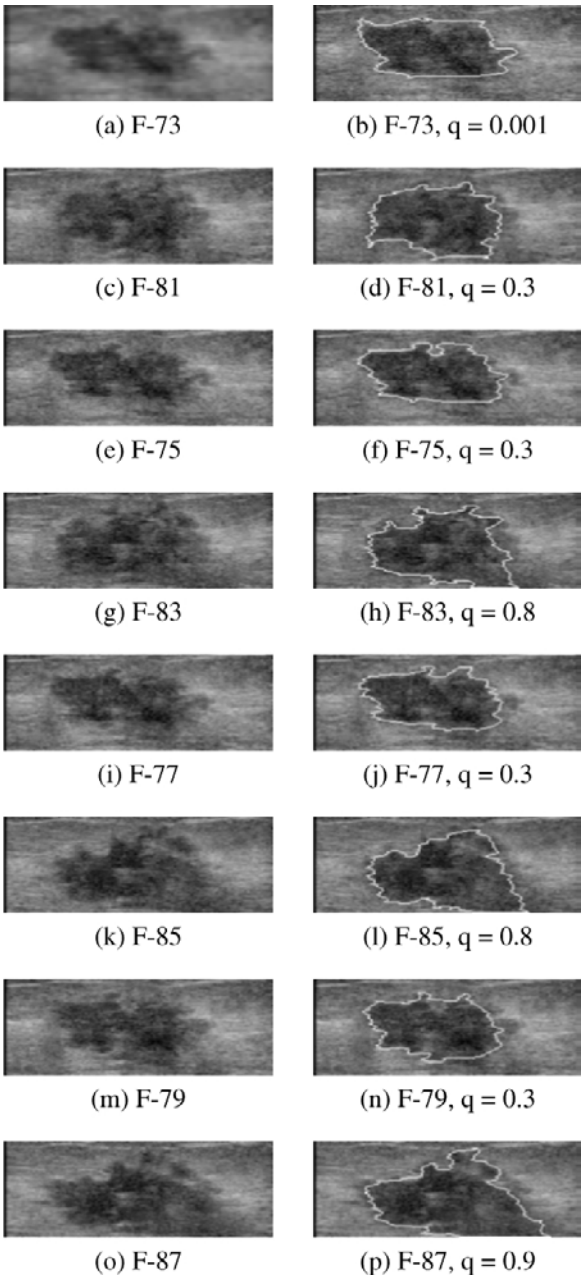


Figure 18. Eight images from a frame sequence of a malignant tumor. The left column includes the original images; on the right are their corresponding achieved boundary, with their respective best q values shown below.

shape and the q values achieved with the proposed method. In the following section we discuss these results.

6. DISCUSSION

Generally speaking, Object Tracking System is a two-step process — shape detection and tracking — and there is a strong demand for improvement in both steps. However, the work presented here focuses on both steps. There are several methods for recognizing the shape of an ROI, and as the noise and time step between consecutive frames are increased, algorithms that obtain general information from frame to frame (such as HTLS) become more and more useful. From the point of view of computational vision, in a frame sequence with a moving object at low SNR, some contours may be difficult (and even impossible) to detect. It is therefore important to try to get information from past images, where it is supposed that detection was reached adequately, as much as possible. In some cases, this is a unique alternative. The HTLS algorithm attempts to improve such region recognition, since, to our knowledge no satisfactory approach for tracking exists for ultrasound images. Another advantage of the HTLS algorithm is the small number of parameters that need to be managed (basically, the q parameter of Tsallis entropy and the time step of the level set framework). This makes HTLS algorithm setup a straightforward task.

In the specific case of q-entropy, the range we obtained in our experiments was 0 and 1. However, this value may be set differently for specific devices and conditions. And in the specific case of a time-step level set parameter, it influences how much the level set should run forward or backward toward the shape. If the time step is set at a small value, forwarding may run slowly; on the contrary, if it is set too long, forwarding could yield results far from the desired shape. In addition, since we are not dealing with real-time applications, this parameter may be set at a small value. In our case, we employed $\Delta t = 0.1$. However, the better the initial curve generated by q-entropy segmentation, the smaller the influence of the parameter will be.

One drawback of the HTLS algorithm, as is often the case with tracking algorithms based on matching objects from previous frames, is its error propagation, especially when the frame sequence is large. A popular approach to minimizing this problem is periodic user intervention (in a semiautomatic system). This occurs whenever the error ranges above a given threshold.

Finally, implementation of a system using the HTLS algorithm demands extension to track several regions in a scene and robustness with regard to topological changes. In the latter case, the method proposed by Giraldi et al. [56–59] may be a good solution for dual models.

7. CONCLUSIONS AND FUTURE WORK

We propose here a methodology for breast cancer lesion recognition. This method combines two frequently used techniques in image analysis — the Hausdorff distance and level set formulation — and also presents a new segmentation approach called q-entropy, which is based on non-extensive systems. This entropy can be applied to any class of images since it is a generalization of so-called BGS entropy.

Our results show that detection and outlining of regions of interest, even with speckle noise images, such as with malignant or benign cancer lesions, may be improved by combining information from the actual frames and past frames. The proposed method could possibly be extended to include cancer classification, first between malignant and benign tumors, and later, including more specialized classifications.

The use of a level set formulation was applied to smoothing shapes after non-extensive segmentation. However, the results with images such as those shown in Figures 18b, n, and p (where there are valleys which are then filled in by the level set frontier), demand improved formulation. One suggestion for this problem is to employ a dual level set or gradient vector flow combined with deformable models. A detailed discussion about these issues can be found in [60]. A T-Snake formulation, such as that discussed in [61], might also ameliorate topological problems.

Our general conclusion is that the HTLS algorithm performs best when the images have low SNR, such as with medical ultrasound images. As a future project, we will carry out experiments over a more complete database of ultrasound images, in which case a breast radiologist will be necessary to trace the ground-truth boundary.

8. ACKNOWLEDGMENTS

The authors are grateful for the support of CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico), a Brazilian agency for scientific financing. We also thank FAPERJ, Rio de Janeiro Stat's Agency for Scientific and Technological Development, by the financial support.

9. REFERENCES

1. Chen CM, Chou YH, Han KC, Hung GS, Tiu CM, Chiou HJ, Chiou SY. 2003. Breast lesion on sonograms: computer-aided diagnosis with neural setting-independent features and artificial neural networks. *J Radiol* **226**:504–514.
2. Zheng Y, Greenleaf JF, Gisvold JJ. 1997. Reduction of breast biopsies with a modified self-organizing map. *IEEE Trans Neural Networks* **8**:1386–1396.
3. Brown ML, Houn F, Sickles EA, Kessler LG. 1995. Screening mammography in community practice: positive predictive value of abnormal findings and yield of follow-up diagnostic procedures. *AJR Am J Roentgenol* **165**:1373–1377.

4. Tsallis C. 1999. Nonextensive statistics: theoretical, experimental and computational evidences and connections. *Braz J Phys* **29**(1):1–35.
5. Tsallis C. 2001. Nonextensive statistical mechanics and its applications. In *Lecture notes in physics*, Vol. 560, pp. 1128–154. Springer, Berlin. Bibliography of subjects is regularly updated at http://tsallis.cat.cbpf.br/biblio.htm.
6. Albuquerque MP, Albuquerque MP, Esquef IA, Mello ARG. 2004. Image thresholding using Tsallis entropy. *Pattern Recognit Lett* **25**:1059–1065.
7. Xu H, Younis A, Kabuka MR. 2004. Automatic moving object extraction for content-based applications. *IEEE Trans Circ Syst Video Technol* **14**(6):796–812.
8. Paragios N, Deriche R. 2000. Geodesic active contours and level sets for detection and tracking of moving objects. *IEEE Trans Pattern Anal Machine Intell* **22**:266–280.
9. Kim C, Hwang J. 2002. Fast and automatic video object segmentation and tracking for content-based applications. *IEEE Trans Circ Syst Video Technol* **12**(3):122–129.
10. Luo Y, Wu T, Hwang J. 2003. Object-based analysis and interpretation of human motion in sports video sequences by dynamic bayesian networks. *Comput Vision Image Understand* **92**(2-3):196–216.
11. Gao J, Hauptmann AG, Wactlar HD. 2004. Combining motion segmentation with tracking for activity analysis. In *Proceedings of the IEEE international conference on automatic face and gesture recognition (FGR'04)*, pp. 699–704. Washington, DC: IEEE Computer Society.
12. Sheidermann H, Kanade T. 2000. A statistical method for 3d object detection applied to faces and cars. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, Vol. 1, pp. 746–751. Washington, DC: IEEE Computer Society.
13. Lim J, Kriegman D. 2004. Tracking humans using prior and learned representation of shape and appearance. In *Proceedings of the IEEE international conference on automatic face and gesture recognition (FGR'04)*, pp. 869–874. Washington, DC: IEEE Computer Society.
14. Yeasin M, Polat E, Sharma R. 2004. A multiobject tracking framework for interactive multimedia applications. *IEEE Transaction on Multimedia* **6**(3):398–405.
15. Cox IJ, Hingorani SL. 1996. An efficient implementation of Reid's multiple hypotesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Trans Pattern Anal Machine Intell* **18**:138–150.
16. Rodrigues PS, Albuquerque A. 2002. A region-based object recognition algorithm. In *Proceedings of the 15th Brazilian symposium on computer graphics and image processing (SIBGRAPI 2002)*, pp. 283–290. Washington, DC: IEEE Computer Society.
17. Rodrigues PS, Rodrigues DMC, Giraldi GA, Silva RLS. 2004. A Bayesian network model for object recognition environment using color features. In *Proceedings of the 17th Brazilian symposium on computer graphics and image processing (SIBGRAPI 2004)*. Electronic Proceedings.
18. Rodrigues PS, Rodrigues DMC, Giraldi GA, Silva RLS. 2004. Object recognition using Bayesian networks for augmented reality applications. In *Proceedings of the 7th symposium on virtual reality (SVR'04)*. Available at http://virtual.lncc.br/ diego/Artigos/Accepteds/SVR-2004.pdf.
19. Chen DR, Chang RF, Huang YL. 1999. Computer-aided diagnosis applied to solid breast nodules by using neural networks. *J Radiol* **213**:407–412.
20. Garra BS, Krasner BH, Horii SC, Ascher S, Mun SK. 1993. Improving the distinction between benign and malignant breast lesions: the value of sonographic texture analysis. *Ultrason Images* **15**:267–285.
21. Sakira A, Shikamoto K, Satake H, Ishigaki T, Koyama S, Obata Y, Ikeda M. 1999. Breast ultrasonography: diagnostic efficacy of computer-aided diagnostic system using fuzzy inference. *Radiat Med* **17**:41–45.
22. Chang RF, Kuo WJ, Chen DR, Huang YL, Lee JH, Chou YH. 2000. Computer-aided diagnosis for surgical office-based breast ultrasound. *Arch Surg* **135**(6):696–699.
23. Chang RF, Wu WJ, Moon WK, Chen DR. 2003. Improvement in breast tumor discrimination by support vector machines and speckle-emphasis texture analysis. *Ultrasound Med Biol* **29**(5):679–668.

