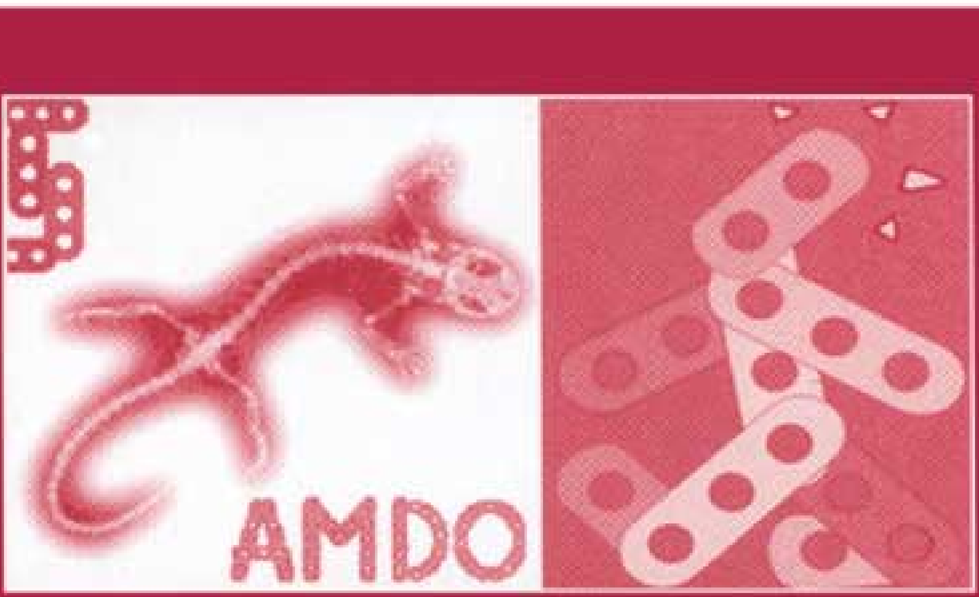
Francisco J. Perales
Bruce A. Draper (Eds.)

LNCS 3179

Articulated Motion and Deformable Objects

Third International Workshop, AMDO 2004
Palma de Mallorca, Spain, September 2004
Proceedings



Springer

This page intentionally left blank

Francisco J. Perales
Bruce A. Draper (Eds.)

Articulated Motion and Deformable Objects

Third International Workshop, AMDO 2004
Palma de Mallorca, Spain, September 22-24, 2004
Proceedings

Springer

eBook ISBN: 3-540-30074-0
Print ISBN: 3-540-22958-2

Preface

The AMDO 2004 workshop took place at the Universitat de les Illes Balears (UIB) on 22–24 September, 2004, institutionally sponsored by the International Association for Pattern Recognition (IAPR), the MCYT (Comision Interministerial de Ciencia y Tecnologia, Spanish Government), the AERFAI (Spanish Association for Pattern Recognition and Image Analysis), the EG (Eurographics Association) and the Mathematics and Computer Science Department of the UIB. Also important commercial sponsors collaborated with practical demonstrations; the main contributors were: Barco Electronics Systems (Title Sponsor), VICOM Tech, ANDROME Iberica, CESA and TAGrv.

The subject of the workshop was ongoing research in articulated motion on a sequence of images and sophisticated models for deformable objects. The goals of these areas are to understand and interpret the motion of complex objects that can be found in sequences of images in the real world. The main topics considered priorities are: deformable models, motion analysis, articulated models and animation, visualization of deformable models, 3D recovery from motion, single or multiple human motion analysis and synthesis, applications of deformable models and motion analysis, face tracking, recovery and recognition models, and virtual and augmented reality systems.

The main objective of this workshop was to relate on fields using computer graphics, computer animation or applications in several disciplines combining synthetic and analytical images. The use of new graphical user interfaces will be very important in the near future; that means that convergence between multidisciplinary areas will be necessary to reach new perceptual or multimodal interfaces, including combination of several perspectives of the topics mentioned above. In this regard it is of particular interest to encourage links between researchers in the areas of computer vision and computer graphics who have common problems and frequently use similar techniques. The workshop included several sessions of orally presented papers and three tutorials. We also had three invited speakers treating various aspects of the main topics. These invited speakers were: Prof. Bruce Draper from Colorado University (USA), Prof. Pere Brunet from Polytechnics Catalonia University (UPC / Spain), and Prof. Richard Bowden from Surrey University (UK).

September 2004 — Francisco J. Perales and Bruce A. Draper

Organization

AMDO 2004 was organized by the Computer Graphics and Vision team of the Department of Mathematics and Computer Science, Universitat de les Illes Balears (UIB) in cooperation with IAPR (International Association for Pattern Recognition), AERFAI (Spanish Association for Pattern Recognition and Image Analysis) and EG (Eurographics Association).

Program Committee

General Workshop Co-chairs	F.J. Perales, Mathematics and Computer Science Department, UIB (Spain) B.A. Draper, Department of Computer Science, Colorado State University (USA)
Organizing Chairs	M.J. Abásolo, A. Amengual, J.M. Buades, M. González, A. Jaume, A. Igelmo, C. Manresa, R. Mas, P.M. Mascaro, P. Palmer, J. Varona, UIB (Spain)
Tutorial Chairs	M. González, J. Varona, F.J. Perales, UIB (Spain)

Referees

Abásolo, M.J.	Universitat Illes Balears, Spain
Aloimonos, Y.	University of Maryland, USA
Aggarwal, J.K.	University of Texas, USA
Ayache, N.	INRIA, Sophia-Antipolis, France
Badler, N.I.	University of Pennsylvania, USA
Bowden, R.	University of Surrey, UK
Brunet, P.	UPC, Spain
Campilho, A.	Univ. of Oporto, Portugal
Carlsson, S.	CVAP, Sweden
Cohen, I.	University of Southern California, USA
Davis, L.S.	University of Maryland, USA
Dugelay, J.L.	EURECOM, France
González, M.	Universitat Illes Balears, Spain
Kakadiaris, I.A.	University of Houston, USA
Kittler, J.	University of Surrey, UK
Hancock, E.R.	University of York, UK
Mas, R.	Universitat Illes Balears, Spain
Medioni, G.	University of Southern California, USA
Pentland, A.	Media Lab, MIT, USA
Pérez de la Blanca, N.	University of Granada, Spain
Pla, F.	University of Jaume I, Spain
Qin, H.	Stony Brook University, New York, USA
Sanfeliu, A.	IRI, CSIC-UPC, Spain
Serón, F.	University of Zaragoza, SPA
Shirai, Y.	University of Osaka, Japan
Skala, V.	University of Plzen, Czech Republic
Susin, A.	UPC, Spain
Terzopoulos, D.	University of Toronto, Canada
Teixeira, J.C.	FCTUC, Portugal
Thalmann, D.	EPFL, Switzerland
Varona, J.	Universitat Illes Balears, Spain
Villanueva, J.	UAB-CVC, Spain

Sponsoring Institutions

IAPR (International Association for Pattern Recognition)
AERFAI (Spanish Association for Pattern Recognition and Image Analysis)
EG (Eurographics Association)
MCyT (Ministerio de Ciencia y Tecnologia, Spanish Government)
Mathematics and Computer Science Department, Universitat de les Illes Balears (UIB)

Commercial Sponsoring Enterprises

Barco Electronics S.A.
VICOM Tech S.A.
ANDROME Iberica S.A.
CESA S.A.
TAGrv S.L.

Table of Contents

This page intentionally left blank

Using a Generalized Linear Mixed Model to Study the Configuration Space of a PCA+LDA Human Face Recognition Algorithm

Geof H. Givens[1], J. Ross Beveridge[2], Bruce A. Draper[2], and David Bolme[2]

[1] Statistics Department, Colorado State University
Fort Collins, CO, 80523 USA
[2] Department of Computer Science, Colorado State University
Fort Collins, CO, 80523 USA

Abstract. A generalized linear mixed model is used to estimate how rank 1 recognition of human faces with a PCA+LDA algorithm is affected by the choice of distance metric, image size, PCA space dimensionality, supplemental training, and the inclusion of test subjects in the training data. Random effects for replicated training sets and for repeated measures on people were included in the model. Results indicate that variation among subjects was a dominant source of variability, and that there was moderate correlation within people. Statistically significant effects and interactions were found for all configuration factors except image size. The most significant interaction is that dhanges to the PCA+LDA configuration only improved recognition for test subjects who were included in the training data. For subjects not included in training, no configuration changes were helpful.
This study is a model for how to evaluate algorithms with large numbers of parameters. For example, by accounting for subject variation as a random effect and explicitly looking for interaction effects, we are able to discern effects that might otherwise have been masked by subject variation and interaction effects. This study is instructive for what it reveals about PCA+LDA.

1 Introduction

Proper configuration or tuning is a critical factor in making effective use of most vision algorithms. Virtually all algorithms have a configuration space, and it is well understood performance varies with different configuration choices. There are a variety of approaches to configuration. Perhaps the most obvious and most common is semi-structured exploration: the developers of the algorithm run the algorithm, examine results, make changes, and repeat until they are satisfied. Somewhat more rigorous is iterative refinement with an objective performance standard as illustrated by the work of Bowyer [1].

We advocate designing experiments, collecting performance data, and applying statistical tools to determine how configuration choices influence performance. We apply this approach here to understand the configuration space of an interesting human face recognition algorithm. The statistical tool is a generalized linear mixed model. The face recognition algorithm uses principal components analysis (PCA) followed by linear discriminant analysis (LDA) and is based closely on the work of Zhao [2].

The use of the generalized linear mixed model in this context provides several very important features lacking from simpler approaches. First, it permits us to model what are called fixed effects that are attributable to various algorithm configuration choices. These include factors such as image size and the selection (or generation) of training data. Second, it also permits us to model random effects that influence performance but are not under our direct control. A good example of a random effect is subject identity.

Accounting for random effects due to subject identity is particularly important in domains such as human face recognition. Some individuals are harder to recognize than others, but we are seldom afforded the luxury of choosing to only recognize the easy ones. Thus, a large source of variation in observed performance is attributable to a recorded variable that is not under our control. In simpler models that do not explicitly account for random effect, variation among subjects can easily obscure other effects.

2 Image Pre-processing

Data pre-processing is arguably as important as any other aspect of an algorithm. In the experiments presented below, the face imagery is pre-processed in essentially the same fashion as in the FERET tests [3, 4]. Specifically, the following pre-processing steps are applied to images:

1. Integer to float conversion.
2. Geometric normalization to align hand-picked eye coordinates.
3. Cropping to an elliptical mask centered on the face.
4. Histogram equalization of non-cropped pixels.
5. Pixel normalization so that every image has a mean of zero and a standard deviation of one.

Optional pre-processing steps considered as part of the configuration space include:

– Adding zero mean Gaussian noise to each pixel.
– Reflecting the image around the vertical axis. This creates a mirror-image of the original face.
– Shifting the image left, right, up or down.
– Reducing the image size by a factor of two.

3 The PCA+LDA Algorithm

The PCA+LDA human face recognition algorithm considered in this study is based upon the work of Zhao [2]. It processes a set of training images and builds a PCA subspace as in the standard Eigenfaces algorithm [5]. The training images are projected into the PCA subspace and grouped according to subject identity. Each subject is treated as a distinct class and the LDA basis vectors are computed. In the experiments that follow, training is performed on 64 randomly selected subjects with 3 images each. Thus, the maximum possible dimensionality of the PCA space is 192, and one of the configuration choices involves how many of these to retain. Since there are 64 subjects, LDA generates 63 Fisher basis vectors. The two linear projections are then composed, yielding a single projection from image space into the PCA+LDA space.

In the testing phase, the PCA+LDA algorithm reads the subspace basis vectors generated during training, projects test imagery into this subspace, and measures distances between image pairs. Two distance measures are considered in the experiments below. The first is the standard L2 norm $(\mathcal{L}_2)$. The second is a heuristic distance measure defined by Zhao [2] called Soft L2 $(\mathcal{L}_{Soft})$. Formally, these two distance measures are defined as:

$$\mathcal{L}_2(A,B) = \sum_{i=1}^{N} (A_i - B_i)^2 \qquad \mathcal{L}_{Soft}(A,B) = \sum_{i=1}^{N} (\lambda_i)^{0.2} (A_i - B_i)^2 \quad (1)$$

where λ_i is the ith eigenvalue from the the solution of the generalized eigenvector problem associated with finding the fisher basis vectors [6].

4 Experimental Design

This study used 1,024 pre-processed images of 256 FERET subjects. The choice of subjects is based upon the data available: there are 256 FERET subjects with at least 4 frontal images. These images are divided into *training, gallery* and *probe* sets. Training images are used to define the PCA+LDA space. Probe images are test images to be identified. Gallery images are the stored images that probe images are compared to. For every subject, one image is selected as the probe image and one of the other three images (selected at random on each trial) is the gallery image. When a subject is used for training, all three non-probe images are training images. Note that whether the sets of gallery and training subjects are disjoint is a configuration parameter.

The response variable is the recognition rank for a given subject and configuration of the algorithm. The PCA+LDA algorithm sorts the gallery images according to their distance from the probe image; recognition rank is the position of the subject's gallery image in this sorted list. Thus a recognition rank of 1 indicates that a nearest neighbor classifier will correctly match the subject to their gallery image. A recognition rank of 2 indicates that one imposter appears more similar to the probe image than the subject's gallery image does.

Any recognition rank larger than 1 suggests an algorithm will do a poor job of recognizing the subject.

One of the most important aspects of algorithm configuration is training. In these experiments, training images were drawn from the 3 non-probe images of each subjects. For each test, the PCA+LDA algorithm was trained on $\frac{1}{4}$ of the subjects. Thus, the algorithm was trained on 192 images, 3 images for each of the 64 randomly selected subjects. An additional 192 images were added to the training in some trials to see if supplementing the training with altered versions of training images improves performance. Three specific sources of supplemental imagery were used (note that in all three cases one supplemental image is generated for each training image):

1. Reflect each image horizontally about its mid-line.
2. Add independent mean zero Gaussian noise (with a standard deviation of 2% of the dynamic range of the image) to the grey-scale value of each pixel in each image.
3. Shift each image left by one pixel.

The entire training procedure using 64 randomly selected subjects was itself replicated 8 times in order to determine how much the exact choice of training subjects matters.

The change in the size of the training set from 192 images without supplemental training to 384 with supplemental training is particularly important if it affects the dimensionality of the PCA space used for discrimination. Therefore, this dimensionality was explicitly controlled by setting it to either 114 eigenvectors (60% of 191 or 30% of 383) or the number of eigenvectors that comprise 90% of the energy, denoted $E_{90\%}$ [7]. Typical values of k using the $E_{90\%}$ rule to truncate the eigenspace ranged from 64 to 107.

To test the effect of gallery image choice on algorithm performance, each probe was tested on 30 different gallery sets. These galleries were generated by randomly selecting 1 of the remaining 3 images per subject. The same 30 randomly generated gallery sets were used throughout the entire experiment.

The tests in this experiment were run using the standard $\mathcal{L}_2$ norm and the $\mathcal{L}_{Soft}$ measure proposed by Zhao [2] and defined above in equation 1. It was also brought to our attention in discussions with Zhao that performance may benefit from reducing the image size prior to recognition. Thus, two different image sizes were tested: the FERET standard size (130 × 150) and smaller images with $\frac{1}{4}$ as many pixels (65 × 75).

One additional variable was observed in this experiment: whether a particular probe had been included in the training set used in each test. Since there were 256 probes and 64 subjects in the training set, only 25% of probes were included in any training set.

The experimental design was a balanced complete factorial; in other words, we ran each choice in every combination. This results in $8 \times 30 \times 4 \times 2 \times 2 \times 2 = 7680$ attempts to recognize each probe image under different circumstances. A summary of the configuration and replication factors used in this experimental design is given in Table 1.