24. Drukker K, Giger ML, Horsch K, Kupinski MA, Vyborny CJ, Mendelson EB. 2002. Computerized lesion detection on breast ultrasound. *Med Phys* **29**(7):1438–1446.
25. Horsch K, Giger ML, Venta LA, Vyborny CJ. 2002. Computerized diagnosis of breast lesions on ultrasound. *Med Phys* **29**(2):157–164.
26. Chen DR, Kuo WJ, Chang RF, Moon WK, Lee CC. 2002. Use of the bootstrap technique with small training sets for computer-aided diagnosis in breast ultrasound. *Ultrasound Med Biol* **28**(7):897–902.
27. Chen DR, Chang RF, Huang YL. 1999. Computer-aided diagnosis applied to solid breast nodules by using neural networks. *Radiology* **213**(2):407–412.
28. Kuo WJ, Chang RF, Moon WK, Lee CC, Chen DR. 2002. Computer-aided diagnosis of breast tumors with different us systems. *Acad Radiol* **9**(7):793–799.
29. Huttenlocher DP, Noh JJ, Rucklidge WJ. 1992. *Tracking non-rigid objects in complex scenes.* Technical Report 1320, Department of Computer Science, Cornell University.
30. Huttenlocher DP, Klanderman GA, Rucklidge WJ. 1993. Comparing images using the Hausdorff distance. *IEEE Trans Med Imaging* **15**(9):850–863.
31. Chalana V, Kim Y. 1997. A methodology for evaluation of boundary detection algorithms on medical images. *IEEE Trans Med Imaging* **16**(5):642–652.
32. Bamford P. 2003. Automating cell segmentation evaluation with annotated examples. In *Proceedings of the APRS workshop on digital image computing, (WDCI 2003)*, pp. 21–25. Available at http://www.aprs.org.au/wdic2003/CDROM/21.pdf.
33. Shannon C, Weaver W. 1948. *The mathematical theory of communication*. Urbana: U Illinois P.
34. Yamano T. 2001. Information theory based in nonadditive information content. *Entropy* **3**:280–292.
35. Kapur JN, Sahoo PK, Wong AKC. 1985. A new method for gray-level picture thresholding using the entropy of the histogram. *Comput Graphics Image Process* **29**:273–285.
36. Abutaleb AS. 1989. A new method for gray-level picture thresholding using the entropy of the histogram. *Comput Graphics Image Process* **47**:22–32.
37. Li CH, Lee CK. 1993. Minimum cross-entropy thresholding. *Pattern Recognit* **26**:617–625.
38. Sahoo PK, Soltani S, Wong AKC. 1988. A survey of thresholding techiniques. *Comput Vis Graphics Image Process* **41**:233– 260.
39. Pun T. 1981. Entropic thresholding: a new approach. *Comput Graphics Image Process* **16**:210–239.
40. Osher SJ, Sethian JA. 1988. Fronts propagation with curvature dependent speed: Algorithm based on Hamilton-Jacobi formulations. *J Comput Phys* **79**:12–49.
41. Sethian JA. 1999. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, 2nd ed. New York: Cambridge UP.
42. Malladi R, Sethian JA, Vemuri BC. 1995. Shape modeling with front propagation: a level set approach. *IEEE Trans Pattern Anal Machine Intell* **17**(2):158–175.
43. McInerney TJ. 1997. *Topologically adaptable deformable models for medical image analysis*. PhD dissertation, Department of Computer Science, University of Toronto.
44. ter Haar Romery BM, Niessen WJ, Viergever MA. 1998. Geodesic deformable models for medical image analysis. *IEEE Trans Med Imaging* **17**(4):634–641.
45. Sethian JA. 1996. *Level set methods: evolving interfaces in geometry, fluid mechanics, computer vision and materials sciences*. New York: Cambridge UP.
46. Suri JS, Liu K, Singh S, Laxminarayan S, Zeng X, Reden L. 2002. Shape recovery algorithms using level sets in 2-d/3-d medical imagery: a state-of-the-art review. *IEEE Trans Inf Technol Biomed* **6**(1):8–28.
47. Mauch S. 2000. *A fast algorithm for computing the closest point and distance function.* Technical Report, CalTech.

48. Sigg C, Peikert R. 2005. Optimized bounding polyhedra for gpu-based distance transform. *Proceedings of Dagstuhl seminar 023231 on scientific visualization: extracting information and knowledge from scientific data sets*. Berlin: Springer. Available at http://graphics.ethz.ch/ peikert/papers/dagstuhl03.pdf/.
49. Giraldi GA, Rodrigues PS, Marturelli LS, Silva RLS. 2005. Improving the initialization, convergence and memory utilization for deformable models. In *Segmentation models*, Vol. 1. Part A of *Handbook of Image Analysis*, Part A, 2nd ed., pp. 359–414. Berlin: Springer.
50. Columbia object image library (COIL-100). Department of Computer Science, Columbia University. http://www.cs.columbia.edu/CAVE/research/softlib/coil-100.html.
51. Lucas B, Kanade T. 1981. An iterative image registration technique with an application to stereo vision. In *Proceedings of the DARPA Image Understanding Workshop*, pp. 121–130. Washington, DC: US Government Printing Office.
52. Suri J. 1998. Error and shape measurement tools for cardiac projection images: a closer look. *Proceedings of an international conference in applications of patterns recognition*, pp. 125–134.
53. Suri J, Haralick RM, Sheehan FH. 1996. Greedy algorithm for error reduction in automatically produced boundaries from low contrast ventriculogram. *Proceedings of the international conference on pattern recognition (ICPR'96)*, Vol. 4, pp. 361–365. Washington, DC: IEEE Computer Society.
54. Suri J, Haralick RM, Sheehan FH. 2000. Greedy algorithm for error reduction in automatically produced boundaries from low contrast ventriculogram. *Int J Pattern Anal Appl* **3**(1):39–60.
55. Suri J, Setarhedran SK, Singh S. 2002. *Advanced algorithm approaches to medical image segmenttion: state-of-the-art applications in cardiology, neurology, mammography and pathology*. Berlin: Springer.
56. Giraldi GA, Oliveira AAF. 1999. Dual-snake model in the framework of simplicial domain decomposition. *Technical poster at the international symposium on computer graphics, image processing and vision (SIBGRAPI'99)*, pp. 103–106. Washington, DC: IEEE Computer Society.
57. Giraldi GA, Gonvalvez LMG, Oliveira AAF. 2000. Dual topologically adaptable snakes. In *Proceedings of the fifth joint conference on information sciences (JCIS 2000), third international conference on computer vision, pattern recognition, and image processing*, Vol. 2, pp. 103–106. Washington, DC: IEEE Computer Society.
58. Giraldi GA, Strauss E, Oliveira AAF. 2000. A boundary extraction method based on dual-t-snakes and dynamic programming. In *Proceedigns of the IEEE computer society conference on computer vision and pattern recognition, (CVPR 2000)*, p. 1044. Washington, DC: IEEE Computer Society.
59. Giraldi GA, Gonzalves LMG, Oliveira AAF. 2000. Automating cell segmentation evaluation with annotaded examples. In *Proceedings of the APRS workshop on digital image computing, (WDCI 2000)*, Vol. 2, pp. 103–107. Washington, DC: IEEE Computer Society.
60. Giraldi GA, Schaefer L, Rodrigues PS. 2005. Gradient vector flow models for boundary extraction in 2d images. In *Proceedings of the 8th international conference on computer graphics and image (IASTED CGIM 2005'*, p. 2000. Washington, DC: IEEE Computer Society.
61. Giraldi GA, Strauss E, Oliveira AAF. 2000. Boundary extraction approach based on multiresolution methods and the t-snakes framework. In *Proceedings of the 13th Brazilian symposium on computer graphics and image processing (SIBGRAPI 2000)*, pp. 82–89. Washington, DC: IEEE Computer Society.

15

DEFORMABLE MODEL-BASED IMAGE REGISTRATION

Jundong Liu

Ohio University, Athens, Ohio, USA

In this chapter we introduce the concept of deformable image registration and point out that *interpolation effects, non-rigid body registration, and joint segmentation and registration frameworks* are among the challenges that remain for this area.

To address the *interpolation effects* challenge, we choose the *Partial Volume Interpolator* (PV) used in multimodality registration as an example, and quantitatively analyze the generation mechanism of the interpolation artifacts. We conclude that the combination of linear interpolation kernel and translation-only motion leads to generation of the artifact pattern. As a remedy we propose to use nonuniform interpolation functions in estimating the joint histogram. The cubic B-spline and Gaussian interpolators are compared, and we demonstrate improvements via experiments on misalignments between CT/MR brain scans.