Table 1. Summary of the configuration and replication factors used in the experimental design

Factor	Index	Values
Training Set	t	$0,\ldots,7$
Gallery Set	g	$0,\ldots,29$
Probe Image	p	$0,\ldots,255$
Training Set Supplement	u	None, Shift, Reflection, or Noise
Metric	m	Standard or Soft L2,i.e. $\mathcal{L}_2$ or $\mathcal{L}_{Soft}$
PCA Space Dimension	d	114 or $E_{90\%}$
Image Size	s	Standard (130×150) or Small (65×75)
Training Set Inclusion Flag	f	No or Yes

5 The Model

Let $R_{tpgusmdf}$ represent the recognition rank when the p^{th} probe image of the s^{th} size and f^{th} value of the training set inclusion flag is matched to the g^{th} gallery, using an algorithm with the m^{th} metric limiting the PCA discrimination space in the d^{th} manner, when the algorithm is trained on the t^{th} baseline training set with the u^{th} type of training set supplementation. Then let $Y_{tpusmdf}$ be the number of gallery sets for which $R_{tpgusmdf} = 1$. Note that $0 \leq Y_{tpusmdf} \leq 30$.

We model $Y_{tpusmdf}$ as a binomial random variable, namely $Y_{tpusmdf} \sim Bin(30, p_{tpusmdf})$. The probability of rank 1 recognition (namely $p_{tpusmdf}$) is modeled as:

$$\log \frac{p_{tpusmdf}}{1 - p_{tpusmdf}} = L + U_u + +M_m + D_d + zI_t(p) + \tau_t + \pi_p + (UM)_{um} + (UF)_{uf} + (UD)_{ud} + (MF)_{mf} + (MD)_{md} + (DF)_{df} + (MDF)_{mdf} \qquad (2)$$

where

L	is the grand mean
U_u	is the effect of including the u^{th} supplement to the training set
M_m	is the effect of using the m^{th} distance metric
D_d	is the effect of the d^{th} strategy for limiting PCA dimensionality
F_f	is the effect of the f^{th} option for including a probe in the training set
$(UM)_{um},\ldots$	are 2- and 3-way interactions between main effects
τ_t	is the random effect of the t^{th} set of people used for training, and
π_p	is the random effect of the p^{th} probe image.

Commonly $\log(p/(1-p))$ is termed the logit transformation.

We are interested in the amount of variability in rank 1 recognition rate attributable to the training set, but the 8 training sets used in our experiment are merely a random sample from the variety of sets that might have been used. Therefore, we fit $\tau_t \sim N\left(0, \sigma_\tau^2\right)$ as independent random effects. Using the same rationale, we modeled the π_p as mean zero normal random effects with constant variance (σ_π^2), but with a more complex covariance structure. In particular, results from matching the same probe image in different circumstances are probably more correlated than matching different probe images. We assumed a compound symmetric covariance structure, which allows correlation ρ between any pair of $Y_{tpusmdf}$ corresponding to the same probe image. Results for different probe images were modeled as independent. Thus, the model accounts for the possibilities that some training sets were more effective than others and that some probes were 'easier' than others. It also allows for correlation induced by repeated measurement of the same probes under different experimental conditions.

This model is a generalized linear mixed model [8], or more precisely a mixed effects logistic regression with repeated measures on subjects. Such models can be fit routinely using restricted pseudo-likelihood or the MIVQEU0 method [9, 10, 11, 12].

6 Results

We originally fit a larger model including the main effect for image size and all its two-way interactions. A major finding of our study was that image size did not significantly affect rank 1 recognition rate, either overall or in any interaction. Through a sequence of model fitting steps, we removed all image size terms from the original model (and added the three-way *MDF* interaction) to obtain the final fitted model given in (2).

On the logit scale, the random training sets accounted for virtually no variability in rank 1 recognition rate. Any randomly selected training set of this size is as good as any other. Also on the logit scale, the correlation between probes of the same person under different conditions was about 0.70. Thus our findings strongly support the conclusion that some people are harder to recognize at rank 1 than others, and that this variation accounts for the majority of the total variability in rank 1 recognition rate on the logit scale.

The best configuration (among those tested) of the PCA+LDA algorithm used the $\mathcal{L}_{Soft}$ distance metric with PCA dimensionality set at 60% of the number of eigenvectors and supplementing the training set with shifted images. Estimated probability of rank 1 recognition for this configuration was .87 for subjects included in the training set and .75 for those who were not. In comparison, the estimated probabilities were .77 and .74, respectively, for the 'baseline' PCA+LDA configuration that use the standard $\mathcal{L}_2$ distance metric and no training supplementation.

The main focus of our study is the estimation of the main effects and interactions included in (2). These allow us to quantify how changes to the PCA+LDA configuration affect estimated probability of rank 1 recognition. Table 2 shows tests of the statistical significance of the various fixed effects. Figures 1 and 2 show the various estimated effects, as described in more detail in the rest of this

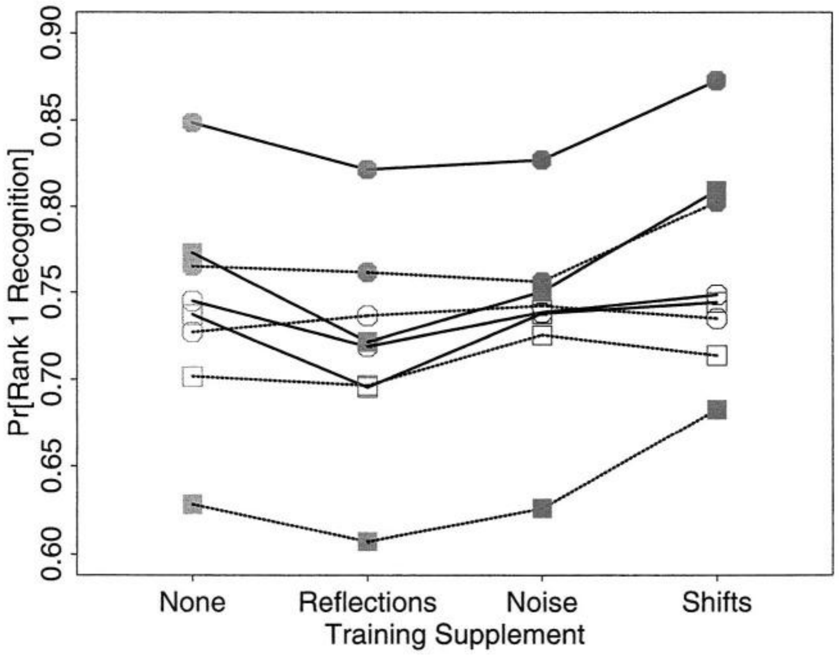


Fig. 1. Effect of training set supplementation on predicted probability of rank 1 recognition. See the text for codings for color, shape, fill, and line type

Table 2. Table of significance of model effects and interactions based on approximate F-tests

Effect	Label	(Num.) d.f.	F value	p-value
Training Supplement	U	3	139.1	<.0001
Distance Metric	M	1	1087.8	<.0001
PCA Dimensionality	D	1	936.6	<.0001
Training Inclusion Flag	F	1	333.8	<.0001
$U \times M$ Interaction	UM	3	14.2	<.0001
$U \times F$ Interaction	UF	3	67.0	<.0001
$U \times D$ Interaction	UD	3	41.5	<.0001
$M \times F$ Interaction	MF	1	717.6	<.0001
$M \times D$ Interaction	MD	1	59.1	<.0001
$D \times F$ Interaction	DF	1	703.7	<.0001
$M \times D \times F$ Interaction	MDF	1	5.04	.0248

section. The threshold for statistical significance in these figures is substantially smaller than the magnitude of difference that would be scientifically important

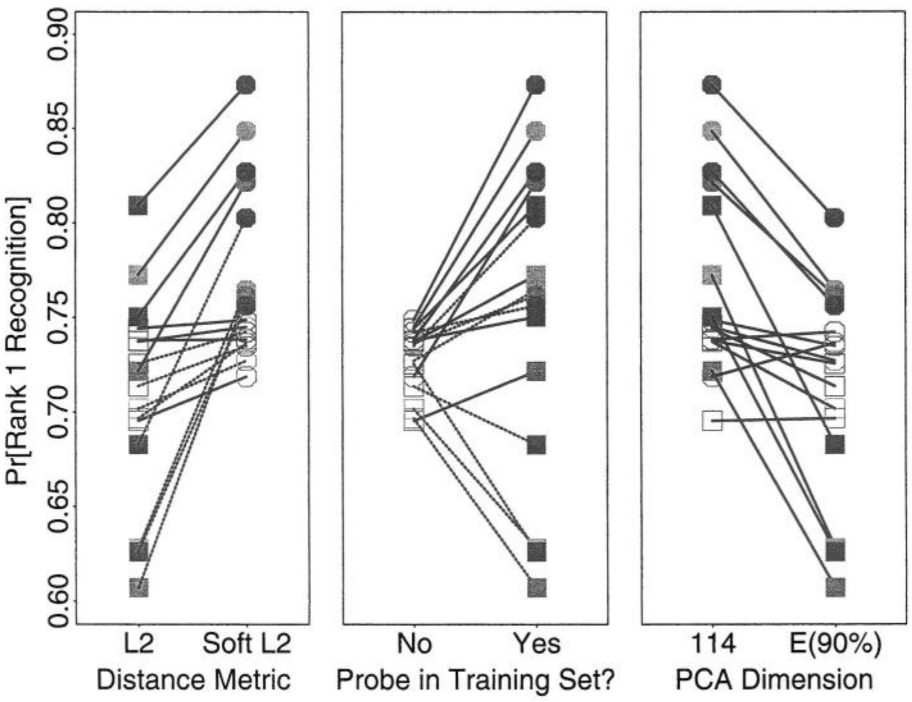


Fig. 2. Effects of distance metric, PCA dimensionality, and inclusion of the probe in the training set on predicted probability of rank 1 recognition. See the text for codings for color, shape, fill, and line type. There is no line type variation in the right panel because PCA dimensionality is labeled on the horizontal axis

in the context of human identification problems. Therefore it suffices to examine these figures for interesting effects with the understanding that any interesting effect would be statistically significant according to a simple F-test. The significance threshold is so small because the large sample size provides a lot of discriminatory power.

Figures 1 and 2 employ a consistent labeling to illustrate the results. Color indicates training supplement (orange for none, green for reflections, red for noise, and blue for shifts). Shape indicates distance metric (square for standard $\mathcal{L}_2$ and circle for $\mathcal{L}_{Soft}$). Fill indicates whether a probe is included in the training set (solid fill for yes and hollow for no). Finally, PCA dimensionality is indicated by line type (solid lines for 114 and dashed for $E_{90\%}$).

Figure 1 shows the effect of training set supplementation on predicted probability of rank 1 recognition. Overall, adding shifted images was best and adding reflected images was actually worse than no supplementation. This refutes a hypothesis we originally held that reflections were effectively equivalent to shifts due to the rough horizontal symmetry of human faces. This figure also illustrates the interaction between the training set inclusion variable and supplementation type: the improvement or interference provided by training set supplementation was substantial only for recognition of probes that were included in the training set.

The left panel of Figure 2 shows the effect of distance metric on predicted probability of rank 1 recognition. Overall, the soft L2, $\mathcal{L}_{Soft}$, metric was superior, but this difference was mostly limited to instances when the probe was included in the training set. When training did not include the probe, very little was gained by switching from $\mathcal{L}_2$ to $\mathcal{L}_{Soft}$. This is the significant interaction.

The effect of including the probe in the training set is further shown in the center panel of Figure 2. Somewhat surprisingly, the overall effect of training set inclusion on predicted probability of rank 1 recognition was only slightly positive. There was an important three-way interaction with distance metric and PCA dimensionality which can be described as follows. The effect of training set inclusion was always positive when the PCA dimensionality was limited to 114. When the PCA dimensionality was set at $E_{90\%}$, the effect of training set inclusion depended on the choice of metric: positive effect for $\mathcal{L}_{Soft}$ and negative for standard $\mathcal{L}_2$. A definitive explanation for why inclusion of the probe in the training set actually impeded recognition when using the $\mathcal{L}_{Soft}$ with $E_{90\%}$ PCA eigenvectors eludes us.

The right panel of Figure 2 shows the effect of PCA dimensionality on predicted probability of rank 1 recognition. Overall, the use of $E_{90\%}$ PCA eigenvectors was inferior to fixing 114 eigenvectors. Since $E_{90\%}$ was virtually always less than 114, it is perhaps not surprising that this configuration was inferior. Again a significant interaction can be seen: the negative effect was mostly confined to probes that were included in the training set.

An important broader conclusion can be distilled from the details above. Variations in distance metric, PCA dimensionality, and training set supplementation exemplify the many sorts of strategies employed by researchers to improve the performance of the basic PCA+LDA algorithm. Such strategies appear to span a conceptual dimension with opposing approaches at each extreme. Direct manipulation of the training lies at one extreme (eg. training supplementation) and changes in discriminatory strategy of the algorithm lies at the other extreme (eg. changing the distance metric). Both ends of the spectrum are motivated by the hope that such changes will improve recognition for probes not used in training. Our results firmly establish that none of these strategies succeed in this goal. On the other hand, our results also show that either end of the strategic spectrum can yield improved results on probes that are included in the training set. Changes to the algorithm provide about twice the improvement as changes to the training.

7 Discussion

Since all the tested variations in the baseline configuration of the PCA+LDA algorithm only improved rank 1 recognition for probes that were included in the training set, our results raise several important questions. First, the earlier work of Zhao indicated the soft-L2 measure would improve performance for subjects not in the training set. The discrepancy between our findings and Zhao's suggests more work may be needed before this matter is fully understood. Second, and more broadly, how might one best configure a PCA+LDA algorithm to enhance performance for such subjects.

Another avenue of future work might consider alternate and more generous definitions of recognition success. For example, do our results hold when a 'success' is defined as ranking the correct gallery image at rank 10 or higher? Although our statistical methodology could be applied equally well to such a problem, the FERRET data are not very suitable for such an analysis. About 47% of the 65,536 attempts at recognition under different configurations and training resulted in a rank 1 match for all 30 galleries; the overall frequency of rank 1 matches was 76%. If the response variable in our analysis were changed to, say, rank 10 recognition, the outcomes would be skewed even more toward successful matches and the reliability of the inference would be decreased, all else being unchanged.

The randomness inherent in our outcome—rank 1 recognition—is generated by the repeated testing of probes under 30 different galleries. In a real application, similar randomness applies to probe images as well. The advantage of the experimental design used here is that the binomial sample size parameter can be set quite large (we chose 30). Consequently, the $Y_{tpusmdf}$ are very informative about the recognition rate parameter, $p_{tpusmdf}$. An alternative experimental design would have been to fix a single gallery and to test different probe images of the same person. In the FERRET data, however, there are only a small number of frontal images for most subjects. Thus, the observed recognition rate for each individual subject is less informative. When we have access to a large data set with many replicates per subject, we plan to run such an analysis varying probe presentation.

8 Conclusion

The results presented here represent one of the largest tuning studies of a significant face recognition algorithm on record. As an exercise in advancing the state of experiment design, it is great success. For example, the existence of strong interactions between whether a subject is in the training set and whether the algorithm does better or worse is expected, but characterizing this dependence in clear quantitative terms is significant. As to what this study tells us about our ability to significantly alter, in particular improve, algorithm performance, the results are somewhat disappointing. Clearly, there is a need for additional study of the PCA+LDA algorithm in order to truly demarcate those factors of algorithm configuration that may, in future, prove to significantly improve performance. At a minimum, such studies should adhere to the standards established here.

Acknowledgements

This work supported by the Defense Advanced Research Projects Agency under contract DABT63-00-1-0007.