A segmentation-guided non-rigid registration framework is proposed to address the second and third challenges. Our approach integrates the available prior shape information as an extra force to lead to a noise-tolerant registration procedure, and it differs from other methods in that we use a unified segmentation + registration energy minimization formulation, and the optimization is carried out under the level-set framework. We show the improvement accomplished with our model by comparing the results with that of the Demons algorithm. To explore other similarity metrics under the same framework to handle more complicated inputs will be the focus of our future work.

1. INTRODUCTION

Medical image analysis technology, with image segmentation, image matching/registration, motion tracking, and measurement of anatomical and physiologi-

Address all correspondence to: Jundong Liu, Professor of Electrical Engineering and Computer Science, Stocker 321A, Ohio University, Athens, OH 45701. Phone: 740-593-1603, Fax: 740-593-0007. jliu@ufl.edu.

cal parameters as the main research areas has seen a tremendous amount of growth over the past decade. The work described in this chapter is concerned with the problem of automatically aligning 3D medical images.

Image registration is one of the most widely encountered problems in a variety of fields, including but not limited to medical image analysis, remote sensing, satellite imaging, optical imaging, etc. One possible definition of this problem is: determine the coordinate transformation, or mapping, relating different views of the same or similar objects. These views may arise from:

- The same object "imaged" with different sensors.
- The same object "imaged" repeatedly with the same sensor.
- Multiple similar objects all "imaged" with the same sensor.
- A single object "imaged" with a sensor, and a model (matching)

Medical imaging plays a very important role in modern clinical practice. Often, a single modality alone cannot provide adequate information about a patient's condition, so they are imaged by a second and sometimes more modalities. Different modalities, as shown in Figure 1 (from the homepage of the Image Science Institute at the University of Medical Center Utrecht; available online at http://www.isi.uu.nl/Research/Registration/
registration-frame.html), can usually provide different, complementary, or partially overlapping aspects of the anatomy under examination. So if the information from them can be combined, it will greatly facilitate the clinicians to perform surgical planning, diagnosis and treatment. Among the widely used modalities, X-ray, CT (computed tomography), MRI (magnetic resonance imaging), and US (ultrasound) depict anatomical information, while PET (positron emission tomography), SPECT (single-photon emission computed tomography), and fMRI (functional MRI) provide information on the metabolism of the underlying anatomy.

Figure 1 shows images from four different modalities—CT, MR, PET, SPECT —illustrating the different types of information they contain. As we can see, CT

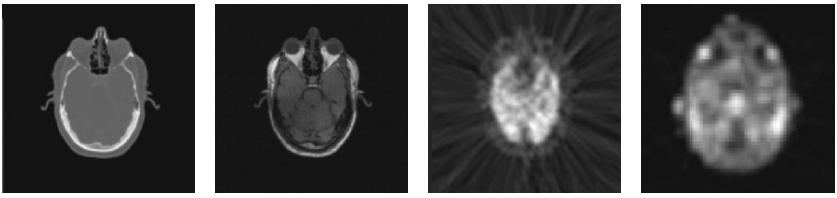


Figure 1. Images from different modalities. From left to right: CT, MR, PET, SPECT. Available online at http://www.isi.uu.nl/Research/Registration/.

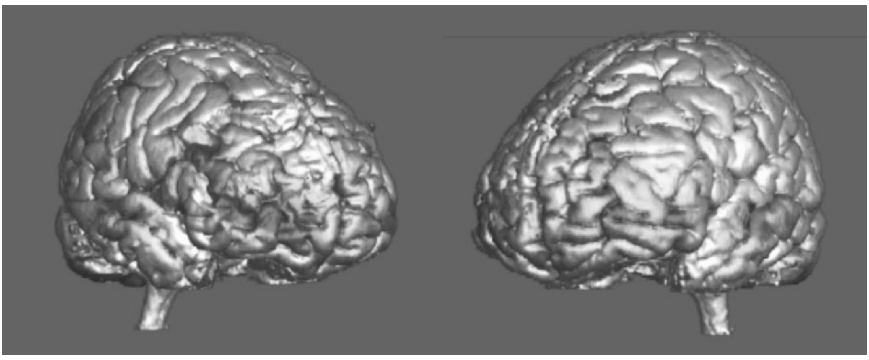


Figure 2. Example of a multimodal visualization. Available online at http://www.isi.uu.nl/Research/Registration/. See attached CD for color version.

is sensitive to bone density and MR to tissue density, whereas PET and SPECT depict the physiology.

Figure 2 (from the homepage of the Image Science Institute, Department of Medical Imaging; available online at http://www.isi.uu.nl/Research/Registration/registration-frame.html) also shows how multimodal data can be used to provide a better understanding of the physiology of the human brain aided by the presence of precise anatomical information. It shows the right and left hemispheres of a brain, segmented from an MR image. Information from a SPECT scan is overlaid on the cortex. Color encoding is used to indicate the amount of cerebral blood perfusion: from light gray for low perfusion, via yellow, to red for high perfusion. As we can see, the picture shows an area with increased blood perfusion in the right hemisphere, and this is indicative of pathology in the right hemisphere.

However, multimodality images are usually acquired with different devices, and at different times, so there will inevitably be some motion between them. This makes accurate geometrical registration a prerequisite for effective fusion of the information from multimodality images.

1.1. Mathematical Definition

Let $I_1(x, y, z)$ and $I_2(x, y, z)$ denote two images. The relationship between these two images can be expressed as

$$I_2(x, y, z) = g(I_1(T(x, y, z))), \tag{1}$$

where T is a 3D spatial coordinate transformation, and g is an intensity transformation.

The registration problem is to find the optimal spatial and intensity transformation so that the images are matched well. Finding the parameters of the optimal geometric coordinate transformation is generally the key to any registration problem, while the intensity transformation is not always the task of interest.

The transformations can be classified into global and local transformations. A global transformation is given by a single equation that maps the entire image. Local transformations map the image differently depending on spatial location, and are thus more difficult to express succinctly. The most common global transformations are rigid, affine, and projective transformations.

A transformation is called rigid if the distance between points in the image being transformed is preserved. A rigid transformation can be expressed as

$$\begin{cases} u(x,y) = (\cos(\phi)\ x - \sin(\phi)\ y + d_x) - x, \\ v(x,y) = (\sin(\phi)\ x + \cos(\phi)\ y + d_y) - y, \end{cases} \tag{2}$$

where $u(x,y)$ and $v(x,y)$ denote the displacement at point (x,y) along the X and Y directions; ϕ is the rotation angle and (d_x, d_y) the translation vector.

A transformation is called affine when any straight line in the first image is mapped onto a straight line in the second image with parallelism being preserved. In 2D the affine transformation can be expressed as

$$\begin{cases} u(x,y) = (a_{11}\ x + a_{12}\ y + d_x) - x, \\ v(x,y) = (a_{21}\ x + a_{22}\ y + d_y) - y, \end{cases} \tag{3}$$

where

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \tag{4}$$

denotes an arbitrary real-valued matrix. Scaling transformation, which has a transformation matrix of $\begin{pmatrix} s_1 & 0 \\ 0 & \hat{s}_2 \end{pmatrix}$, and shearing transformation, which has a matrix of $\begin{pmatrix} 1 & s_3 \\ 0 & 1 \end{pmatrix}$, are two examples of affine transformation, where s_1, s_2, and s_3 are positive real numbers.

A more interesting case, in general, is that of a planar surface in motion viewed through a pinhole camera. This motion can be described as a 2D projective transformation of the plane

$$\begin{cases} u(x,y) = \frac{m_0 * x + m_1 * y + m_2}{m_6 * x + m_7 * y + 1} - x, \\ v(x,y) = \frac{m_3 * x + m_4 * y + m_5}{m_6 * x + m_7 * y + 1} - y, \end{cases} \tag{5}$$

where m_0 ... m_7 are the global parameters.

In clinical practice, the most commonly used global registration transformations are rigid and affine. For the brain images taken from the same patient using

different devices, e.g., CT/MR, or the same devices but at different times, usually a rigid transformation is adequate to explain the variation between them. When two images depicting the same scene are taken from the same viewing angle but from different positions, i.e., the camera is zoomed in/out or rotated around its optical, an affine transformation is required to match these two images. In this thesis, we will mainly be dealing with these types of global transformations.

When a global transformation does not adequately explain the relationship of a pair of input images, a local transformation may be necessary. To register an image pair taken at different times with some portion of the body that has experienced growth, or to register two images from different patients, falls into this local transformation registration category. A motion (vector) field is usually used to describe the change/displacement in local transformation problem.

1.2. The Challenges Remaining

During the past two decades, a broad range of techniques has been developed to deal with the image registration problem for various types of data. For a comprehensive review, we refer the reader to [10]. Generally speaking, rigid registration of mono-modal images from the same patient will not solve the problem. Many methods, including the Fourier transform-based method or the voxel-similarity-based method, can align images of this type with subpixel accuracy. For registering images with significant contrast differences, such as those from different modalities or acquired with different matching MR sequences, the most effective matching criterion is the mutual information metric. With the advance in available computational power over the last few years, registration of non-rigid mono-modal or multimodal images has become a hot research area. One of the reasons why non-rigid registration has gained popularity is that registration can be used to solve various image segmentation problems. This strategy is called atlas-based segmentation.

The challenges remaining for image registration include [2]:

- *Interpolation effects:* Mutual information is the most popular similarity measure for registering multimodal images. However, together with most of the existing matching metrics, the Mutual Information metric is sensitive to image interpolation artifacts, which limits registration accuracy. To overcome this problem, special interpolators are needed.

- *Non-rigid body registration:* Nonrigid body image registration, in general, is still an open problem. Much work remains to develop practically useful deformable registration techniques.

- *Registration-assisted segmentation and segmentation-guided registration:* The traditional atlas-based segmentation approaches requires a two-stage philosophy: non-rigid registration between two images is first carried out,

and then the resulting transformation is applied to the atlas to obtain segmentation for the input image. The prior information embedded in the atlas is totally neglected during the registration step. Registration-assisted segmentation and segmentation-guided registration are directions worth pursuing.

1.3. Chapter Outline

The rest of this chapter is organized as follows: Section 2 reviews and analyzes the generation mechanism of the interpolation artifacts when using Mutual Information [14]. Section 3 reviews the related work for joint segmentation and registration problem. A variational framework that integrates prior segmentation information into the non-rigid registration procedure will be discussed. Section 4 concludes this chapter.

2. MUTUAL INFORMATION METRIC AND ARTIFACT EFFECTS [15]

This section is dedicated to the first challenge we pointed out in Section 1.2—*Interpolation effects*. We will take Mutual Information as an example to analyze the generation mechanism of the artifacts inherent in the interpolation procedure. Remedies to reduce the artifacts will also be presented.

Consider two images $I_r(x, y)$ and $I_f(x, y)$. We designate I_r as the reference image and I_f as the floating image. The registration task is to find the coordinate transformation, denoted as T, such that transformed floating image $I_f(T(x, y))$ is aligned with reference $I_r(x, y)$. The alignment is usually obtained by optimizing a certain similarity metric. So normally a registration algorithm consists of three components [3]: a coordinate transform, a similarity criteria, and a numerical scheme to seek the optimum.

Mutual information is currently the most popular matching metric used in handling the registration problem for multimodal images. The MI between two discrete random variables, A and B, is defined as [4]:

$$MI(A, B) = \sum_{a,b} p_{AB}(a, b) \log \frac{p_{AB}(a, b)}{p_A(a) \cdot p_B(b)}, \tag{6}$$

where $p_A(a)$, $p_B(b)$, and $p_{AB}(a, b)$ are the marginal probability distributions and joint probability distribution, respectively. The relationship between MI and entropy is

$$MI(A, B) = H(A) + H(B) - H(A, B), \tag{7}$$

with $H(A)$ and $H(B)$ being the entropy of A and B, and $H(A, B)$ their joint entropy:

$$\begin{aligned} H(A) &= -\textstyle\sum_a p_A(a) \log\ p_A(a), \\ H(A, B) &= -\textstyle\sum_{a,b} p_{AB}(a, b) \log\ p_{AB}(a, b). \end{aligned} \tag{8}$$

Given a set of samples, there are several approaches to estimating probability functions $p_{AB}(a, b)$, most notably the histogram-based method [5] and the Parzen window method [6, 7]. In this chapter we focus on histogram-based method because it is widely used in image registration. To register the images the mutual information needs to be maximized.

The advances in mutual information-based methods reside not only in the impressive accuracy in the reported registration experiments, but also the generality that MI methods can provide. Very few assumptions have ever been made regarding the relationship between image intensities, so mutual information is especially suited for multimodality matching, and that it is completely automatic.

Studholme et al. [8] pointed out that standard mutual information is sensitive to the field of view of the scanner used in the image acquisition, i.e., with different choices of field of view, the maximization process may lead to incorrect alignment. The authors then extended mutual information to a normalized mutual information to alleviate this problem:

$$NMI(A, B) = (H(A) + H(B))/H(A, B). \tag{9}$$

2.1. Interpolation Artifact Effects

For digital images $I_r(x, y)$ and $I_f(x, y)$ to be aligned, interpolation is necessary to evaluate the values of $MI(I_r(x, y)$ and $I_f(T(x, y))$. A number of interpolators are available, including nearest neighbor (NN), linear, cubic B-spline, Hamming-windowed sinc, and partial volume (PV) interpolators. Among them, PV is regarded as the best choice for MI-based metrics, as pointed out in several studies [9, 10].

Partial volume interpolation is not an interpolation in the ordinary sense. It is a strategy being used to update the joint histogram. As shown in Figure 3, instead of

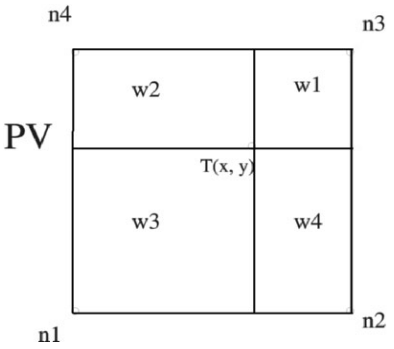


Figure 3. PV interpolation.

interpolating the new intensity value under current transformation T, PV directly updates the histogram of the nearest neighbor bins with the same weights used in bilinear (for 2D) or trilinear (for 3D) interpolation.

In [11] Maintz et al. qualitatively explained the reason why artifacts are generated in the partial volume interpolation process and verified their arguments through several well-designed experiments. While their work is very informative, we believe that a theoretically quantitative analysis concerning the generation mechanism of artifacts will be more instructive to guide related research.

As interpolation affects the registration function of normalized MI and traditional MI in a similar way [11], we will construct our arguments based on traditional MI here, but it should be noted that the conclusions also hold for normalized MI.

As given above, mutual information $MI(A, T(B))$ consists of 3 terms: $H(A)$, $H(B)$, and $H(A, T(B))$. $H(A)$ is a constant. The computation of $H(T(B))$ is also affected by the interpolation effect, but to a much smaller extent. Figure 4 shows a pair of aligned MR/CT images and the associated marginal entropies, joint entropy, and MI values as functions of translations up to ± 7 pixel distances. As is evident, the variation of MI is dominated by the changes in $H(A, T(B))$; $H(A)$ and $H(T(B))$ are close to constants. So from now on we will focus only on $H(A, T(B))$.

Let's first consider the situation where the reference and floating images have exactly the same pixel sizes and the motion is limited to translations only. We use the CT image in Figure 4 as the reference, while MR is the floating image. Now we analyze the variation in MI function between the two images when the floating image moves from the alignment position to 1 pixel away along the x axis.

Suppose at the alignment position (translation is equal to zero), a certain histogram bin $\mathrm{his}(a, b)$ has a value of M_1. $\mathrm{his}(a, b)$ records the total number of pixels in the image pair where the reference image has intensity a, and floating image b. Suppose when the translation is 1 pixel, $\mathrm{his}(a, b)$ becomes M_2. Let's redefine $\mathrm{his}(a, b)$ to $\mathrm{his}(a, b, t)$ to include the translation variable, then $\mathrm{his}(a, b, 0) = M_1$ and $\mathrm{his}(a, b, 1) = M_2$. In between, with the translation being t, there are a group of intensity grids $X1 = \{(x_1, y_1), (x_2, y_2)...\}$ that were originally contributing to bin $\mathrm{his}(a, b)$, that gradually wipe out their support when the floating image is moving. Let's call this group of grids as "moving-out set," and let A_1 be the total number of these grids. Because the motion here is limited to translation only, all the grids in the moving-out set are withdrawing their contributions to bin $\mathrm{his}(a, b)$ at the same rate, as the translation increases from 0 to 1. When the translation is 0, each of them contribute a $'1'$ to $\mathrm{his}(a, b)$; when the offset is 1, they no longer make a contribution. In between, the contribution of each moving-out grid is $1 - t$.

Similarly, there might be another group of grids X_2 (let A_2 be the total number), which were not originally contributing to $\mathrm{his}(a, b)$, that start moving in to contribute to $\mathrm{his}(a, b)$ as the translation increases from 0 to 1. Their individual contribution to $\mathrm{his}(a, b)$ is $'t'$ at translation t.

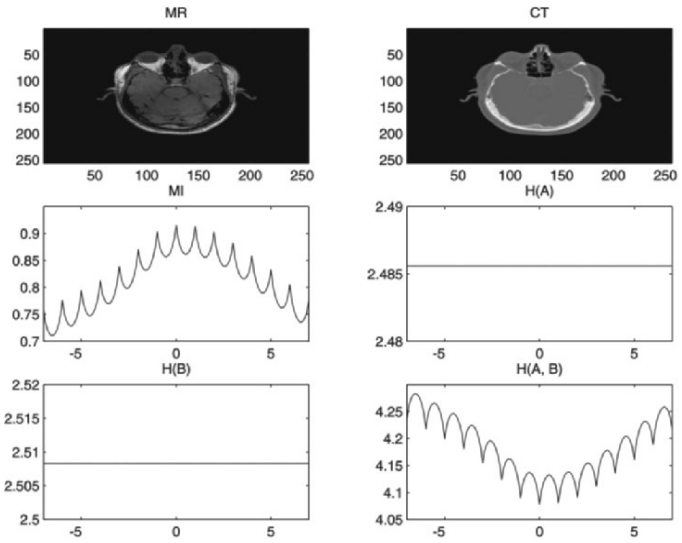


Figure 4. The mutual information value of a pair of multimodal images. Row 1 contains a pair of MR/CT images. (Available online at http://www.isi.uu.nl/Research/Registration/). Rows 2 and 3 show the mutual information (MI),marginal entropies ($H(A)$ and $H(T(B))$) and joint entropy ($(HA, T(B))$) values as functions of translations t (up to ± 7 pixels).