References

[1] Heath, M., Sarkar, S., Sanocki, T., Bowyer, K.: A robust visual method for assessing the relative performance of edge detection algorithms. PAMI **19** (1997) 1338–1359

[2] Zhao, W., Chellappa, R., Krishnaswamy, A.: Discriminant analysis of principal components for face recognition. In: In Wechsler, Philips, Bruce, Fogelman-Soulie, and Huang, editors, Face Recognition: From Theory to Applications. (1998) 73–85

[3] Phillips, P., Moon, H., Rizvi, S., Rauss, P.: The FERET Evaluation Methodology for Face-Recognition Algorithms. T-PAMI **22** (2000) 1090–1104

[4] FERET Database: http://www.itl.nist.gov/iad/humanid/feret/. NIST (2001)

[5] Turk, M. A., Pentland, A. P.: Face Recognition Using Eigenfaces. In: Proc. of IEEE Conference on Computer Vision and Pattern Recognition. (1991) 586 – 591

[6] Beveridge, J. R.: The Geometry of LDA and PCA Classifiers Illustrated with 3D Examples. Technical Report CS-01-101, Computer Science, Colorado State University (2001)

[7] Kirby, M.: Dimensionality Reduction and Pattern Analysis: An Empirical Approach. Wiley (2000)

[8] Breslow, N. E., Clayton, D. G.: Approximate inference in generalized linear mixed models. J. Amer. Statist. Assoc. **8** (1993) 9–25

[9] Rao, C. R.: Estimation of variance and covariance components in linear models. J. Amer. Statist. Assoc. **67** (1972) 112–115

[10] LaMotte, L. R.: Quadratic estimation of variance components. Biometrics **29** (1973) 311–330

[11] Wolfinger, R., O'Connell, M.: Generalized linear models: a pseudo-likelihood approach. Journal of Statistical Computation and Simulation **48** (1993) 233–243

[12] Wolfinger, R., Tobias, R., Sall, J.: Computing gaussian likelihoods and their derivatives for general linear mixed models. SIAM Journal of Scientific Computing **15** (1994) 1294–1310

Discrete Techniques for Real-Time Inspection and Interaction in Virtual Reality

P. Brunet

UPC, Barcenola, Spain
pere@lsi.upc.es

Abstract. Virtual Reality (VR) Systems allow immersive inspection of very complex environments, with direct interaction with the objects in the virtual scene. Low cost and affordable VR systems are becoming essential in application areas like Industrial Design, Medical Applications and Cultural Heritage. The specific requirements in these areas will be discussed, together with the advantages of using VR devices and immersive inspection techniques.
Discrete geometric models can very useful for some involved geometry processing algorithms like model repair, occlusion culling and mutiresolution and level of detail. The interest of these techniques and their potential in present and future VR applications will be discussed. The talk will also address the present limitations and future perspectives of VR techniques in medicine, industrial design and cultural applications.

Progress in Sign and Gesture Recognition

Richard Bowden

CVSSP, EPS, University of Surrey
Guildford, Surrey, GU2 7XH, UK
r.bowden@surrey.ac.uk

Abstract. Sign Language is a visual language (without a conventional written form) and consists of 3 major components, 1) Fingerspelling – used to spell words on a letter by letter basis, 2) Word level sign vocabulary – used for the majority of communication, 3) Non manual features – Facial expressions, tongue/mouth position and body posture used to modify sign meaning. Todate no system exists which has sufficient reliability and vocabulary to convert sign to speech at the level required for translation. This talk will present our work towards this goal. The talk will cover static pose recognition and tracking of the human hand, followed by the use of boosting for head and hand detection. It will be shown how prior statistical models of body configuration can be used to locate body parts, disambiguate the hands of the subject and predict the likely position of elbows and other body parts. Finally, it will be shown how these components are combined in a novel two-stage classifier architecture that provides a flexible monocular vision system capable of recognising sign lexicons far in excess of previous approaches. The approach boasts classification rates as high as 92 training instance per word, outperforming previous approaches where thousands of training examples are required.

A Haptic Sculpting Technique Based on Volumetric Representation

Laehyun Kim and Se Hyung Park

Korea Institute of Science and Technology, System Research Division
39-1 Haweolgok-dong Sungbuk-gu, Seoul, Korea
{laehyunk,sehyung}@kist.re.kr

Abstract. We present a novel haptic sculpting technique where the user intuitively adds to and carves out material from a volumetric model using virtual sculpting tools in the similar way to handling real clay. Haptic rendering and model deformation are implemented based on volumetric implicit surface extracted from geometric model. We enhance previous volume-based haptic sculpting systems. Collision and force computation are implemented on an offset surface rather than the implicit surface to provide consistent contact sensation in both visual and haptic displays. In order to bridge the gap between fast haptic process (1 KHz) and much slower visual update frequency (30Hz), the system generates intermediate implicit surfaces between two consecutive physical models being deformed resulting in smooth force rendering. Magnetic surface established between an offset and the implicit surface is used to simulate a feeling of pulling in adding operation. The volumetric model being sculpted is visualized as a geometric model which is adaptively polygonized according to the surface complexity. We also introduce various visual effects for the real-time sculpting system including mesh-based solid texturing, painting, and embossing/engraving techniques.

Keywords: Haptic sculpting, Volumetric implicit surface, Interactive shape modeling, Haptic rendering

1 Introduction

By sculpting 3D digital models in virtual environment, we can easily create the desired model. In addition, haptic interface allows the user to perform the sculpting process in a natural way using the sense of touch. In recent years, many haptic sculpting systems have been proposed for virtual design, art, or surgical training. *FreeForm* [5] is the first commercial haptic sculpting system from SensAble. However, they still have many limitations for realistic sculpting.

In this paper, we introduce volume-based haptic sculpting techniques to attack these limitations as follows:

– Using existing geometric models as an initial volumetric model in the sculpting process instead of simple volumetric models such as box, sphere, and cylinder.

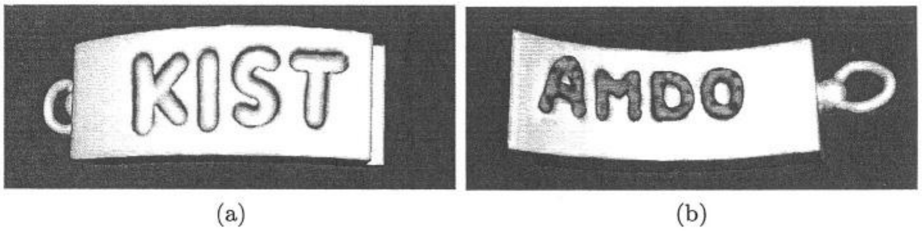


Fig. 1. A key holder created by our haptic sculpting system. (a) front side (b) back side

- Maintaining shape features of volume model being sculpted using an adaptive polygonization method in which the generated mesh is adapted to surface complexity.
- Smooth force rendering by directly simulating intermediate implicit surfaces created by interpolating potential values between two consecutive physical models being sculpted.
- Simulating the pulling (adding) operation using virtual magnetic surface which attracts the tool tip within magnetic field established between the implicit surface and an offset surface.
- Simulating internal material properties including stiffness and color saved into a volumetric representation.
- Visual effects, such as mesh-based solid texture and painting operation.

Fig. 1 shows a key holder with KIST and AMDO logos created by our haptic sculpting system.

Our haptic system consists of two basic parts: the visual and haptic processes run on a PC with dual Intel Xeon 2.4-GHz CPUs, 1 Gbyte of RAM, and FireGL X1 video card. The visual rendering implementation (including simulation) is OpenGL based and a 3-DOF Phantom haptic device for haptic display.

We proceed with a discussion on previous related work in section 2 and implicit surface in section 3. Section 4 discusses our haptic model and section 5 describes our volume-based haptic sculpting techniques. Visual effects including mesh-based solid texturing and painting are described in section 6. We conclude and suggest future work in section 7.

2 Related Work

2.1 Haptic Rendering Algorithm for Stiff Objects

Traditional haptic rendering methods can be classified into roughly three groups according to the surface representation used, such as geometric, volumetric (including implicit), or parametric.

One approach for the geometric models is the constraint-based method. Zilles and Salisbury [21] introduced a constraint-based "god-object" method in which the movement of a god-object (a virtual contact point on the surface) is constrained to stay on the object's surface. Ruspini et al. [16] proposed a more general constraint-based method and simulating additional surface properties such as friction and haptic texture.

In the haptic rendering for volumetric data, the force field is computed directly from the volume data without conversion to geometric model. A haptic rendering algorithm for volumetric data was introduced by Avila and Sobierajski [1]. They used the gradient of the potential value to calculate the direction of the force and the amount of force was linearly proportional to the potential value.

Kim et al. [8] used a hybrid surface representation which is a combination of geometric model for visual rendering and implicit surface for haptic display to take advantage of both data representations.

Thomson et al. [18] introduced a haptic system to directly render *NURBS* model using the advantage of mathematical properties of parametric surface representation.

2.2 Force Rendering on 3D Model Being Sculpted

While sculpting, typically, visual frequency (30 Hz) is much slower than haptic update rate (1 KHz). This performance gap makes the force rendering discontinuous if the force is computed directly on the physical surface being sculpted.

In order to smooth the force rendering, [4] and [12] use spring-based force established between the initial contact point on the surface and tool tip position. The user can move the tool tip freely and force computation is independent of the surface.

Another method is "proxy blending" [17] which smoothly interpolates the goal point from old proxy point constrained on the old model to the new proxy point on the new surface. During the blending period, the user can not adjust the blending speed and direction since the new surface should be defined when the blending starts.

2.3 Volume-Based Sculpting System

Galyean and Hughes [6] introduced a voxel-based approaches to volume sculpting that used the marching cube algorithm to display the model. Barentzen [2] proposed to use octree-based volume sculpting to reduce the memory requirement. However, these approaches have limitation such as low resolution due to the data size of the volume, or the large number of triangles for the displayed surface. In order to address these limitation, multi-resolution [7] and adaptive approaches [14] have been suggested resulting in high image quality with less number of triangles.

Mouse-based computer interface in the 3D sculpting system is unnatural and inefficient inherently. Avila [1] introduced a volume-based haptic sculpting system which allows the user to intuitively sculpt a volumetric data using a haptic interface.

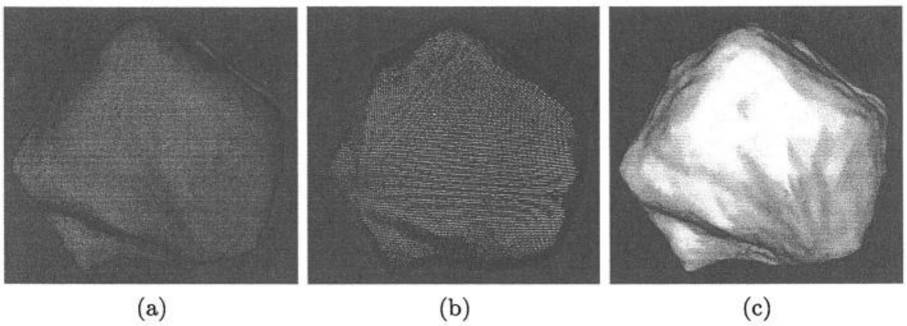


Fig. 2. Surface representation in our system (a) original geometric model. (b) volumetric implicit surface extracted from (a) using a closest point transform. (c) geometric model created from (b) using an adaptive polygonization method

3 Implicit Surface

We now briefly describe some of the properties of the implicit surface representation that are used for our haptic rendering algorithm. The implicit representation of the external surface S of an object is defined by the following implicit equation [3]:

$$S = \{(x, y, z) \in R^3 | f(x, y, z) = 0\}$$

where f is the implicit function (also called potential), and (x, y, z) is the coordinate of a point in 3D space.

The set of all points for which the potential value is 0 defines the implicit surface. If the potential is positive, then the point (x, y, z) is outside the object. If $f(x, y, z) < 0$, then the point (x, y, z) is inside the surface. The surface normals of an implicit surface can be obtained using the gradient of the implicit function as follows:

$$n = \nabla f / ||\nabla f|| \tag{1}$$

$$\nabla f = [\frac{df}{dx}, \frac{df}{dy}, \frac{df}{dz}] \tag{2}$$

In order to create a volumetric implicit surface representation from a geometric model (Fig. 2a), we use a fast closest point transform (CPT) algorithm suggest by Mauch [11]. The algorithm computes the closest point to a surface and its distance from it by solving the Eikonal equation using the method of characteristics (Fig. 2b).

4 Haptic Model

In this section we give a detailed presentation of our basic haptic model.

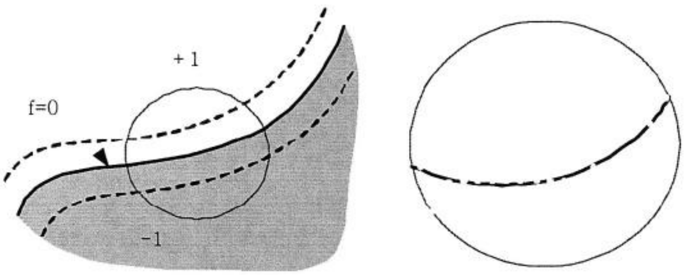


Fig. 3. Data representation for surface modeling and haptic rendering

4.1 Data Representation

In our algorithm, volumetric implicit surface is used for surface modeling and haptic rendering while sculpting. In the volumetric representation, only potential values close to the implicit surface (Fig. 3) are involved in the computation. The potential values inside the close neighborhood of the surface range from -1 to 1 according to the proximity to the closet point on the surface. The values inside the surface are negative and positive outside. The values out of this narrow band are nothing to do with the surface modeling and haptic rendering. Therefore, to reduce the memory requirement, we use an octree-based data structure, avoiding the representation of empty portions of the space. In addition, the system can search and manage the data in such an octree very fast.

4.2 Collision Detection

Most haptic rendering algorithms use only one point of the virtual tool for collision detection and force computation. However, the visual contact interaction between the sculpting tool and model surface may not correspond with the haptic feeling on on complex models with shape edges or small holes. To address this problem, Persik [15] uses multiple points distributed on the sculpting tool to detect collision and calculate the force in a petrous bone surgery simulator. However, when the number and position of the points involved in force computation is changed, the force rendering can be unstable.

In our system, we use a single point of the sculpting tool but collision detection and force computation are performed on an offset surface. The offset value from the implicit surface is determined by the radius of bounding sphere of the tool. The single point becomes the center point of the bounding sphere. If the point collides with the offset surface, the system computes the force vector on the offset surface rather than the implicit surface (Fig. 4). This approach provides consistent contact information in both visual and haptic feeling and stable force rendering.

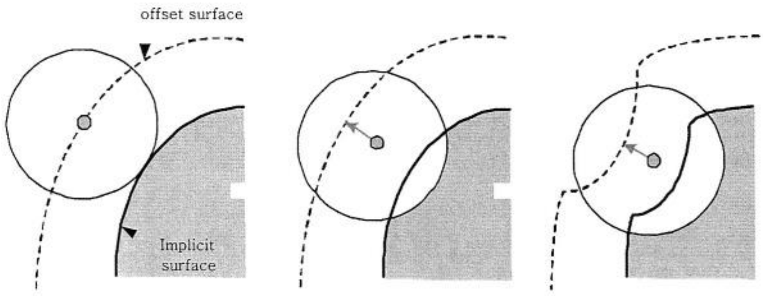


Fig. 4. Collision detection and force computation based on offset surface

4.3 Force Vector

The force direction is computed by interpolating the gradients of 8 cell points around the tool tip. The interpolation function makes the system avoid the force discontinuity. In order to determine the amount of force, we first find the virtual contact point (VCP) on a surface which is the intersection point between the surface and a ray along the computed force direction. The amount of force is proportional to the distance between the VCP and the tool tip. Once the VCP is determined, a spring-damper model [20] is used to compute the final force vector that tries to keep the virtual contact point and the tool tip at the same position.

$$\boldsymbol{F} = k\,(\mathbf{p_c} - \mathbf{p_t}) - b\,\boldsymbol{V} \tag{3}$$

where $\boldsymbol{F}$ is the force vector, $\mathbf{p_c}$ is the coordinate of the contact point, $\mathbf{p_t}$ is the coordinate of the tool tip, k is stiffness, V *is* velocity of the tool tip, and b is viscosity. Spring stiffness has a reasonably high value and viscosity is to prevent oscillations.