Overall, the combined effects of the moving-in and moving-out sets lead to a change in $\mathrm{his}(a, b)$. So we have

$$A_2 - A_1 = M_2 - M1. \tag{10}$$

At translation t,

$$\begin{array}{rcl} \mathrm{his}(a, b, t) & = & M_1 + A_2\, t - A_1\, t \\ & = & M_1 + (M2 - M1)\, t \\ & = & t\, M_2 + (1 - t)\, M_1. \end{array} \tag{11}$$

So basically within interval [0,1], bin value $\mathrm{his}(a, b, t)$ is a linear function of offset variable t, denoted here by $f(t)$:

$$\begin{array}{l} f(0) = M1, \;\; f(1) = M2, \\ f(t) = t\, f(1) + (1 - t)\, f(0). \end{array} \tag{12}$$

Since we use a histogram to approximate distribution, the joint entropy of two images can be rewritten as $H(A, T(B)) = -\sum_{a,b} \mathrm{his}(a, b, t) \log\, \mathrm{his}(a, b, t)$. As we know, function $x \log\, x$, denoted by $g(x)$ here, is a convex function within

interval (0,1] (note that its second derivative is positive), i.e.,

$$g(t\ x_1 + (1-t)\ x_2) \leq t\ g(x_1) + (1-t)\ g(x_2).$$

Therefore, the individual contribution of bin $\mathrm{his}(a,b,t)$ to joint entropy $H(A,T(B))$ follows:

$$\begin{aligned} g(f(t)) &= g((1-t)\ f(0) + t\ f(1)), \\ &\leq (1-t)\ g(f(0)) + t\ g(f(1)). \end{aligned} \tag{13}$$

The above inequality indicates that each component of $H(A,T(B))$ is a convex function within [0,1]. Since the summation of convex functions is still a convex function, $H(A,T(B)) = -\sum\sum g(\mathrm{his}(a,b,t))$, as a negative combination of certain number of convex functions, is a concave function in [0,1]. Correspondingly, the MI responses are a convex function in the same interval. This property can be easily extended to any intervals $[n, n+1]$ where n is an integer. That is the reason why the responses of $H(A,T(B))$ as a function of translation t (Figure 4) bear a concave-shaped artifact within each integer interval.

If we take a closer look at the above analysis, we can find that the heart of the artifact generation mechanism lies in the following facts:

1. When translation is the only motion type, bilinear interpolation causes the bin values to change in a linear fashion with respect to displacement. In other words, due to the linear kernel used in PV interpolation, all the moving-in and moving-out grids contribute to the changes in the bin values at a synchronized pace.

2. $x \log x$ is a convex function.

As a consequence, a general guideline to reduce the artifact effects can be to "break" the synchronization.

In addition, the following prediction can be made based on the above analysis:

- The artifact effect for pure rotations would be less severe than that of pure translations. This is because the moving-in and moving-out grids, under the pure rotation motion scenario, do not contribute to the change in the histogram at uniform rate. Figure 5a shows the $H(A,T(B))$ values as a function of rotations (up to $\pm 15^\circ$). As is evident, the responses for rotations are much smoother than the translation counterpart.

In the past years, several remedies for reducing artifact effects have been proposed [12, 13] that either rely on resampling one of the input images into different grid sizes, or applying higher-order functions as the interpolation kernel. Although a number of impressive results have been reported, we believe the analysis given in

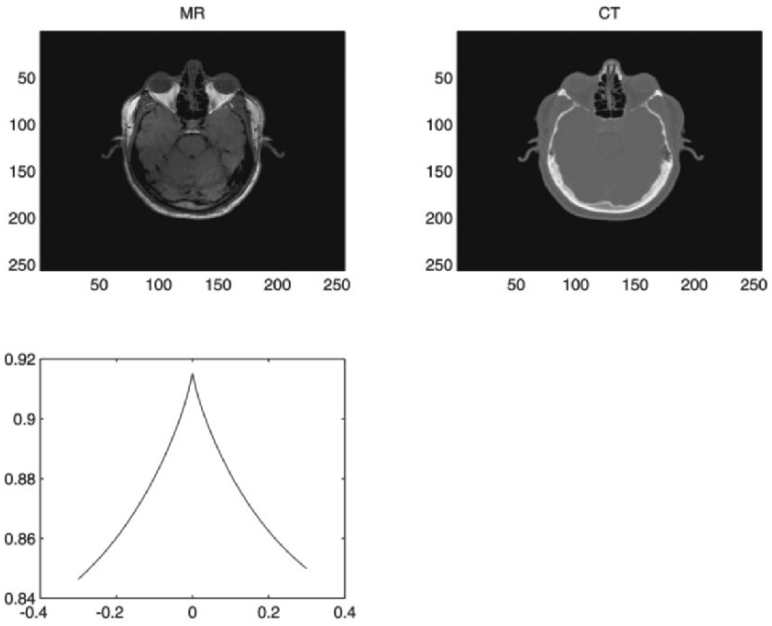


Figure 5. Row 1 contains a pair of MR/CT images. From the homepage of the Image Science Institute, Department of Medical Imaging. Available online at http://www.isi.uu.nl/Research/Registration/registration-frame.html) Row 2 shows the mutual information (MI) value as a function to rotations ($\pm 20^\circ$).

previous sections can provide a deeper insight into which kernel should be chosen and how it will work.

When a bilinear function is used as the PV interpolation kernel, all the relevant grids contribute to the change in a histogram bin value at a synchronized pace. As we mentioned earlier, "to break the synchronization" is the key to reducing/removing interpolation artifacts. For a new interpolation kernel to be chosen to avoid this translation-caused synchronization, two desired properties have to be satisfied:

1. The filter support width should be greater than 2. As being exemplified in bilinear interpolation, if the support comes only from the four surrounding grid points (filter width of 2), at the time that translation is the only motion form, regardless of what interpolation kernel is being used, the changes in bin values will always take place in a synchronized fashion.

2. A uniform (linear) function should not be used as the kernel. If a bilinear function is kept as the interpolation kernel, even with a broader support,

e.g., 16 points (filter width = 4) are used to update the histogram bin values, and due to the consistency of image intensities, synchronization is still likely to occur.

In a nutshell, broad-support nonuniform interpolation kernels should be chosen for the purpose of artifact reduction. We now verify this thought by utilizing two nonuniform functions together with a uniform function, and compare their performance. In the following experiments, we consider three interpolation kernels: the cubic B-spline approximation [16], a Gaussian function, and superlinear interpolation.

The cubic B-spline approximation used in this chapter has a filter support width of 4, and the function kernel is given as follows:

$$h_{\text{cubic}}(x) = \begin{cases} (1/2)|x|^3 & -|x|^2 & & +2/3, & 0 \le |x| < 1, \\ -(1/6)|x|^3 & +|x|^2 & -2|x| & +4/3, & 1 \le |x| \le 2, \\ & & & 0, & \text{elsewhere.} \end{cases} \tag{14}$$

The Gaussian function used here also has a support of 4, and its standard deviation σ is set at 1. The kernel function is given as

$$h_{\text{Gaussian}}(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{x^2}{2\sigma^2}). \tag{15}$$

The superlinear function is an interpolator built only for the purpose of verifying the second claim: *a uniform function should not be used as the kernel*. This filter has a support width of 4, which is broader than the bilinear kernel; however, it is still a uniform function:

$$h_{\text{superlinear}}(x) = \begin{cases} 1/2 - |x|/4, & 0 \le |x| \le 2, \\ 0, & \text{elsewhere.} \end{cases} \tag{16}$$

Figure 6 shows the above three interpolation kernels. Their corresponding MI responses with respect to displacements along the x-axis are given in Figure 7. The left-top subfigure of Figure 7 is the artifact-filled MI response from the linear PV interpolation. The MI response using a cubic B-spline is given in the right-top subfigure. Due to the nonuniformity of the cubic B-spline function, for each transformed pixel its surrounding 4 points contribute to the histogram bin values at a different pace from that of the outside 12 points. As a consequence, synchronization is broken and the artifact effect is reduced. As is evident, the MI response curve for the cubic B-spline kernel is quite smooth. The same arguments can be applied to the Gaussian kernel function. As shown in the left-bottom subfigure, artifact patterns almost disappear. One should note that the MI optima for both cubic the B-spline and Gaussian, as evident in the figure, have not been visibly

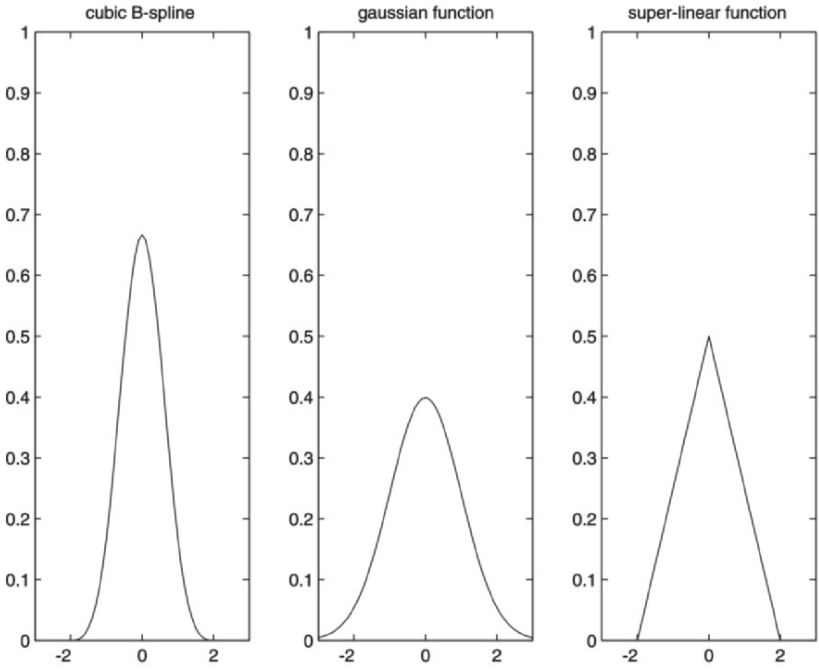


Figure 6. Three interpolation kernels. Left: cubic B-spline function. Middle: Gaussian function. Right: superlinear function.

dislocated, which implies that the ultimate registration estimation can still be very accurate even though both the cubic B-spline and Gaussian are "blurring" filters. The right-bottom subfigure shows the registration response from the superlinear interpolator. Compared with the narrower-support bilinear function, the superlinear interpolator brings more points to smooth up the interpolation procedure; thus, the artifact pattern becomes less severe. However, as the superlinear interpolator is still a uniform kernel, the resulting response pattern is far from artifact-free.