4.4 Magnetic Surface

Kim et al. [9] first proposed a magnetic surface which attracts the tool tip to the closet point on the surface if the tool tip is within the virtual magnetic field established between the offset surface and the original surface. It forces the tool tip to stay in contact with the surface while the user explores complex 3D models.

We employ the magnetic surface to simulate the feeling of ‘pulling’ while the adding operation in haptic sculpting system (section 5). The direction of magnetic force is determined by the surface normal at the position of tool tip and the amount is proportional to the distance between the tool tip and the closest point on the original surface.

5 Volume-Based Haptic Sculpting Technique

This section describes our haptic sculpting techniques.

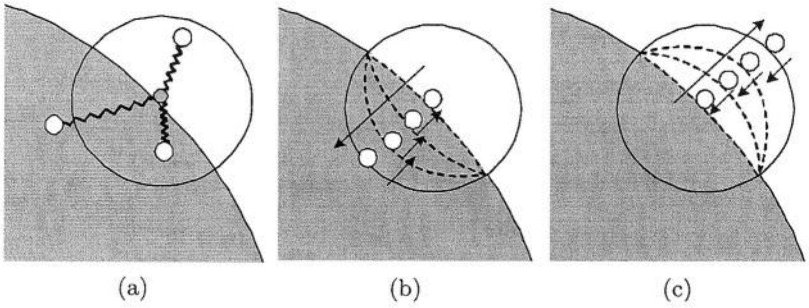


Fig. 5. Spring-based approach and our method. (a) Spring-based sculpting (b) Carving operation in force-based sculpting (blue arrow: moving direction of the tool, red arrow: responsive force) (c) Adding operation in force-based sculpting

5.1 Polygonization Method

Model modification during sculpting is implemented by updating the local potential values around the sculpting tool, depending on the shape and size of the tool and operation type (carving or adding). The volumetric model is converted into the geometric model every graphical frame for the visual rendering. Many sculpting systems [6, 2] use a uniform polygonization method such as Marching cube [10] which suffers from large triangle count and a resolution limited by a fixed sampling rate.

In order to address these limitations, we employ the adaptive polygonization method suggested by Velho [19]. This adaptive method consists of two steps. In the initial polygonization step, a uniform space decomposition is used to create the initial mesh that serves as the basis for adaptive refinement. In the second step, for adaptive polygonization, the mesh is refined adaptively according to the surface curvature until the desired accuracy is achieved. The resulting mesh effectively represents sharp edges with a smaller number of triangles compared to uniform polygonization methods (see Fig. 8(c)). The volumetric implicit surface and generated mesh are saved into an octree-based data structures to locally manage the data.

5.2 Sculpting Mode

In our system, there are two sculpting modes; haptic sculpting with visual and force feedback and block sculpting with only visual feedback.

Force-Based Sculpting The system typically could not update the physical model at a rate sufficient for haptic rendering (at least 1 KHz). Due to the performance gap, most previous haptic sculpting systems could not directly simulate physical models being deformed. Instead, a virtual spring-based force [12] (Fig. 5(a)) established between the current tool tip and the initial position is

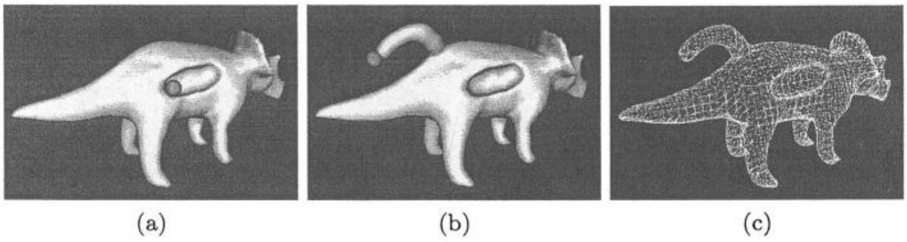


Fig. 6. Models created in force-based sculpting mode (a) carving operation (b) adding operation (c) mesh model

used to provide feedback. However, the user cannot directly feel the physical surface being deformed and is not allowed to move the tool tip to sculpt along the surface without retouching the surface.

To bridge the gap between the update rate of the visual and haptic rendering, we introduce an intermediate implicit surface smoothly transiting between discrete old model and target models in haptic process. The target model is defined by CSG(Constructive Solid Geometry) point-set operation based on the old model and sculpting tool.

The system updates the potential values inside the close neighborhood of the intermediate surface on which haptic rendering is performed. The intermediate surface moves toward the target surface in the haptic process at the variable speed depending on the applied force by the user and the local material properties such as stiffness (see Fig. 5(b),(c)). When the system meets the next visual frame, it updates the physical model using current intermediate implicit surface instead of the target surface.

Unlike pushing operation, pulling operation makes the tool tip leave the surface and looses the contact point immediately. In order to simulate a feeling of pulling, we use the *Magnetic surface* (section 4), which attracts the tool tip to the closest point on the implicit surface being sculpted. It allows the user to intuitively perform the adding operation (see Fig. 5(c), 6(b)).

Fig. 6 shows examples created by carving and adding operations in haptic sculpting mode.

Geometry-Based Sculpting The haptic force often hinders the user from performing accurate and detailed sculpting. In block sculpting mode, the system turns off the haptic force and allows the user to freely move in any direction in 3D space.

To sculpt a virtual object in this mode, the user first moves the sculpting tool onto the desired region using visual feedback. The sculpting tools are represented by simple wire frame showing the boundary to be sculpted. When the switch on the stylus is pressed down, material is either added or carved from the surface at

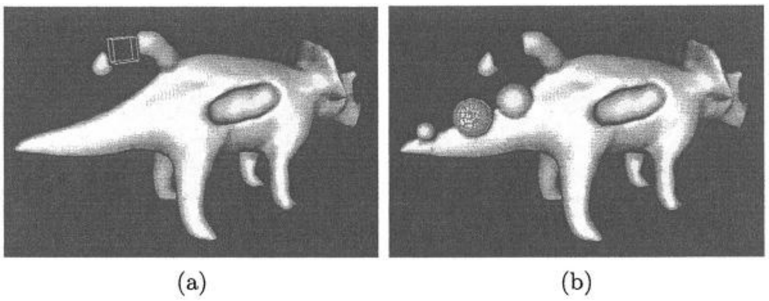


Fig. 7. Models created in geometry-based sculpting mode (a) caving operation (b) adding operation

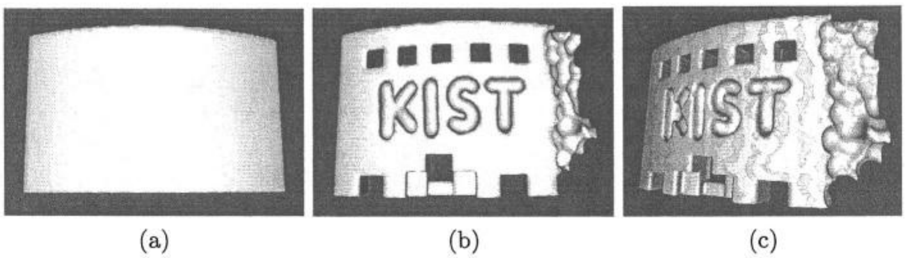


Fig. 8. Examples created by our haptic sculpting system. (a) Original model. (b) The model applied various sculpting tools. (c) The model with mesh-based solid texturing

once. Target potential values at grid points to be sculpted are computed by CSG boolean operation. (Fig. 7) shows examples sculpted in block sculpting mode.

5.3 Volume Material Properties

While sculpting, material properties such as stiffness inside the model affect the responsive force. In our system, volumetric properties including stiffness and color are saved into the volumetric representation and simulated while sculpting. The system calculates the local stiffness by averaging stiffness values of the grid points within the sculpting tool. Volumetric color information represents internal color of the model. The local mesh being sculpted is adapted to the local color complexity in the same way to painting in section 6 as well as surface complexity.

6 Visual Effects

We use the adaptive polygonization method to enhance visual realism for the sculpting system. The mesh is adaptively refined according to the detail of color in solid texturing and painting. The system also create easily embossed or engraved model by modulating potential values.

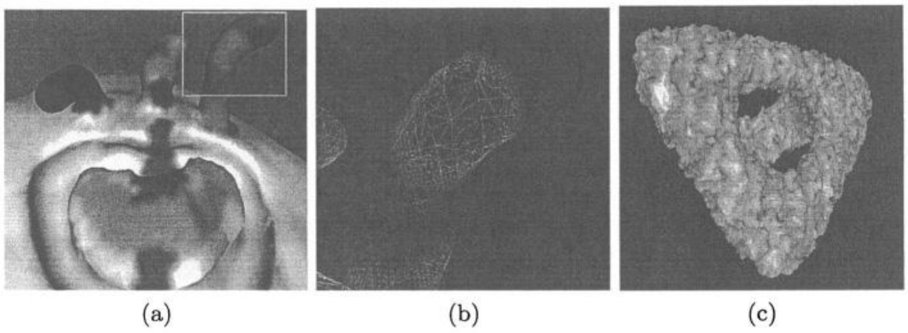


Fig. 9. Mesh-based visual effects. (a) the model created by sculpting and painting tools. (b) mesh data in the rectangular of (a). (c) the model created by embossing tool

6.1 Solid Texturing

In order to avoid texture distortion for the sculpted model, we employ the *Solid Texture* method introduced by Perlin [13], which uses a procedural function to calculate the color for each point, instead of a look-up texture image. However, solid texture rendering such as *ray marching* and *solid map* requires high computational power or a large texture memory which are not suitable for a real-time sculpting system.

In our system, color generated from a noise function is applied to the surface by assigning per-vertex colors to make the solid texturing process go faster. However, the color of each vertex blends with nearby vertices and much of the detail of the solid texture is often lost depending upon the surface complexity. To address this limitation, we use the adaptive polygonization approach in which the mesh is adapted to the detail of the solid texture. If the difference between per-vertex colors along a edge is greater than a desired accuracy, this edge is spit (see Fig. 9(b)) recursively. As a result, we obtain a clear solid texture on the surface in real-time without special hardware and graphical rendering method (see Fig. 9(a)).

6.2 Painting

The user also can paint directly on the surface while or after sculpting. The applied color is saved at grid points around the painting tool and are used to determine vertex color while the mesh is adaptively polygonized in a similar manner to the mesh-based solid texturing. It makes the color boundaries clear (see Fig. 9(a),(b)). The size of the brush volume is proportional to the force applied by the user in a natural way. If the user pushes hard the tool against the surface, the brush volume is getting bigger.

6.3 Embossing and Engraving

The user can easily create embossed or engraved models along the solid texture or painted image. It is implemented by modulating potential values in volumetric representation (see Fig. 9(c)).

7 Conclusion

We introduced a novel haptic sculpting technique based on a volumetric representation which contains implicit surface, material properties, such as stiffness and color information. Based on this idea, we have implemented smooth force rendering on the offset surface while sculpting. We simulate volume material properties such as stiffness and color and introduce mesh-based solid texture and painting for better visual quality.

We plan to enhance the haptic sculpting system by adding various sculpting tools such as smoothing and stamping, modeling by sketching and a cut-and-paste operator for fast editing. We also plan to apply this technique to various applications such as medical training simulators and digital art.

References

[1] R. S. Avila, L. M. Sobierajski, "A Haptic Interaction Method for Volume Visualization" IEEE Visualization proceedings, (1996) 197-204

[2] J. Andreas Barentzen, "Octree-based Volume Sculpting", IEEE Visulalization proceedings, (1998) 9-12

[3] J. Bloomenthal, et al. "Introduction to Implicit surface", Morgan Kaufmann Publishers, Inc. (1997)

[4] Mark Foskey, Miguel A. Otaduy, and Ming C. Lin, "ArtNova: Touch-Enabled 3D Model Design", IEEE Virtual Reality proceedings, (2002) 119-126

[5] SensAble Technologies Inc. freeform modeling system, http://www.sensable.com/freeform, (1999)

[6] Tinsley A. Galyean, John F. Hughes, "Sculpting: An interactive volumetric modeling technique", ACM SIGGRAPH proceedings, (1991) 267-274

[7] Jing Hua, Hong Qin, "Haptic Sculpting of Volumetric Implicit Functions", The ninth Pacific Conference on Computer Graphics and Applications, (2001)

[8] Laehyun Kim, Gaurav S. Sukhatme, Mathieu Desbrun, "A Haptic Rendering Technique Based on Hybrid Surface Represenation", IEEE computer graphics and applications as a special issue of Haptic Rendering: Beyond visual computing, Vol. 24(2), (2004) 66-75

[9] Laehyun Kim, Gaurav S. Sukhatme, Mathieu Desbrun, "Haptic Editing for Decoration and Material Properties", in 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, (2003) 213-221

[10] W. E. Lorensen and H. E. Cline, "Marching Cubes: a high resolution 3D surface reconstruction algorithm," Computer Graphics, Vol. 21, No. 4, (1987) 163-169

[11] Sean Mauch, "A Fast Alogorithm for Computing the Closest Point and Distance Transform" Technical Report at Caltech. http://www.its.caltech.edu/ sean/ (2000)

[12] Kevein T. McDonnell, Hong Qin and Robert A. Wlodarczyk, "Virtual Clay: A Real-time Sculpting System with Haptic Toolkits", ACM Symposium on Interactive 3D Techniques, (2001) 179-190
[13] Ken Perlin, "An Image Synthesizer", ACM SIGGRAPH proceedings, vol 19, No 3, (1985) 287- 296
[14] R. N. Perry, S.F. Frisken, "Kizamu: A system for sculpting digital charaters", ACM SIGGRAPH proceedings, (2001) 47-56
[15] A. Petersik, B. Pflesser, U. Tiede, K. Hohne, R. L euwer, "Realistic Haptic Interaction in Volume Sculpting for Surgery Simulation", IS4TM 2003, Lecture Notes in Computer Science 2673, Springer-Verlag, Berlin, (2003) 194-202
[16] Diego C. Ruspini, Krasimir Kolarov, and Oussama Khatib. "The haptic display of complex graphical environment" ACM SIGGRAPH proceedings, vol. 1, (1997) 295-301
[17] Diego C. Ruspini, Oussama Khatib. "Dynamic Models For Haptic Rendering Systems", Advances in Robot Kinematics, (1998) 523-532
[18] T.v. Thompson II, E. Cohen, "Direct Haptic Rendering of Complex Trimmed NURBS Models", Haptic Interfaces for Virtual Environment and Teleoperator System, (1999)
[19] Luiz Velho, Luiz Henrique D. Figureiredo, Jonas Gomes, "A Unified Approach for Hierarchical Adaptive Tesselation of Surfaces", ACM Transactions on Graphics, Vol. 18, No. 4, (1999) 329-360
[20] A. Yoshitaka, T. Kumano, K. Ogino, "Intermediate Representation for Stiff Virtual Objects", IEEE Virtual Reality Annual Symposium, (1995) 203-210
[21] C. Zilles, J. K. Salisbury, "A Constraint-based God-object Method For Haptic Display", Haptic Interfaces for Virtual Environment and Teleoperator Systems, (1994) 146-150

Adaptation of Mesh Morphing Techniques for Avatars Used in Web Applications

Iker Aizpurua[1], Amalia Ortiz[1], David Oyarzun[1], Iosu Arizkuren[1], Ana C. Andrés[1], Jorge Posada[1], and Ido Iurgel[2]

[1] VICOMTech Research Centre, Edutainment and Tourism Dept.
Mikelegi Pasealekua, 57 bajo, 20009, Donostia-San Sebastián, Spain
{iaizpurua,aortiz,doyarzun,iarizkuren,acandres,jposada}@vicomtech.es
http://www.vicomtech.es

[2] ZDGV Computer Graphics Center, Digital Storytelling Dept.
Germany

Abstract. Multilingual 3D avatars combine rich emotion with multilingual speech gesticulation. They require complex animation. The most performing facial animation engines for avatars have been developed using morphing techniques. Unfortunately, these techniques cannot be directly utilized to generate animation through the Internet. The information needed to make them work is over the capacities of the connections of most Internet users. This article presents a solution to this problem. We have elaborated the adaptation of a very performing multilingual face animation engine to fit it into the Internet. We also evaluate the pros and the cons of such an adaptation from the animation quality perspective.