2.2. Experiment Results

In this section we demonstrate the robustness of the new interpolation kernels proposed in the previous section. All the examples contain synthesized misalignments applied to a pair of aligned CT/MR images. This pair of CT/MR images was obtained from the homepage of the Image Science Institute at the University of Medical Center Utrecht (available online at http://www.isi.uu.nl/Research/Registration/registration-frame.html).

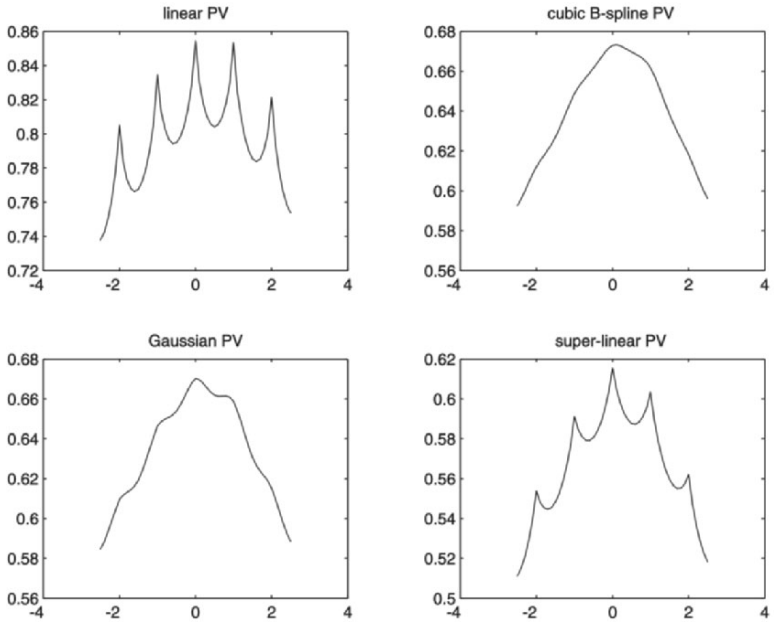


Figure 7. Mutual information responses with respect to translations along the x-axis. Left-top: values for bilinear interpolator. Right-top: for cubic B-spline function. Left-bottom: for Gaussian function. Right-bottom: for superlinear function.

The experiments were designed as follows: with a 2D CT slice as the reference image, the floating image was obtained by applying a rigid transformation to a previously aligned 2D MR image. With 15 randomly generated rigid transformations, we applied three different functions—bilinear, cubic B-spline, and Gaussian—as the interpolation kernels to estimate the motion parameters. These transformations were normally distributed around the values of (10°, 10 pixel, 10 pixel), with standard deviations of (3°, 3 pixel, 3 pixel) for rotation and translation in x and y, respectively.

Table 1 depicts the mean and standard deviation of the estimation errors obtained from the three different interpolation kernels. In each cell the left-most value is the rotation angle (in degrees), while the two right values show the translations in the x and y directions, respectively. Out of the 15 trials, the linear PV method failed 5 times, while the cubic B-spline and Gaussian interpolations all succeeded ("failed" here means that the results had unacceptably large errors). If we only count the cases that yielded reasonable results, as shown in the first (for cubic B-spline), second (for Gaussian function), and third (for linear PV) rows, our approach and the traditional MI have comparable performances, all being very

Table 1. Comparison of Estimation Errors for Rigid Motion between the cubic B-spline, Gaussian, and Bilinear Functions as the Interpolation Kernels

	Mean			Standard deviation		
Cubic B-spline	0.149°	-0.293	0.148	0.049°	0.091	0.198
Gaussian	0.056°	0.250	0.128	0.032°	0.201	0.218
Bilinear	0.087°	0.293	0.383	0.031°	0.121	0.129

accurate. Note that Powell's method was used as the optimization scheme in these experiments.

Figure 8 depicts an example of the registration results. Figure 8a is the reference CT image. Figures 8b,c show the floating MR images, before registration and after registration, respectively. An edge map of the reference CT image is superimposed on the transformed floating image. As is evident, the registration is visually quite accurate.

3. SEGMENTATION-GUIDED DEFORMABLE IMAGE REGISTRATION FRAMEWORKS [14]

In this section we address the second and third challenges pointed out in Section 1.2: *how to integrate segmentation and registration into a unified procedure so that the prior information embedded in both processes can be better utilized.*

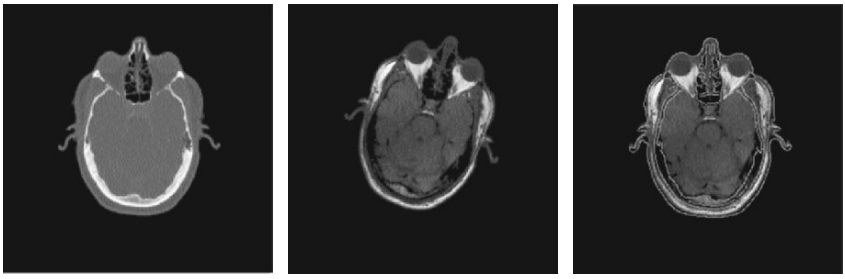


Figure 8. Example of the registration results. The left-most is the reference CT image; the middle is the floating MR image prior to registration, and the right-most is the MRI after registration. Images were obtained from the homepage of the Image Science Institute at the University of Medical Center Utrecht, and are available online at http://www.isi.uu.nl/Research/Registration/registration-frame.html).

Registration and segmentation are the two most fundamental problems in the field of medical image analysis. Traditionally, they were treated as separate problems, each having numerous solutions proposed in the literature. In recent years, the notion of integrating segmentation and registration into a unified procedure has gained great popularity, partially due to the fact that the more practical problems, e.g., atlas-based segmentation, subsume both segmentation and registration components.

Yezzi et al. [17] pointed out the interdependence existing in many segmentation and registration solutions, and a novel geometric, variational framework was then proposed that minimizes an overall energy functional involving both pre- and post-image regions and registration parameters. Geometrical parameters and contour positions were simultaneously updated in each iteration, and segmentations were obtained from the final contour and its transformed counterpart. While this model and its variants [18, 19] are enlightening and pave a promising way toward unifying registration and segmentation, their applicability range is either limited to relatively simple deformation types (rigid/affine) [17], or to relatively simple input images [18, 19].

Vemuri et al. [20] proposed a segmentation + registration model to solve the atlas-based image segmentation problem where the target image is segmented through registration of the atlas to the target. A novel variational formulation was presented that place segmentation and registration processes under a unified variational framework. Optimization is achieved by solving a coupled set of nonlinear PDEs.

Another segmentation + registration model proposed by Noble et al. [21] seeks the best possible segmentation and registration from the *maximum a posteriori* point of view. Improvements in accuracy and robustness for both registration and segmentation have been shown, and potential applications identified. This model is primarily designed for combining segmentation and *rigid* registration. While a non-rigid algorithm was also implemented, the motion field estimation was based on block-matching of size (7×7), which is not dense enough for most non-rigid registration applications.

3.1. Proposed Registration Method

Inspired by the above-mentioned approaches, the work presented in this section is aimed at establishing a segmentation-assisted framework to boost the robustness of non-rigid image registration. Segmentation information is integrated into the process of registration, leading to a more stable and noise-tolerant shape evolution, while a diffusion model is used to infer the volumetric deformation across the image.

Our approach differs from other models in that we use a unified segmentation + registration energy minimization formulation, and optimization is carried out under the level set framework. **A salient feature of our model is its robustness**

against input image noise. We present several 2D examples on synthetic and real data in the implementation results section.

3.2. Segmentation Guided Registration Model

Commonly, the basic input data for a registration process are two images: one is defined as the *fixed* (or *target*) image, $I_1(X)$, and the other as the *moving* (or *source*) image, $I_2(X)$. In addition to these two image, our model requires segmentation of the fixed image, indicating a study area of $I_1(X)$ as another input component.

Let C be the boundary curve of the segmentation. We denote by C_{in} and C_{out}, representing the inside and outside areas of curve C. Let C_1 and C_2 be the average values for C_{in} and C_{out}, respectively.

Contour C can be either input by the user or derived from a training set. We assume that the region captured by C contains a single object of the fixed image; therefore, the intensity values of both the inside and outside of the region should be relatively homogenous. Suppose the fixed and moving images are well corresponded; then, at the time a perfect alignment is achieved, the intensities in the warped moving image should also be relatively uniform within both C_{in} and C_{out}. This observation provides the justification for our model, which is designed based on the following considerations:

> In addition to the set of forces generated by the intensity similarity measure (e.g., SSD) to warp the moving image toward the target, another set of forces, derived from the region homogeneity constraint, should be utilized to pull the moving image toward the correct alignment. This set of forces can provide an extra guideline for the registration process to avoid local energy optima, which is especially helpful when input images are noisy.

Our solution to the registration problem is to minimize the following energy:

$$\begin{aligned} E(V) &= \int_{\Omega} [I_1(X) - I_2(X + V(X))]^2 dX \\ &\quad + \lambda_1 \int_{C_{\text{in}}} [I_2(X + V(X)) - C_1]^2 dX \\ &\quad + \lambda_2 \int_{C_{\text{out}}} [I_2(X + V(X)) - C_2]^2 dX \\ &\quad + \lambda_3 \int_{\Omega} ||\nabla V(x)||^2 dX, \end{aligned} \tag{17}$$

where Ω is the image domain and $V(X)$ denotes the deformation field. λ_1, λ_2, and λ_3 are three constant parameters that weight the importance of each term in

the optimization energy. The $\int[I_1(X) - I_2(X + V(X))]^2 dX$ term provides the main force for matching two images, while terms $\int[I_2(X+V(X))-C_1]^2 dX$ and $\int[I_2(X + V(X)) - C_2]^2 dX$ allow the prior segmentation to exert its influence, aiming to enforce the homogeneity constraints. $\int ||\nabla V(X)||^2 dX$ is a diffusion term to smooth the deformation field.