1 Introduction

Avatars are intelligent agents with graphical representation, often humanoid, that can realize several functions as useful interfaces in Virtual Environments. This kind of interfaces, called *"Social User Interface",* are based on an intuitive human-machine interaction, and use both, speech and non-verbal communication.

When avatars were first developed, researchers were mainly focused in making them the perfect help assistants. This initial role has been overcome by the increasing demand for useful graphical representations in daily life. On the one hand, users are accustomed to interacting with people and not machines when doing non technical tasks like shopping, studying, etc. On the other hand, technological advances in hardware and software in Computer Graphics are associated to the possibility of spreading high technology outside working environments, e.g., entertainment. In parallel, the Internet is becoming an important part in people's life, not only at work but also during their leisure time. It is the most inexpensive way to communicate with other places in the World. However, most people have problems to manage typical computer user interfaces because they have not been taught how to use these technologies. Avatars can help to improve communications within user interfaces.

We find the best applied examples about avatars in e-business [1, 2, 3, 4], e-leaning [5, 6, 7] or virtual environments [8, 9, 10, 11] where the Avatar's main mission is to interact with and motivate the user. We have learnt from experience that, in order to make avatars useful and attractive to non specialized users, it is necessary to take good care of at least the following points:

- *Language:* The user likes interacting in his/her mother tongue;
- *Emotions:* The avatar's emotive expressions should catch the user's attention and maintain his/her motivation;
- *Speed:* The avatar application must work in real-time so a feeling of natural interaction can be created.

The most performing avatars have been developed over their own platforms and for very concrete needs. As we have pointed out, the Web is becoming the *de facto* platform for human-to-human, human-to-machine interactions. Unfortunately, the use of avatars on the Web requires to previously install applications basically devoted to the animation and communications tasks, and generally, these applications consume much of the user's PC resources.

This article addresses the issue of adapting an off-line TTS[1]-driven avatar platform to make it work through the Internet. The designed on-line version does not require any special software or plugin that, so often, discourage people to utilize on-line animation engines. In order to achieve this goal, all the original morphing techniques have been adjusted and redefined, and software choices rethought.

Knowing that element size is critical, we present a deep study about how morph targets and software have been reduced in size. This problem is currently being tackled in the Computer Graphics community. In [12], we find the solution given by researchers at the MIRALab. They propose an MPEG-4 based system that animates avatars using Java applets. We agree on applets being the optimal solution not to force the downloading of specialized software, but our work focuses in reusing morphing techniques already well set and suitable for multilingual avatars, we believe MPEG-4 should not drive the animation solution but simply complement it. People at GRIS of TU Darmstadt have developed optimizations to the system we propose (see [13]). We have inspired our solution in their work and we have added the on-line capabilities for the Internet. Nevertheless, most of the solutions found in the literature, including the aforementioned ones, lack a visual evaluation of the quality improvement or worsening. In this article, we do not only study the technical details of the online adaptation but we also explain how we judge the adaptation in terms of visual quality.

Section 2 presents the original off-line platform. In Section 3, we explain the adaptations required on the facial animation engine in order to make it work through the Internet. Section 4 contains the experimental evaluation of the adapted on-line platform. And we conclude with some future perspectives in Section 5.

[1] Text-to-Speech Synthesizer

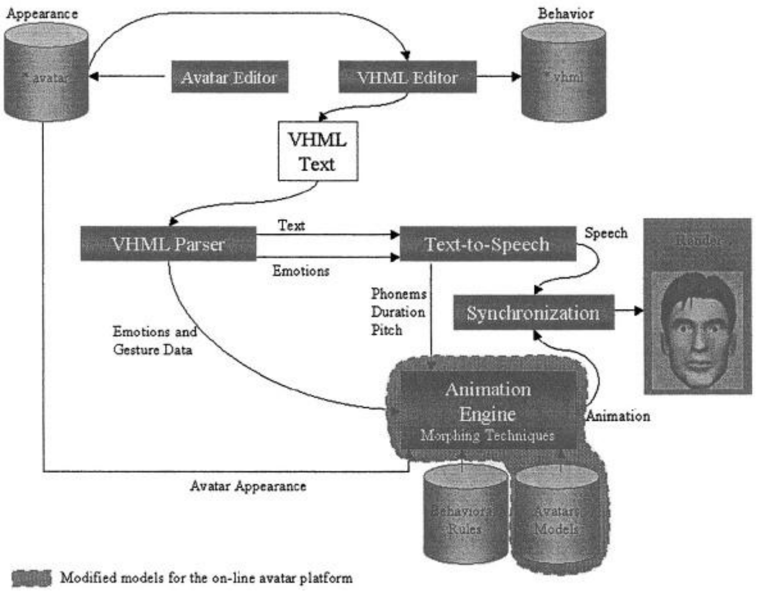


Fig. 1. Facial Animation platform architecture

2 Off-Line Animation Engine

2.1 Original Architecture

To achieve natural communication with avatars, it is very important both, to obtain correct synchronization between audio and lip motion, and to recreate natural facial expressiveness. Lip-synch and realistic simulation of facial expressions have been the pillars of our research. Fig. 1 shows the architecture of our TTS-driven avatar platform. The animation engine needs input of two origins, one comes from the mark-up text that controls the avatar's emotions and the other one comes from the synthesis motor that performs the lip-sync from the TTS results. These two animation processes work as follows:

Emotions: We divide facial expressiveness into two groups. First, we consider mood emotions such as normality, severity, happiness, sadness, or anger; and second, we have a set of emotional events like smiling, winking, denying, etc. Through the markup text, we associate and synchronize each emotion of the avatar. Our markup text counts on special labels that indicate: *mood* emotions, *emotional* events, some *special gestures* to be reproduced in certain instants of the speech, and *user-profile* dependent interactions. Our mark-up language is a selected mixture of VHML [14] (Virtual Human MarkUp Language)with some marks from EML (Emotional MarkUp Language) and GML (Gesture MarkUp Language). The *VHML Parser* interprets the VHML and extracts emotions and gestures, plus the exact moment when these have to be reproduced. This information is transferred to the graphic platform

(TTS + Animation Engine) for animating facial expressions using morphing techniques.

Lip Synch: The text to vocalize, as well as the emotions and events related to them, are transferred to the TTS module. The TTS calculates the chain of phonemes necessary to vocalize the message contained in the text with the indicated emotions, assigning to each phoneme its prosodic characteristics, mainly his duration and pitch. The TTS also produces the audio file that is synchronized with the animation data that is animated in the Animation Engine. Each phoneme will be associated to its corresponding viseme (visual representation of the phoneme) by means of morphing techniques. The final vocalized animation is the result of the output coming from the TTS and a set of internal behavioral rules (associated with emotions).

Animations from both origins use the morphing techniques explained in the next section.

2.2 Animation Techniques: Morphing Implementation

Our facial animation techniques derive from the adaptation of a previous platform developed by the ZGDV group. Facial expressions are obtained through the animation of the head, lips, eyes, pupils, eyebrows and eyelids. These animations are easily mapped for humanoids. We refer the reader to [15] for more details about the original platform. We have also implemented some additional typical animations for cartoon avatars, like movements for their ears or nose. Some animations are generated making individual deformations or translations over the object in a determined trajectory. This is the technique used for the pupils or the global head pose. Some other animations, like lip motion, are achieved using the morphing techniques developed by Alexa [16].

Let us briefly summarize this technique: first, we establish an appropriate set of basic objects (B_i in Fig.2 and Eq. 1), in such a way that all face expressions necessary to produce the animation can be obtained from combinations of these basic objects. We use different sets of basic objects formed by groups of 14 to 22 morph targets depending on the avatar and the parts of the face that we want to animate. In any case, 8 basic objects are used for phonemes, and the remaining for emotional expressiveness and natural facial gestures (e.g., blinking). It is necessary to have a flexible set of morph targets depending on the personality of the Avatar because this set will define the facial expressiveness. For instance, for Arrati, a 3D cartoon cow used in Basque television [17], we used 16 different morph targets. Arrati usually speaks Basque [18], although her 8 basic visemes allows her to speak English, Spanish and German. It has also rich expressiveness thanks to the combination of the other 8 morph targets.

One geometric base and a set of key frames (defined by a vector of weights) represent animations. Each value of this vector corresponds to the interpolated value (a_i in Fig.2 and Eq. 1). The sequences of animations are set defining the

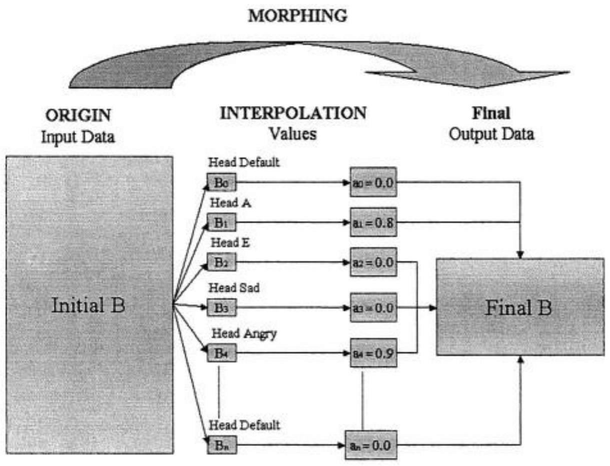


Fig. 2. Morphing adaptation

operations in eq. 1 with the required input values, Fig.2 illustrates this process.

$$V(i) = \sum_{j=1}^{n-1} a_i B_i = (\sum_{j=1}^{n-1} a_i V_{ij}) \tag{1}$$

3 Adaptation of the Animation Engine to the Internet

The facial animation architecture presented in Fig. 1 needs two major changes in order to meet the requirements of software platform independence and animation efficiency: the TTS must be outside the user's PC and the Animation Engine should run on a universal platform. The client-server structure chosen for our adapted platform is detailed in Section 4. In this section we explain the major changes made on the Animation Engine in terms of morphing and software.

The previously detailed Facial Animation Engine is based on the Alexas animation method. This technique relies on a Java 3D morph Node and uses each morph target for creating the keyframes of an animated sequence. To adapt this method to use it on an Internet platform, we have focus our work mainly in two actions. First, we have developed an efficient way to animate and store our morph targets. Second, since for us the development of a web application implies not forcing the user to download any plugin, we had to discard the use of the Java 3D library, instead we propose a platform where the morphing algorithms have been developed using the Anf3D libraries [19].

3.1 Creation of Key Frames

For the generation of the key frames we need to mix different type-expressions on models of the avatar, which has a static expression, such as, happy-face, angry-face or any phoneme-face.

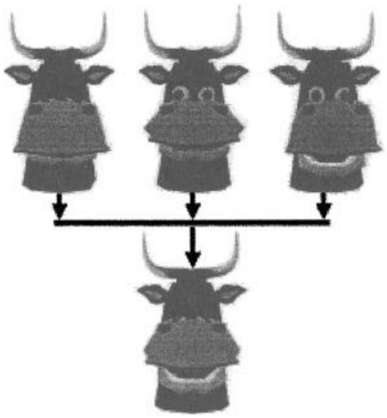


Fig. 3. Illustrating the Morphing technique during animation

For example, in the Fig.3 we can see the morph targets used for generating one keyframe: using three morph targets (cow blinking, cow smiling and cow opening the mouth), we can make a cow blinking, smiling and opening the mouth for saying "ah" with this emotion. As is said in [16], for the generation of each key frame creation we have to calculate a new geometry where the coordinates of each vertex it's the sum of the same vertex of the morph targets. Using the previous example:

$$V(i) = p(1) \cdot EyesClosed(i) + p(2) \cdot HappyV(i) + p(3) \cdot MounthOpenedV(i) \quad (2)$$

Where $V(i)$ is the i vertex and $p(1)$, $p(2)$, $p(3)$ are the model weights to sum:

$$V(i) = \sum_{j=1}^{n} p(j) \cdot V_j(i) \quad (3)$$

Mixing the models with different weights we obtain an extended set of facial expressions that permit rich emotional person-machine interaction.

3.2 Our New Storage Method

Morphing-based animation techniques have proved to be the most performing methods in terms of animation quality but they have high weight cost. For example, in our off-line system, animating a small emotionless avatar represents using 851KB (14 targets x 61KB per target with 393 vertices per model). Unfortunately, most Internet connections are not fast enough for instant downloading of this amount of data.

To decrement the downloadable information we store the neutral-expression face and the differences between this neutral expression and the other face targets, instead of all the needed face models. Generally, the differences between neutral-expression and other expressions, as a happy-face or any phoneme-face,

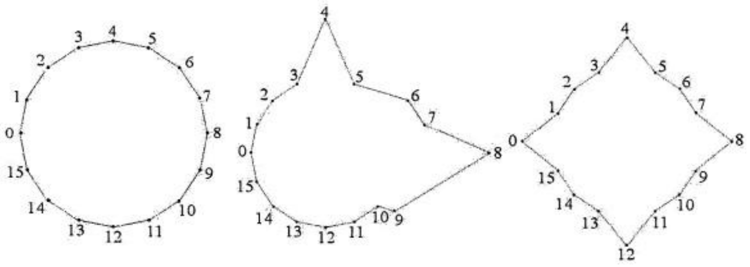


Fig. 4. Example of our storage method in 2D. a) Original Model, b) Expression 1, and c) Expression 2

are significant only on a few vertices. This techniques follows the same philosophy as the MPEG-4 [20] coding for vertex motion, but it is applied to a morphing-based facial animation technique. Let us illustrate our method in 2D for better understanding. In Fig. 4a we can see a polygon with 16 vertices. Fig.4b and the Fig.4c show the same vertices after actions have been applied. Using the original model and difference vertices with the other morph targets it is possible to reconstruct all expression models. In our adapted platform we have implemented a data structure based in 2D matrices that contain the required data. The first row of these matrices is granted to the vertex identifiers, and the remaining ones contain the X, Y, Z coordinates of each vertex. For the 2D example, we would obtain Table 1.

The new data structure reduces the size of the information we need to store. The reason for that being:

- *Invariable vertices:* in all humanoid heads, more than half of head points are invariable during face animation. Only the front part of the face has mobile points.
- *Polygonal and material information:* all models have the same polygonal information (material, texture) that is downloaded once only.

Table 1. Matrices stored for the 2D example. a) For Fig. 4a, b) for Fig. 4b

4	6	7	8	9
X_4	X_6	X_7	X_8	X_9
Y_4	Y_6	Y_7	Y_8	Y_9

0	4	8	12
X_0	X_4	X_8	X_{12}
Y_0	Y_4	Y_8	Y_{12}

3.3 Our New Animation Method

Since we store information in a new way, we need to change the animation method and use new data structures. As we previously said, avatar animation requires two phases: creation of the new key frames, and the interpolation between the last key frame and the next one. Repeating this process generates the animation. The key frame creation is now simple and efficient because all key frames are made from the neutral-expression model after being modified. That is to say, using the neutral model, for all modified vertices and only for these vertices, we have to do the modified weighed sum:

$$V(i) = \sum_{j=1}^{n} p(j) \cdot (V_j(i) - V_{def}(i)) \tag{4}$$

Where $V(i)$ is the vertex i of the new key frame, $p(j)$ is the weight of the model j to sum, $V_j(i)$ is the vertex i of the model j and $V_{def}(i)$ is the vertex i of the neutral model. At this point, the interpolation is only computed on the modified vertices by means of listing the indexes of all modified vertices. Since the required list will be static thorough the process, we can compute it the moment we calculate the new data structures. In the example we have used before, this would imply being aware of the data stored in the first row of Table 1.