3.2.1. Level Set Formulation of the Model

Functional (17) can be minimized under the level set framework. Introduce a continuous function $\phi : \Omega \to R$, so $C = \{(X) \in \Omega : \phi(X) = 0\}$, and we choose ϕ to be positive in C_{in} and negative in C_{out}. For the level set formulation, we adopt the model presented in Chan et al. [22]. The new energy, still denoted by $E(V)$ is changed to

$$\begin{aligned} E(V) &= \int_\Omega [I_1(X) - I_2(X + V(X))]^2 dX \\ &\quad +\lambda_1 \int_{\phi\geq 0} [I_2(X + V(X)) - C_1]^2 dX \\ &\quad +\lambda_2 \int_{\phi<0} [I_2(X + V(X)) - C_2]^2 dX \\ &\quad +\lambda_3 \int_\Omega ||\nabla V(X)||^2 dX. \end{aligned} \tag{18}$$

Using the Heaviside function H, defined by

$$H(x) = \begin{cases} 1 & \text{if } x \geq 0, \\ 0 & \text{if } x < 0, \end{cases}$$

and the one-dimensional Dirac measure δ function, defined by

$$\delta(x) = \frac{d}{dx} H(x),$$

the second and third terms can be rewritten as

$$\begin{aligned} E(V) &= \int_\Omega [I_1(X) - I_2(X + V(X))]^2 dX \\ &\quad +\lambda_1 \int_\Omega [I_2(X + V(X)) - C_1]^2 H(\phi(X)) dX \\ &\quad +\lambda_2 \int_\Omega [I_2(X + V(X)) - C_2]^2 (1 - H(\phi(X))) dX \\ &\quad +\lambda_3 \int_\Omega ||\nabla V(X)||^2 dX. \end{aligned} \tag{19}$$

The Euler-Lagrange equation of the functional (19) is given by

$$\begin{aligned}\frac{\partial E}{\partial V} &= 2(I_1(X) - I_2(X+V))(-\nabla I_2(X+V)) \\ &+2\lambda_1(I_2(X+V) - C_1)\nabla I_2(X+V)\cdot H(\phi(X)) \\ &+\lambda_1(I_2(X+V) - C_1)^2 H'(\phi(X))\cdot\nabla\phi(X) \\ &+2\lambda_2(I_2(X+V) - C_2)\nabla I_2(X+V)\cdot(1 - H(\phi(X))) \\ &+\lambda_2(I_2(X+V) - C_2)^2(-H'(\phi(X)).\nabla\phi(X)) \\ &+\lambda_3\nabla^2 V,\end{aligned} \tag{20}$$

where

$$C_1 = \frac{\int_\Omega I_2(X+V)H(\phi(X+V)dX}{\int_\Omega H(\phi(X+V))dxdy}, \tag{21}$$

$$C_2 = \frac{\int_\Omega I_2(X+V)(1 - H(\phi(X+V))dX}{\int_\Omega (1 - H(\phi(X+V)))dxdy}. \tag{22}$$

The level set function being used here is $\phi(X,0) = D(X)$, where $D(X)$ is the signed distance from each grid point to zero level set C. This procedure is standard, and we refer the reader to [23] for details.

3.3. Experimental Results

In this section we demonstrate the improvement made by our algorithm, with synthetic as well as real data sets. In both cases we compare the results using our model with that obtained with the popular Demons algorithm. We apply both schemes to the same data sets.

The synthetic data example contains a pair of synthetically generated images, where the fixed image was generated from the movement of a known non-rigid field. Zero-mean Gaussian noise was then added to each image. The standard deviation was 20. Figures 9a–b show the two images. In the following examples, we chose constants $\lambda_1 = 0.1$, $\lambda_2 = 0.1$, and $\lambda_3 = 1$, respectively.

Segmentation of the fixed image was manually obtained, as superposed on the moving image in Figure 9c. Two registration approaches were then applied: the *Demons algorithm* as well as our *segmentation-guided registration* model. The Demons algorithm used here relies purely on intensity for registration.

The results are shown in the second row of Figure 9. Figure 9c is the transformed moving image from the Demons algorithm, after the registration was completed. Figure 9d shows the result with our model. As is evident, the Demons algorithm had trouble in warping the moving image to a perfect match, which is partially due to the numerous local energy minima resulting from the huge amount

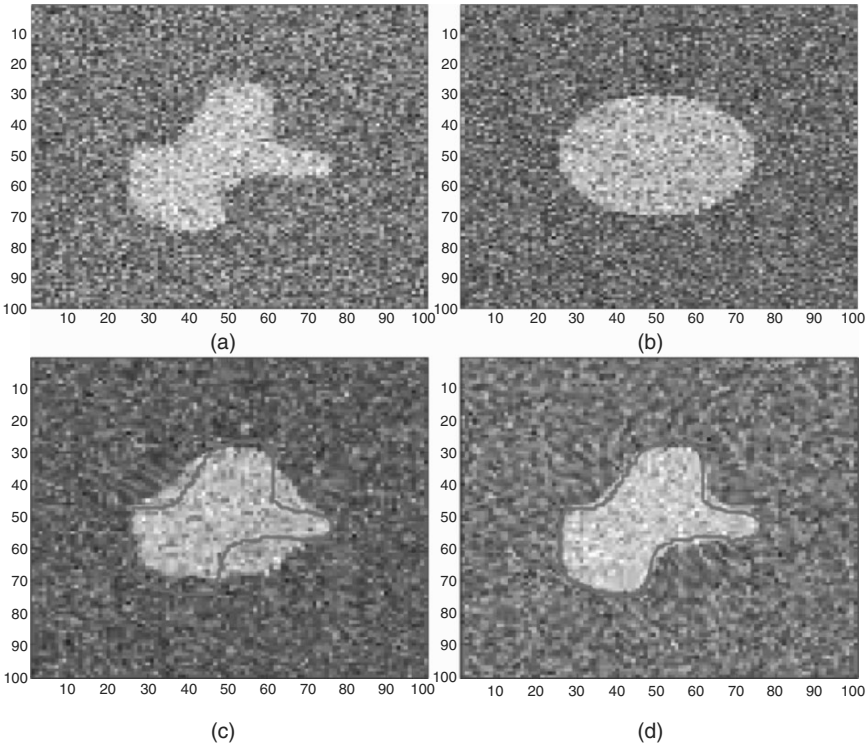


Figure 9. Registration results for image set 1. First row: (a) the fixed image, (b) the moving image. Second row: registration result of (c) using the Demons algorithm, and (d) using our segmentation-guided registration model. The edge map from the fixed image is superposed. See attached CD for color version.

of noise existing in the images. However, the registration result generated from our model is quite accurate, which indicates that the integrated segmentation information is very helpful in pulling the moving image toward a correct matching.

We designed and carried out a similar experiment on a pair of MRI brain slices. The images were obtained from the Davis-Mills Magnetic Resonance Imaging and Spectroscopy Center (MRISC) at the University of Kentucky. The two slices have substantial disparity in the shape of the ventricles, which is the region of interest.

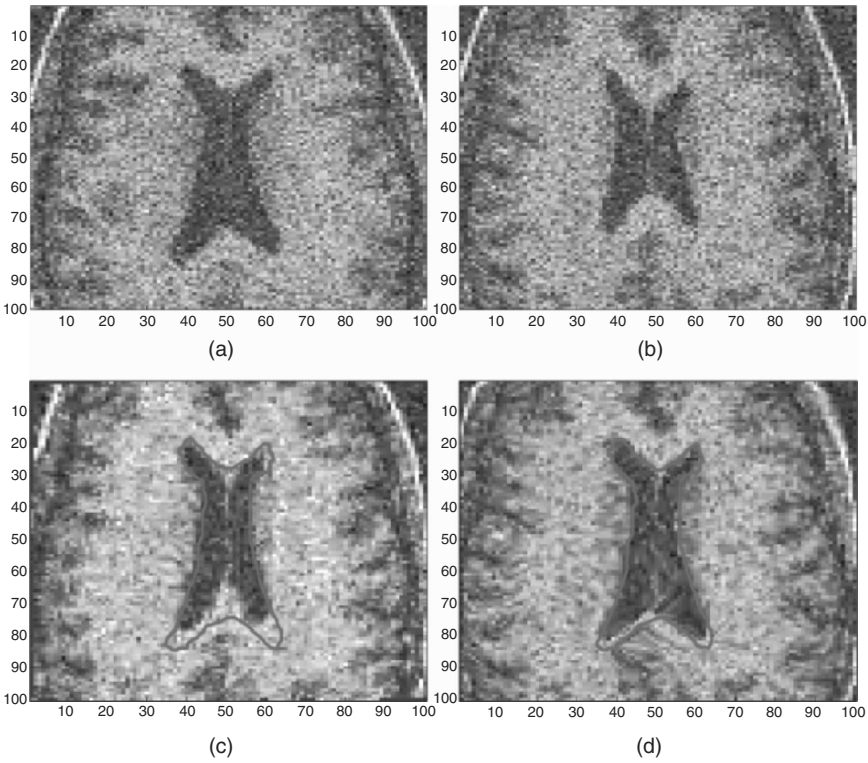


Figure 10. Registration results for image set 2. The images were obtained from the Davis-Mills Magnetic Resonance Imaging and Spectroscopy Center (MRISC) at the University of Kentucky. First row: (a) fixed image, (b) moving image. Second row: registration result of (c) using the Demons algorithm, and (d) using our segmentation-guided registration model. See attached CD for color version.