The table computation is done in a pre-process before the animation is set. The use of these tables has two advantages: the downloading time will be reduced, and animation will be more efficient. Section 4 provides a deep evaluation that enhances the aforementioned advantages.

To evaluate the prototype we have used three avatars. Each one of them with different characteristics (polygonal weight, number of models –morph targets– per avatar, etc.).

Table 2. Evaluation of the reduction improvement with our method

	Patxi	Betizu	Arrati
Number of morph targets	14	14	16
Animated parts without morphing (KB)	64.0	161.00	112.0
Morphing objects on-line (KB)	77.0	211.0	338.0
Morphing objects off-line (KB)	854.0	2198.00	3664.0
% of reduction	**90.98 %**	**90.40 %**	**90.77 %**
All objects after Gzip on-line (KB)	59.2	185.0	240.7
All objects after Gzip off-line (KB)	330.2	584.2	1028.0
% of reduction	**82.1 %**	**68.3 %**	**76.6 %**
Vertices where morphing is applied on-line	106	313	604
Vertices where morphing is applied off-line	393	835	1231
% of reduction	**73.0 %**	**62.3 %**	**51.0 %**

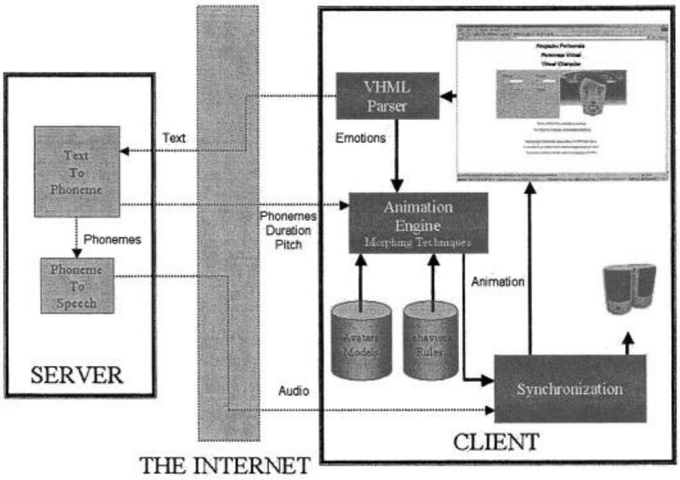


Fig. 5. The Internet architecture

4 Evaluation over an Application Prototype

4.1 Description of the Application Prototype

Our prototype is an avatar that inquires a Web user about some personal data, like name or place of birth, in order to create a customized story for him/her. Our technical approach consists of one applet composed by a set of java classes zipped in a low weight .JAR (optimized by our new storage method). The visitor can connect to our Web page and visualize the avatar without downloading any accessory application, because, the animation can run on the java virtual machine (jvm1.1) integrated by default in the majority of browsers. All executions are transparent to the visitor. As previously stated, the Graphical API used is called Anfy3D [19], available for jvm1.1, but the method we are presenting can be utilized with any other real time java 3D rendering library. Fig.5 illustrates the architecture of our system. The present architecture does not perform any streaming of audio or animation, thus making our solution suitable for dialogs and short speeches. The system needs to download the entire audio from the server in order to synchronize facial animation and speech. In practice, we are synthesizing long speeches by utilizing signed applets that can access the user's hard disc where the TTS has been previously installed. Streaming capabilities should be implemented in order to free the user from downloading any software related to the TTS. This issue was out of the scope of our research and herein we present the results from evaluating the animation engine designed.

In Fig.6, we present the three avatars used: Patxi, Arrati and Betizu. In Table 3, we can see the size in KB of all the components needed by the Internet Avatar Platform. In the "Other parts" row we see the size of Patxi's hair, Arrati's body and Betizu's horns and body.

Table 3. Total load in KB for each avatar animation without any compression

	Patxi	Arrati	Betizu
Head	61.0	229.0	157.0
# of morph targets	14.0	16.0	14.0
Other parts (globe, iris, pupil...)	64.0	112.0	161.0
Textures (jpeg)	27.5	28	34.8
Java classes	60.0	61.0	60.0
Total	1005.5	3865.0	2443.8

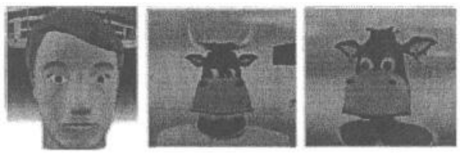


Fig. 6. Avatar models used for the tests. a) Patxi, b) Arrati, c) Betizu

4.2 Quantitative Evaluation

In Table 2, some of the improvements of our system are shown. The table is subdivided in three parts.

In the first five rows we can see the number of morph targets used for each avatar (Patxi, Betizu and Arrati) and the weight used for each application, on-line and off-line. We can appreciate that we manage to reduce up to the tenth part of the weights needed for the off-line application with the on-line storage method.

In the following three rows, the same comparative is shown after the Gzip compression method has been applied. We notice that the reduction of the total weights is around 24%. As it is described in the previous section, the method permits to interpolate between variable vertices thus reducing the number of operations required to achieve the animation and making the animation more efficient. We have estimated the algorithmic-efficiency improvement in around 50 % to 70 %, depending on the avatar's size, as it is shown in the last row of Table 2.

The performance of the designed off-line system allows us to integrate our avatars in any Web-based application. This has been the result of the decrease on the downloadable information and the improvement of the efficiency on the new system developed for the animation transmission through the Internet.

4.3 Qualitative Evaluation

Adapting an off-line animation engine to make it run on-line should not imply loss in animation quality. We have visually evaluated the performance of the off-line facial animation against the original on-line animation results (which we

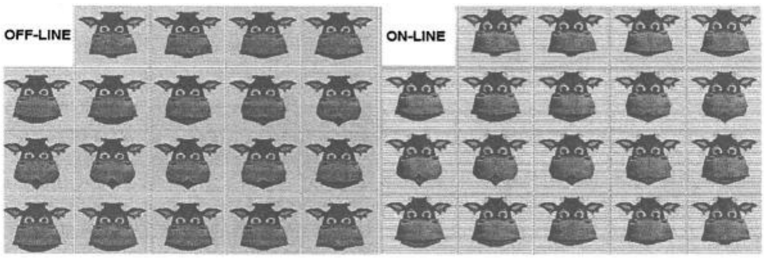


Fig. 7. Frames obtained from: a) Off-line application and b) On-line application

consider to be ground-truth). From this visual evaluation we draw one major conclusion: the change in visual quality is due to the 3D rendering applet (based on Anfy3D) and not to the animation off-line adaptation.

In order to verify this fact, we have compared animation sequences in pairs: on sequence coming from the on-line animation and the other from the off-line. The test shown in Fig. 7 was made with two animations generated for the avatar to reproduce the text "ah-uh-ah". The animation was slowed down to 5000 ms for each phoneme in order to appreciate the details. The slides shown are taken from the animation every 30 frames.

In Fig. 7 we notice that the renderings from the platforms are different, this is because in the on-line and off-line applications use different graphic libraries, Anfy3Dand Java3D respectively, that use different kinds of smoothing and illumination rendering. Focusing on the animation method, we notice that the same frames are generated at the same time by both applications, producing the same animation in both platforms. Based on the tests, we can conclude that, concerning the animation method, the quality of the animation is the same for on-line and off-line platforms.

5 Conclusions and Future Perspectives

The main advantage of the adaptation framework proposed in this article is the possibility of having an interactive web application using avatars that rely on the real user needs: diversity of speech (lip-synch), naturalness (emotions) and real communication (fast transmission). In this paper we have proposed a method to store and animate avatars through the Internet using key frame animation techniques, and morph targets. Thanks to this method, along with the use of ordinary compression, we have reduced the information needed to download for avatar animation over a total of 89 %.

Moreover, with this method we automatically identify the variable vertices, and make the interpolation computations, only in the required points. Therefore, we have reduced the calculations by around 37%.

We consider that Avatars are one of the best tools available to create proper user interfaces with natural effect because they imitate verbal and non-verbal communication replicating the way humans communicate among themselves.

Good design of personality, appearance and behavior for the Avatar is essential to catch the user's the attention making him/her believe that he/she is speaking to someone. Our future research lines will be specially centered on working on avatars for TV environments and Internet applications such as commerce and education, where having an Avatar is becoming crucial to create interaction between customers and salesmen, teachers and students, and among students. In teaching, avatars are also necessary to motivate the student, issue very often forgotten in distance education.

We are also interested, in parallel with the effort of other research groups, in multiple integration of avatars in "social" contexts and the animation of other parts of the body in Web applications. These two lines will be the relevant paths of our future work.

References

[1] Gurzki, T. "An architecture for integrating virtual sales assitant technology into e-business environment", E-business and eWork Conference, Italy (2001)

[2] Fashion Shopping with Individualized Avatars. Retrieved on March 2003; from: http://www.fashion-me.com/

[3] COGITO: E-Commerce with Guiding Agents based on Personalized Interaction Tools. Retrieved on March 2004; from: http://www.darmstadt.gmd.de/ cogito/Description.htm

[4] Kamyab, K., Guerin F., Goulev, P. and Mamdani, E. "Context Sensitive Virtual Sales Agents". In Proceedings of the Conference on Artistic, Cultural and Scientific Aspects of Experimental Media Spaces (cast0l), FhG - Institut Medienkommunikation (IMK), Germany, pp. 79-83. (2001)

[5] Vcom 3D, Inc. Retrieved on March 2004; from: http://www.vcom3d.com

[6] Eurocitizen: a web-based game of knowledge and collaboration for Europe. Retrieved on March 2004; from: http://eurocitizen.mls.gr

[7] Extempo. Retrieved on December 2004; from: http://www.extempo.com

[8] Oliveira, J.C., Shen, X. and Georganas, N.D. "Collaborative Virtual Environment for Industrial Training and e-Commerce", Proc. Workshop on Application of Virtual Reality Technologies for Future Telecommunication Systems, IEEE Globecom'2000 Conference, Nov.-Dec.2000 , San Francisco.

[9] University of Saldfor. The Centre for Virtual Environments. Retrieved on March 2004; from: http://www.nicve.salford.ac.uk/

[10] FNRS. 3D Behavioral Animation and Virtual Reality. Retrieved on March 2004; from: http://ligwww.epfl.ch/ thalmann/FNRS.html

[11] Marques Soares, J., Horain, P. and Bideau, A. "Sharing and immersing applications in a 3D virtual inhabited world", In Proc. 5th Laval Virtual Reality International Conference, France (2003) 27–31

[12] Gachery, S. and Magnenat-Thalmann, N. "Designing MPEG-4 facial animation tables for Web applications". In Proc. Multimedia Modeling, Amsterdam (2001) 39–59

[13] Berner, U. "Optimized face animation with morph-targets". Journal of WSCG, Vol.12, No 1-3 (2003)

[14] VHML Stardard. Retrieved in March 2004; from: www.vhml.org

[15] Müller, W., Spierling, U., Alexa, M. and Rieger, T. "Face-to-face with your assistant –realization issues of animated user interface agents for home appliances". In Proc. IMC 2000 - Workshop on Intelligent Assitance and MobileComputing, Germany (2000)

[16] Alexa M., Behr, J. and Müller, W. "The morph node". In Proc. Web3D/VRML 2000, Monterey CA (2000) 29–34

[17] Ortiz, A., and Posada, J.: "ABATEUS: Conversational 3D Avatar in Basque Language for Multimedia and TV Applications". TOPICS 4/2002 Vol. 14: Special Volume dedicated to VICOMTech of "TOPICS".

[18] Oyarzun, D., Ortiz, A., Aizpurura, I. and Posada, J. "Asistentes tridimensionales multilingües para entornos educacionales y de turismo virtual" [*3D multilingual assistants for learning environments and virtual tourism*]. JIS 2004, V Jornadas de Informática y Sociedad. Bilbao, Spain, March 2004

[19] ANFY 3D Home Page: Retrieved on March 2004; from: http://www.anfy3d.com

[20] MPEG-4. Signal Processing: Image Communication. Tutorial Issue on the MPEG-4 Standard, Vol. 15, Nos. 4-5, January 2000.

Improvement of Modal Matching Image Objects in Dynamic Pedobarography Using Optimization Techniques

Luísa Ferreira Bastos[1,2] and João Manuel R.S. Tavares[2]

[1] Laboratório de Óptica e Mecânica Experimental do Instituto de Engenharia Mecânica e Gestão Industrial,
[2] Departamento de Engenharia Mecânica e Gestão Industrial,
Rua Dr. Roberto Frias, s/n, 4200-465, Porto, Portugal
{lbastos,tavares}@fe.up.pt

Abstract. The paper presents an approach for matching objects in dynamic pedobarography image sequences, based on finite element modeling and modal analysis. The determination of correspondences between objects' nodes is here improved using optimization techniques and, because the elements number of each object is not necessarily the same, a new algorithm to match excess nodes is proposed. This new matching algorithm uses a neighborhood criterion and can overcome some disadvantages that the usual matching "one to one" in various applications can have. The proposed approach allows the determination of correspondences between 2D or 3D objects and will be here considered in dynamic pedobarography images.

1 Introduction

One problem of several areas of computational vision is the determination of objects correspondences in different images and on the computation of robust canonical descriptors used for the recognition of 2D or 3D objects, either rigid or non-rigid.

In this paper, a methodology to address the above problem is presented, based in the approach initially proposed by Sclaroff [1], [2], improved by us through the use of optimization algorithms in the matching search step, and a new method to determine matches between nodes that could not be considered in this optimization phase (see section 4). With this new method, we can successfully match objects with different number of points and also overcome some disadvantages that the usual match "one to one" can present in various real applications, because with that algorithm we can now have matches of type "one to many" and vice-versa.

Application results of the proposed matching methodology improved, and of the new "one to many" and vice-versa matching algorithm, to the analysis of objects represented in dynamic pedobarography image sequences are discussed.

1.1 Background

There is an eigen methods class [1], [2], that derives its parameterization directly from the data shape. Some of these techniques also try to determine, explicitly and automatically, the correspondences between characteristic points' sets, while others avoid specifying correspondences based in characteristics, but try to match images using more global approaches. Each eigen method decomposes the object deformation in an orthogonal and ordered base.

Usually, the matching problem resolution methods include restrictions that prevent inadequate matches according to the considered criteria. Examples of these restrictions are: the order [3], [4]; rigidity restriction [3], [4]; unicity [5]; visibility [6]; and proximity [2]. To determine the correspondences it is used, for example, the correlation between images (i.e., images similarity is assumed) [7]; points proximity [8]; or disparity fields smoothness [3]. The matching problem can be interpreted as an optimization problem, where the objective function can depend, for example, on any relation mentioned above and the restrictions must form a non-empty space of possible solutions for the optimization problem. To solve the optimization problem it can be used dynamic programming [3], graphs [4] and convex minimization [7]. Non-optimal approaches include, for example greedy algorithms [9], simulated annealing [10], relaxation [5], etc.

Belongie [11] applied an optimization technique, similar to the one used in this work, to determine the correspondences between two objects using shape contexts. Although shape description algorithms have usually a higher computational efficiency, the modeling methodology presented here has the major advantage of attributing a physical behavior to each object to be modeled through the consideration of a virtual material.

2 Dynamic Pedobarography

Pedobarography refers to measuring and visualizing the distribution of pressure under the foot sole. The recording of pedobarographic data along the duration of a step in normal walking conditions permits the dynamic analysis of the foot behavior; the introduction of the time dimension augments the potential of this type of clinical examination as an auxiliary tool for diagnostics and therapy planning [12], [13].

The basic pedobarography system consists of glass or acrylic plate trans-illuminated through its polished borders in such a way that the light is internally reflected; the plate is covered on its top by a single or dual thin layer of soft porous plastic material where the pressure is applied (see Fig. 1) [13].

Using a practical set-up as the one shown in Fig. 2, a time sequence of pressure images is captured; Fig. 3 shows two of the images captured in a sample sequence (displayed with inverted brightness); the image data is very dense, as opposed to other used measuring methods, and very rich in terms of the information it conveys on the interaction between the foot sole and the flat plate [13].

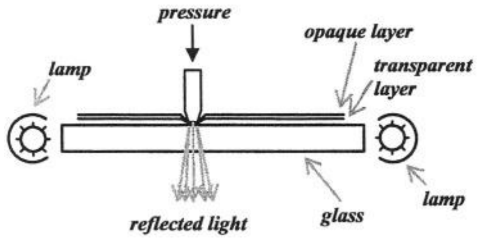


Fig. 1. Basic (pedo)barography principle [13]

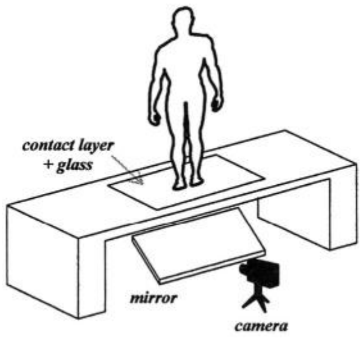


Fig. 2. Set-up of a basic pedobarography system [13]

Fig. 3. Two consecutive images of a sample pedobarography images sequence

3 Objects Model

In the initial stages of this work [2], [13], the object contours in each image were extracted and the matching process was oriented to the contours pixels. However, an additional problem is present: the possibility that along the images sequence various contours will merge or split. In order to accommodate this possibility a new model was developed [13], similar to the one used in various applications with controlled environment, such as in face analysis and recognition [14], [15]. The brightness level of each pixel is considered as the third co-ordinate of a 3D surface point. The resulting single surface model solves the two aforementioned problems [13].

The use of the surface model also simplifies the consideration of isobaric contours, which are important in pedobarographic analysis, either for matching contours of equal pressure along the time sequence or for matching contours of different pressure in a single image [13].

The following sections describe the object models used and their construction. Each model has its own advantages and shortcomings; for every particular problem, the best choice must be made [13].

3.1 Contour Model

Two modeling approaches were used to determine the correspondence between two contours in distinct images:

- A single 2D isoparametric Sclaroff model is used for each contour. In building this type of element, no previous ordering of the nodes is required; Gaussian shape functions are used. The method to determine the mass and stiffness matrices for this 2D element is described, for example, in [1], [2].
- Each contour is built by linear axial 2D finite elements. For this type of discretization a previous ordering of the contour nodes is required. The matrix formulation for these elements can be found in [2], [16], for example.

Standard image processing and analysis techniques are used to determine the contour pixels [2], namely thresholding, edge enhancement, hysteresis line detection and tracking. For example, Fig 4(b) and Fig. 4.(c) show an intermediate result and the final contour determination for the image in Fig. 4(a).

3.2 Surface Model

For the surface model, two approaches were also used [2], [13]:

- A single 3D isoparametric Sclaroff finite element model is used for each surface. Again, it must be noticed that there is no requirement for previous ordering of the nodes. The matrix building for these finite elements can be found in [1], [2].
- Each surface is built by linear axial 3D finite elements (Fig. 5). The previous ordering of the surface nodes is required. The matrix formulation for these finite elements can be found in [2], [16], for example.

The method to determine the nodes that form the surface in each image can be summarized as follows [2], [13]:

1. Noise pixels (i.e., pixels with brightness lower than a calibration threshold) are removed and a Gaussian-shaped smoothing filter is applied to the image (Fig. 6(a));
2. The circumscribing rectangle of the object to be modeled is determined and the image is sampled within that area (Fig. 6(b));
3. A 2D Delaunay triangulation is performed on the sampled points, using the point brightness as the third co-ordinate;
4. The triangular mesh is simplified using a decimation algorithm in order to reduce the number of nodes and thus the computational cost;
5. A Laplacian smoothing algorithm is used to reduce the high frequency noise associated to the mesh;
6. A scale change is performed on the third co-ordinate (derived from brightness) in order to have similar ranges of values in all co-ordinates (Fig. 6(c)).

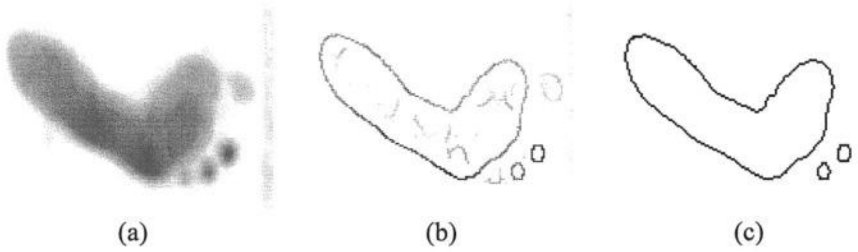


Fig. 4. (a) Image (negated) where contours must be found; (b) Result image after edge enhancement; (c) Contours obtained by a line detection and tracking algorithm with hysteresis

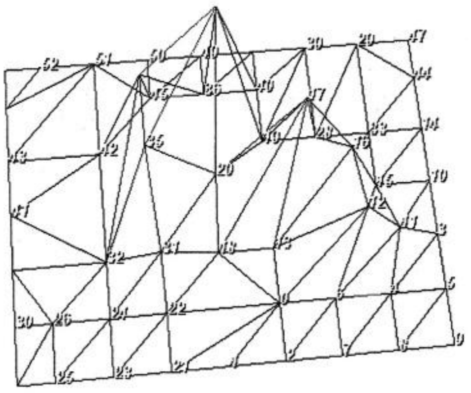


Fig. 5. Modeling of a surface by a set of axial 3D finite elements (each node is connected to its neighbors through axial elements)

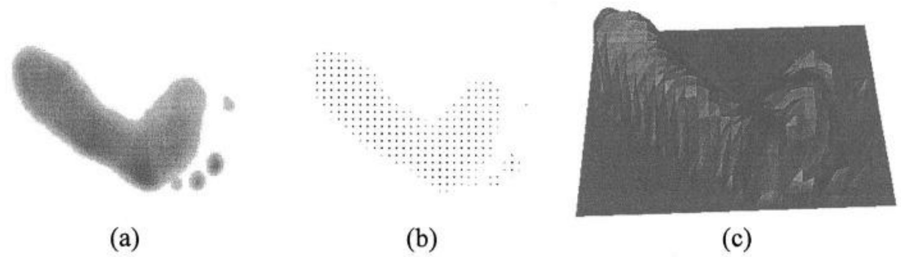


Fig. 6. (a) Image (negated) after noise removal and Gaussian filtering; (b) Object sampling; (c) Resulting surface

3.3 Isobaric Contour Model

As in the two previous models, the approaches used to match isobaric contours and to determine the deformation energy are [2], [13]:

- A single Sclaroff isoparametric finite element, either 2D or 3D, to model each contour.
- Linear axial finite elements, either 2D or 3D, to build the contours.

The isobaric contours are extracted (Fig. 7.) after using the procedure described in the previous section.

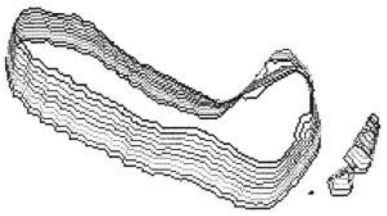


Fig. 7. Ten isobaric contours extracted from the surface in Fig. 6.(c)

4 Matching Methodology

Fig. 8 displays a diagram of the adopted matching methodology. The locations of the image data points $X = [X_1 \dots X_m]$ in each image are used as the nodes for building a finite element model of elastic material. Next, the eigenmodes $\{\phi\}_i$ of the model are computed, providing an orthogonal description of the object and its natural deformations, ordered by frequency. Using a matrix based notation, the eigenvectors matrix $[\Phi]$ and the eigenvalues diagonal matrix $[\Omega]$ can be written as in eq. (1) for 2D objects and as in eq. (2) for 3D objects. The eigenvectors, also called shape vectors, for each mode [1], [2], [16], [17], describe how each mode deforms the object by changing the original data point locations: $X_{deformed} = X + a\{\phi\}_i$.

The first three (in 2D) or six (in 3D) modes are the rigid body modes of translation and rotation; the remaining modes are non-rigid [1], [2], [16], [17]. In general, lower frequency modes describe global deformations while higher frequency modes essentially describe local deformations. This type of ordering from global to local behaviour is quite useful for object matching and comparison.

The eigenmodes also form an orthogonal, object-centred co-ordinate system for the location of the point data, i.e., the location of each point is uniquely described, in terms of each eigenmode displacement. The transformation between the Cartesian image co-ordinates and the modal system co-ordinates is achieved through the eigenvectors of the finite element model.

Two sets of image data points, corresponding to two different images in a sequence, are to be compared in the modal eigenspace. The main idea is that the low order modes of two similar objects will be very close even in the presence of an affine transformation, a non-rigid deformation, a local shape variation, or noise [1], [2].

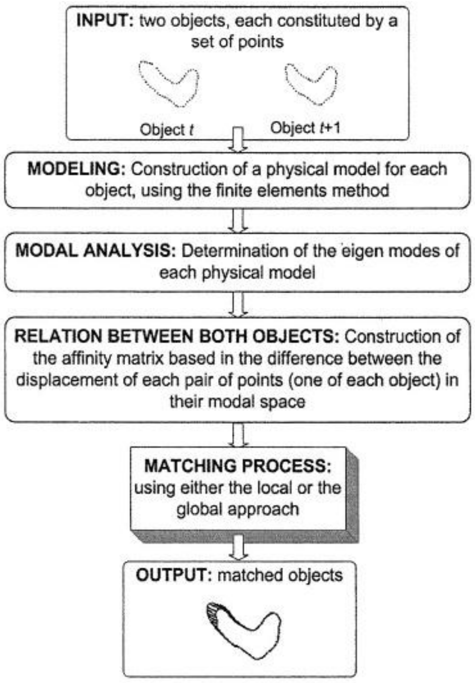

INPUT: two objects, each constituted by a set of points
Object t
Object t+1
MODELING: Construction of a physical model for each object, using the finite elements method
MODAL ANALYSIS: Determination of the eigen modes of each physical model
RELATION BETWEEN BOTH OBJECTS: Construction of the affinity matrix based in the difference between the displacement of each pair of points (one of each object) in their modal space
MATCHING PROCESS: using either the local or the global approach
OUTPUT: matched objects

Fig. 8. Diagram of the adopted methodology

$$[\Phi] = \left[\{\phi\}_1 \middle| \cdots \middle| \{\phi\}_{2m}\right] = \begin{bmatrix} \{u\}_1^T \\ \vdots \\ \{u\}_m^T \\ \{v\}_1^T \\ \vdots \\ \{v\}_m^T \end{bmatrix} \text{ and } [\Omega] = \begin{bmatrix} \omega_1^2 & & 0 \\ & \ddots & \\ 0 & & \omega_{2m}^2 \end{bmatrix}, \tag{1}$$

$$[\Phi] = \left[\{\phi\}_1 \middle| \cdots \middle| \{\phi\}_{3m}\right] = \begin{bmatrix} \{u\}_1^T \\ \vdots \\ \{u\}_m^T \\ \{v\}_1^T \\ \vdots \\ \{v\}_m^T \\ \{w\}_1^T \\ \vdots \\ \{w\}_m^T \end{bmatrix} \text{ and } [\Omega] = \begin{bmatrix} \omega_1^2 & & 0 \\ & \ddots & \\ 0 & & \omega_{3m}^2 \end{bmatrix}. \tag{2}$$

Using the above concept, data correspondence is obtained by modal matching. Two matching search methods are considered: (1) a local search or (2) a global search. Both methods use an affinity matrix [Z], constructed from the Euclidian distance between the characteristic vectors of each model, whose elements are, for 2D and for 3D, respectively:

$$Z_{ij} = \left\| \{u\}_{t,i} - \{u\}_{t+1,j} \right\|^2 + \left\| \{v\}_{t,i} - \{v\}_{t+1,j} \right\|^2 , \tag{3}$$

$$Z_{ij} = \left\| \{u\}_{t,i} - \{u\}_{t+1,j} \right\|^2 + \left\| \{v\}_{t,i} - \{v\}_{t+1,j} \right\|^2 + \left\| \{w\}_{t,i} - \{w\}_{t+1,j} \right\|^2 . \tag{4}$$

The local search was previously implemented [1], [2], [13] and basically it consists in seeking each row of the affinity matrix for its lowest value, considering the associated correspondence if that value is also the lowest of the associated column. This method has the main disadvantage of disregarding the object structure as it searches for the best match for each node.

The global search, proposed in this work, consists in describing the matching problem as an assignment problem [18], [19], and solving this last using an appropriate algorithm [20], [21], [22].

Case the number of nodes of both objects is not the same, with the usual matching restriction of type "one to one", there will be, necessarily, nodes that will not be matched. The solution found [20] was initially to add fictitious nodes to the model with fewer points, solving the requirement of the matrix involved in the optimization process (assignment problem) having, necessarily, to be square. All fictitious nodes have a zero matching cost associated. After the global search, the excess nodes (real nodes matched with fictitious points) are matched adequately (with real nodes), using a neighborhood criterion. In this way matches of type "one to many" or vice versa are allowed for the "extra" nodes. We will call this method *ADCom.*

5 Results

The methodology just presented was integrated in a generic software platform for deformable objects matching [23], previous developed using Microsoft Visual C^{++}, the Newmat [24] library for matrix computation and the VTK - The Visualization Toolkit [25], [26] for 3D visualization, mesh triangulation, simplification and smoothing, and for isobaric contours extraction.

This section presents some obtained results with dynamic pedobarography images, using the methodology previously described.

The first example presents the matching of two contours obtained from the sequence images previously presented in Fig. 3. The first contour has 64 points and the second one has 58 points. Fig. 9(a) shows the obtained match using the local search adopted in the previous work and Fig. 9(b) shows the one obtained using the proposed global search. Table 1 presents some numerical results for this example. Using the proposed global search 100% of matches of type "one to one" was successfully obtained. Using the local search was only obtained 72% of those matches.

Table 1. Numeric results of the matching between the dynamic pedobarography contours

Search method	Nr. Matches (%)	Fig.
Local	42 (72%)	9 (a)
Global	58 (100%)	9 (b)

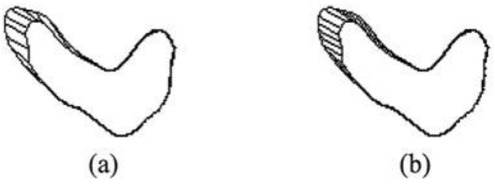


Fig. 9. Matches between two contours using: (a) the local search; (b) the global search

The second example presents the matching between two isobaric contours, obtained from the previously presented image in Fig. 3.(b). The first isobaric contour is composed by 54 points and the second one by 46 points. Fig. 10(a) shows the matching result using the local search and Fig. 10(b) shows the matching result using the proposed global search. Table 2 presents some numerical results for this example. Using the proposed global search we obtained 100% of type "one to one" satisfactory matches. Using the local search we only obtained 50% of those matches.