Figure 10 shows the images and results. Figures 10a,b show the fixed and moving images, respectively. Figures 10c,d depict the results with the Demons algorithm (10c) and our segmentation-guided registration model (10d). As can seen, the former model fails to transform the ventricle area into the desired position, while the latter accurately achieves the registration goal.

4. DISCUSSION AND CONCLUSIONS

As we mentioned in the introduction, rigid registration for mono-modality images is fundamentally a solved problem, which is not the case for multimodality images. Methods based on statistical similarity measures, such as *Mutual Information, Joint Intensity Distributions*, and *Correlation Ratio*, are currently the most popular solutions, but all of them suffer from the interpolation artifact problem.

In Section 2 we quantitatively analyzed the generation mechanism of interpolation artifacts [15]. We concluded that the combination of a linear interpolation kernel and a translation-only motion type leads to generation of the artifact pattern. As a remedy we proposed to use nonuniform interpolation functions in estimating the joint histogram. The cubic B-spline and Gaussian interpolators were compared and we demonstrated improvements via experiments on misalignments between CT/MR brain scans.

Though successful in analyzing *PV interpolation*, we have to point out that *linear interpolation* is a more widely used interpolator than *PV* in many image processing tasks, such as segmentation and transformation. MI artifacts also exist for *linear interpolation*. Several studies [5, 11] have shown that the artifact patterns generated from linear interpolation is quite in contrast to those with PV: linear interpolation tends to generate concave registration functions within integer intervals. An intuitive explanation is that *linear interpolation* blurs the intensity values, which likely reduces dispersion of the image histogram.

All in all, the artifacts generated by these two major interpolation schemes reflect a very general phenomenon: in image registration, when the algorithm involves combination of a specific interpolation method and a nonlinear measure based on the joint histogram of the image, serious artifacts in the registration function will often resulted. Further research on the following topics are worth undertaking:

- What would be the best (in terms of accuracy) interpolation scheme for Mutual-Information-based image registration?
- What are the artifact patterns for other popular similarity measures, such as the *Correlation Ratio (CR)* and *Local Correlation (LC)*.
- Is there a general and systematic solution to this entire interpolation artifact problem?

Comparing recovery of non-rigid deformation from two or multiple images with rigid registration is a far more complicated problem, from both the theoretical and computational points of view. Unifying/combining registration with segmentation provides a promising direction that should lead to a better solution of this problem.

In Section 3.2 we presented a segmentation-guided non-rigid registration framework [14] that integrates the available prior shape information as an extra force to lead to a noise-tolerant registration procedure. Our model differs from other methods in that we use a unified segmentation + registration energy minimization formulation, and optimization is carried out under the level set framework. We showed the improvement accomplished with our model by comparing the results with that of the Demons algorithm. To explore other similarity metrics under the same framework to handle more complicated inputs will be the focus of future work.

Compared to the registration algorithms that rely purely upon intensity correspondence, our segmentation-guided registration approach has the particular advantage of being robust. However, limitations of our technique also exist, one of which is that we assume that *the segmentation of the reference image* is available before registration is carried out. In many atlas-based applications, certain prior segmentation/shape data are available, but it is not necessary that one generate exactly from the reference image. Such prior shapes are more likely obtained from certain previous population analyses, bearing more statistical commonality than precise details. A novel approach to interpreting commonality can be found in [24]. These investigators modeled the statistical shape information of a training set as a sparse group of critical points, each of which can provide an individual force to lead to a smoother and more consistent registration process. A number of experiments performed on both synthetic and real images demonstrated the beneficial effects of the statistical boundary information in improving registration robustness. However, it is foreseeable that these methods will produce less smooth results for noisy inputs if the boundary points are not sampled densely enough. All in all, how to better interpret and integrate available statistical information, and, more ambitious, how to unify segmentation and registration into a single procedure are challenging research directions well worth exploring.

5. REFERENCES

1. Maintz JBA, van den Elsen PA, Viergever MA. 1996. Comparison of edge-based and ridge-based registration of CT and MR brain images. *Med Image Anal* **1**(2):151–161.
2. Haacke E, Liang Z-P. 2000. Challenges of imaging structure and function with MRI. *IEEE Eng Med Biology Magn* **19**(5):55–62.
3. Barron JL, Fleet DJ, Beauchemin SS. 1994. Performance of optical flow techniques. *Int J Comput Vision* **1**(12):43–77.
4. Cover TM, Thomas JA 1991. *Elements of information theory*. New York: John Wiley and Sons.
5. Collignon A, Maes F, Delaere D, Vandermeulen D, Suetens P, Marchal G. 1995. Automated multimodality image registration using information theory. In *Proceedings of the 14th international conference on information processing in medical imaging (IPMI)*, pp. 263–274. Ed YJC Bizais. Washington, DC: IEEE Computer Society.
6. Viola PA, Wells WM. 1995. Alignment by maximization of mutual information. In *Proceedings of the fifth international conference on computer vision (ICCV'95)*, pp. 16–23. Washington, DC: IEEE Computer Society.

7. Wells WM, Viola P, Atsumi H. 1997. Multi-modal volume registration by maximization of mutual information. *Med Image Anal* **1**(1):35–51.
8. Studholme C, Hill D, Hawkes D. 1999. An overlap invariant entropy measure of 3D medical image alignment. *Pattern Recognit* **32**71–86.
9. Ji X, Pan H, Liang ZP. 1999. A region-based mutual information method for image registration. In *Proceedings of the 7th international meeting of the international society for magnetic resonance in medicine*, Vol. 3, p. 2193. Washington, DC: IEEE Computer Society.
10. Maintz JB, Viergever MA. 1998. Survey of medical image registration. *Med Image Anal* **2**:1-36.
11. Pluim J, Maintz J, Viergever M. 2000. Interpolation artefacts in mutual information-based image registration. *Comput Vision Image Understand* **77**:211–232.
12. Taso J. 2003. Interpolation artifacts in multimodality image registration based on maximization of mutual information. *IEEE Trans Med Imaging* **22**(7):845–864.
13. Chen H, Varshney PK. 2003. Mutual information-based CT-MR brain image registration using generalized partial volume joint histogram estimation. *IEEE Trans Med Imaging* **22**(9):1111–1119.
14. Liu J, Wang Y, Liu J. 2006. A unified framework for segmentation-assisted image registration. In *Proceedings of the 7th Asian conference on computer vision. Lecture notes in computer science*, Vol. 3852, pp. 405–414. Berlin: Springer.
15. Liu J, Wei M, Liu J. 2004. Artifact reduction in mutual-information-based CT-MR image registration. In *Proceedings of the SPIE*, Vol. 5370, pp. 1176-1186. *Medical imaging 2004: physiology, function, and structure from medical images*. Ed AA Amini, A Manduca. Bellingham, WA: International Society for Optical Engineering.
16. Lehmann TM, Gonner C, Spitzer K. 1999. Survey: interpolation methods in medical image processing. *IEEE Trans Med Imaging* **18**(11):1049–1075.
17. Yezzi A, Zollei L, Kapur T. 2003. A variational framework for joint segmentation and registration. *Med Image Anal* **7**(2):171–185.
18. Unal G, Slabaugh G, Yezzi A, Tyan J. 2004. *Joint segmentation and non-rigid registration without shape priors*. Siemens Technical Report SCR-04-TR-7495.
19. Unal G, Slabaugh G. 2005. Coupled PDEs for non-rigid registration and segmentation. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Vol. 1, pp. 168–175. Washington, DC: IEEE Computer Society.
20. Vemuri B, Chen Y, Wang Z. 2002. Registration-assisted image smoothing and segmentation. *Proceedings of the 7th European Conference on Computer Vision, Part IV. Lecture notes in computer science*, Vol. 2353, pp. 546–559.
21. Wyatt PP, Noble JA. 2003. MAP/MRF joint segmentation and registration of medical images. *Med Image Anal* **7**:539–552.
22. Chan T, Vese L. 2001. An active contour model without edges. *IEEE Trans Image Process* **10**(2):266–277.
23. Sussman M, Smereka P, Osher S. 1994. A level set approach for computing solutions to incompressible two-phase flow. *J Comput Phys* **114**:146–159.
24. Wang Y, Staib LH. 2000. Physical model-based non-rigid registration incorporating statistical shape information. *Med Image Anal* **4**(1):7–20.
25. Simulated brain database. Available at http://www.bic.mni.mcgill.ca/brainweb/.
26. Thirion JP. 1998. Image matching as a diffusion process: an analogy with Maxwell's demons. *Med Image* **Anal***2*(3):243–260.
27. Fischer B, Modersitzki J. Image registration using variational methods. Paper presented at Visual Analysis, Image-Based Computing and Applications Workshop. Available at http://64.233.161.104/search?q=cache:EyP6NVf6DhsJ:www.icm.edu.pl/visual/moderzitski.ps+
28. Leventon M, Grimson WEL. 1999. Multi-modal volume registration using joint intensity distributions. In *Lecture notes in computer science*, Vol. 1496, pp. 1057–1066. Berlin: Springer. Available at http://www.ai.mit.edu/people/leventon/Research/9810-MICCAI-Reg/.

29. Likar B, Pernus F. 2001. A hierachical approach to elastic registration based on mutual information. *Image Vision Comput* **19**:33–44.
30. Vemuri BC, Huang S, Sahni S, Leonard CM, Mohr C, Gilmore R, Fitzsimmons J. 1998. An efficient motion estimator with application to medical imaging. *Med Image Anal* **2**(1):79–98.
31. Woods R, Maziotta J, Cherry S. 1993. MRI-PET registration with automated algorithm. *J Comput Assist Tomogr* **17**:536–546.
32. Zhu S, Yuille A. 1996. Region competition, unifying snakes, region growing, and Bayes/MDL for multibank image segmentation. *IEEE Trans Pattern Anal Machine Intell* 18(9):884–900.

INDEX