Table 2. Numeric results of the matching between two isobaric contours

Search method	Nr. Matches (%)	Fig.
Local	23 (50%)	10 (a)
Global	46 (100%)	10 (b)

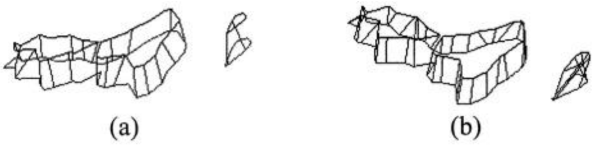


Fig. 10. Matches between two isobaric contours using: (a) the local search; (b) the global search

Fig. 1.1 presents the matching results for the first and second examples using the proposed global matching search and the developed *ADCom* algorithm to match the excess nodes. As it can be seen, all objects' nodes were successfully matched for both examples.

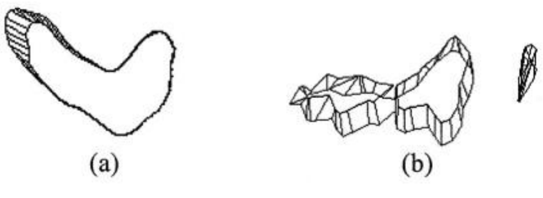


Fig. 11. (a) Matches between two contours using the global search and *ADCom* algorithm; (b) Matches between two isobaric contours using the global search and *ADCom* algorithm

The last example presents the matching of two surfaces obtained from the same images presented in Fig. 3. The first surface, obtained from the image in Fig. 3(a), has 117 points and the second one, obtained from the image in Fig. 3(b), has 112 points. Fig. 12(a) shows the matching result using the local search and Fig. 12(b) shows the matching result using the proposed global search. Table 3 presents some numerical results of this last example. Using the global search 100% of satisfactory matches of type "one to one" was established. Using the local search was only established 31% of those matches.

6 Conclusions

A methodology was presented for obtaining the correspondence between 2D and 3D objects, rigid or non-rigid, and it was illustrated for real dynamic pedobarography images. The objects are modelled by finite elements and modal analysis is used to define an eigenspace where the matching is performed.

The experimental tests carried out (some shown in this paper) confirm that satisfactory matching results can be obtained for dynamic pedobarographic image data.

Table 3. Numeric results of the matching between two dynamic pedobarography surfaces

Search method	Nr. Matches (%)	Fig.
Local	35 (31%)	12 (a)
Global	112 (100%)	12 (b)

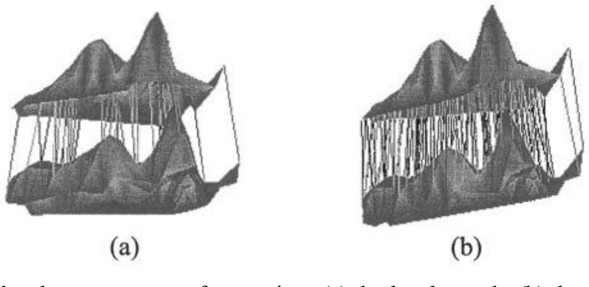


Fig. 12. Matches between two surfaces using: (a) the local search; (b) the global search

It is also verified that the proposed global search considerably improves these results, when compared with the previously local search used in [2], [13], in the obtained number and in the quality of the founded matches.

Another advantage of the proposed search strategy, based on optimization techniques, is that the overall matching methodology becomes easier to use and to adapt to experimental applications.

The developed *ADCom* algorithm satisfactorily matches all the excess objects nodes, avoiding loss of information along image sequences.

In the future, we will try to improve the matching methodology just presented in order to considerer all the images sequence and use this along time information in the nodes correspondence establishment process.

References

[1] S. E. Sclaroff and A. Pentland, "Modal Matching for Correspondence and Recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 17, 1995

[2] 2. J. M. R. S. Tavares, "Análise de Movimento de Corpos Deformáveis usando Visão Computacional", in *Faculdade de Engenharia da Universidade do Porto,* Portugal, 2000

[3] 3. Y. Ohta and T. Kanade, "Stereo by Intra- and Inter-Scanline Search using Dynamic Programming," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 7, pp. 139-155, 1985

[4] 4. S. Roy and I. J. Cox, "A Maximum-Flow Formulation of the N-camera Stereo Correspondence Problem," presented at International Conference on Computer Vision (ICCV'98), Bombay, India, 1998

[5] 5. S. Gold, A. Rangarajan, C. P. La, S. Pappu, and E. Mjolsness, "New algorithms for 2D and 3D point matching: pose estimation and correspondence," Pattern Recognition, vol. 31, pp. 1019-1031, 1998

[6] C. Silva, "3D Motion and Dense Structure Estimation: Representation for Visual Perception and the Interpretation of Occlusions," in Instituto Superior Técnico: Universidade Técnica de Lisboa, 2001

[7] J. L. Maciel and J. P. Costeira, "A Global Solution to Sparse Correspondence Problems," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, pp. 187-199, 2003

[8] Z. Zhang, "Iterative Point Matching for Registration of Free-Form Curves," INRIA, Technical Report RR-1658, April 1992

[9] M. S. Wu and J. J. Leou, "A Bipartite Matching Approach to Feature Correspondence in Stereo Vision," Pattern Recognition Letters, vol. 16, pp. 23-31, 1995

[10] J. P. P. Starink and E. Backer, "Finding Point Correspondences using Simulated Annealing," Pattern Recognition, vol. 28, pp. 231-240, 1995

[11] S. Belongie, J. Malik and J. Puzicha, "Shape Matching and Object Recognition using Shape Context," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, pp. 509-522, 2002

[12] A. J. Padilha, L. A. Serra, S. A. Pires, and A. F. N. Silva, "Caracterização Espacio-Temporal de Pressões Plantares em Pedobarografia Dinâmica", FEUP/INEB 1995

[13] J. M. R. S. Tavares, J. Barbosa, and A. Padilha, "Matching Image Objects in Dynamic Pedobarography", presented at *11th Portuguese Conference on Pattern Recognition (RecPad'00),* Porto, Portugal, 2000

[14] T. F. Cootes, C. J. Taylor, "Modelling Object Appearence Using The Grey-Level Surface", presented at *British Machine Vision Coference,* 1994

[15] B. Moghaddam, C. Nastar, A. P. Pentland, "Bayesian Face Recognition using Deformable Intensity Surfaces", MIT Media Laboratory - Technical Report N° 371, 1996

[16] K.-J. Bathe, *Finite Element Procedures,* Prentice Hall, 1996

[17] S. Graham Kelly, *Fundamentals of Mechanical Vibrations,* McGraw-Hill, 1993

[18] L. F. Bastos and J. M. R. S. Tavares, "Optimization in Modal Matching for Correspondence of Objects Nodal Points", presented at *7th Portuguese Conference on Biomedical Engineering (BioEng'2003),* Fundação Calouste Gulbenkian, Lisboa, Portugal, 2003

[19] F. S. Hillier and G. J. Lieberman, *Introduction to Operations Research.* Mcgraw-Hill International Editions, 1995

[20] L. F. Bastos, “Optimização da Determinção das Correspondências entre Objectos Deformáveis no Espaço Modal”, in *Faculdades de Engenharia e Ciências:* Universidade do Porto, Portugal, 2003

[21] A. Löbel, “MFC - A Network Simplex Implementation”: Konrad-Zuse-Zentrum fur Informationstechnik Berlin, Division Scientific Computing, Department Optimization, 2000

[22] A. Volgenant, “Linear and Semi-Assignment Problems: A Core Oriented Approach”, *Computers and Operations Research,* vol. 23,1996

[23] J. M. R. S. Tavares, J. Barbosa, and A. Padilha, “Apresentação de um Banco de Desenvolvimento e Ensaio para Objectos Deformáveis,” *Revista Electrónica de Sistemas de Informação,* vol. 1, 2002

[24] R. Davies, *Newmat, A matrix library in* C^{++}, 2003

[25] W. Schroeder and K. Martin, *The VTK User’s Guide,* Kitware Inc., 2003

[26] W. Schroeder and K. Martin, B. Lorensen, *The Visualization Toolkit,* 3nd Edition, Kitware Inc., 2002

Ear Biometrics Based on Geometrical Method of Feature Extraction

Michał Choraś

Institute of Telecommunication, University of Technology and Agriculture,
ul. Prof. Kaliskiego 7, 85-796, Bydgoszcz, Poland.
chorasm@mail.atr.bydgoszcz.pl

Abstract. Biometrics identification methods proved to be very efficient, more natural and easy for users than traditional methods of human identification. In fact, only biometrics methods truly identify humans, not keys and cards they posses or passwords they should remember. The future of biometrics leads to passive physiological methods based on images of such parts of human body as face and ear. The article introduces to ear biometrics and presents its advantages over face biometrics in passive human identification systems. Then the geometrical method of feature extraction from human ear images in order to perform human identification is presented. The proposed method is invariant to rotation, translation and scaling due to coordinates normalization and placing the major reference point in the centroid. The feature extraction algorithm consists of two steps, so that in the process of classification two feature vectors for each ear image are used.

1 Introduction

There are many known methods of human identification based on image analysis. In general, those biometrics methods can be divided into behavioural and physiological regarding the source of data, and can be divided into passive and invasive biometrics, regarding the way the data is acquired.

The first class is based on the behavioural features of human actions and it identifies people by how they perform something. The most popular of such methods is voice verification. Other methods are basically based on the dynamics of specific actions like making the signature, typing on the keyboard and simply moving or walking. Those methods are not that natural and they require users to take part in the process of identification by repeating specific actions, every time they are examined. Physiological (anatomical) biometrics methods are based on the physiological features of humans thus they measure and compare features of specific parts of human body in the process of identification. So far the main interest is in the head and the hand with face, eye and fingerprint features being the most important discriminants of human identity. The major advantage of physiological biometrics is that it is passive and the implemented systems work only with the acquired images of specific body parts. All the user has to do is to place his face or ear in front of the camera or alternatively touch the sensor with his fingers and wait for the identification process to take place. But some systems can verify the identity of

humans even without their cooperation and knowledge, which is actually the future of biometrics. Such methods include well-examined face recognition but one of the most interesting novel approaches to human passive identification is the use of ear as the source of data. The article presents some advantages of ear biometrics over face in human identification.
Human ears have been used as major feature in the forensic science for many years. Recently so called earprints, found on the crime scene, have been used as a proof in over few hundreds cases in the Netherlands and the United States [9].

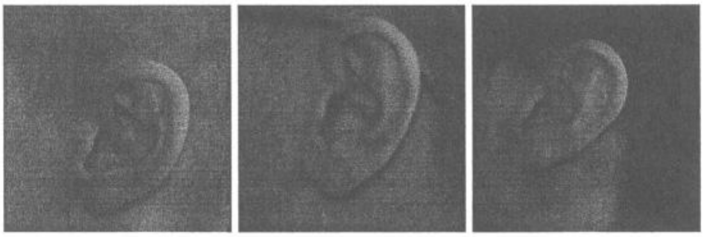


Fig. 1. Some examples of "easy" ear images from our database

There are many advantages of using ear in comparison with the face. Firstly, ear does not change during human life, and face changes more significantly with age than any other part of human body. Face can also change due to cosmetics, facial hair and hair styling. Secondly, face changes due to emotions and expresses different states of mind like sadness, happiness, fear or surprise. In contrast, ear features are fixed and unchangeable. Moreover, the colour distribution is more uniform in ear than in human face, iris or retina. Thanks to that fact, not much information is lost while working with the greyscale or binarized images, as we do in our method. Ear is also smaller than face, which means that it is possible to work faster and more efficiently with the images with the lower resolution. Ear images cannot be disturbed by glasses, beard nor make-up. However, occlusion by hair or earrings is possible. Some examples of "easy" and "difficult" ear images from our database are shown in Figs 1 and 2, respectively.

The first, manual method, used by Iannarelli in the research in which he examined over 10000 ears and proved their uniqueness, was based on measuring the distances between specific points of the ear [11]. The major problem in ear identification systems is discovering automated method to extract those specific, key points. Another well-known method by Burge and Burger [4] is based on building neighbourhood graph from Voronoi diagrams of the detected edges. Hurley et al. [10] introduced a method based on energy features of the image. They proposed to perform force field transformation in order to find energy lines, wells and channels, but their feature vector consisted of only two parameters. Another method used by Victor at al. [17], in the experiment comparing ear and face properties in order to successfully identify humans in various conditions, was based on PCA. Their work proved that ear images are a very suitable source of data for identification and their results for ear images were not significantly different than for face images. The method, however, was not fully automated, since the reference points had to be manually inserted into images.

Fig. 2. Some examples of “difficult” ear images from our database

We propose a straightforward method to extract features needed to classification. Our method represents the geometrical approach, but it is fully automated and no manual operations are needed.

Our method is divided into image normalization, contour extraction (edge detection), calculation of the centroid, coordinates normalization and 2 steps of geometrical feature extraction, as described in the next section. We treat the centroid as the specific point in our method, even though it is not a specific point within the ear topology. Our method consists of the following steps (Fig. 3.):

- contour detection,
- binarization,
- coordinates normalization,
- feature extraction (2 steps),
- classification.

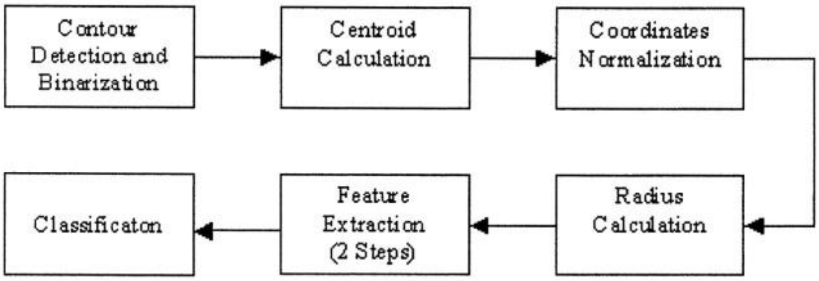

Contour Detection and Binarization
Centroid Calculation
Coordinates Normalization
Radius Calculation
Feature Extraction (2 Steps)
Classificaton

Fig. 3. The block diagram of our method

2 Contour Detection

First we perform the edge detection step. In our case it is a crucial operation, since it is obvious that lines are the most prominent features that could be obtained from the ear image, and our goal is to detect major outer and inner curves of the earlobe. We tested many known methods as Canny operator, Sobel filters and CWT (Complex Wavelets) but we propose another method, which proved to be most convenient in our experiments.

We propose to use the local method which examines illumination changes within the chosen window $n \times n$. We usually use 3×3 window and we divided the image into many overlapping regions of that size.

For each of those regions we calculated mean μ and standard deviation σ of the pixel intensity values in 8-pixel neighbourhood.

$$\mu = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} I(i,j) \tag{1}$$

$$\sigma = \sqrt{\frac{1}{(n^2-1)} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(I(i,j)-\mu\right)^2} \tag{2}$$

Then we perform decision if the centre pixel of the examined region belongs to the line or to the background. For the maximum value of the pixel intensity I_H, and the minimum value of the pixel intensity in the region I_L, we calculate the difference S(i,j) such as:

$$S(i,j) = I_H - I_L \tag{3}$$

and we compare it to certain threshold value. Even though thresholding is one of the basic operations of image processing, there is always the major problem in selecting appropriate threshold value.

We propose the usage of mean and standard deviation of pixel intensities in calculation of the threshold value $T(i,j)$ used in contour detection as given in equation 4:

$$T(i,j) = \mu - k\sigma \tag{4}$$

where *k* is a certain value.

Then the rule for the contour detection is:

$$g(i,j) = \begin{cases} 1 & \text{if } S(i,j) \geq T(i,j) \\ 0 & \text{if } S(i,j) < T(i,j) \end{cases} \tag{5}$$

In result we obtain the binary image $g(i,j)$ with the detected contours. Moreover, the constant *k* allows to adjust and change the sensivity of the edge detection algorithm.

An example of the edge detection algorithm is shown in the Fig. 4.

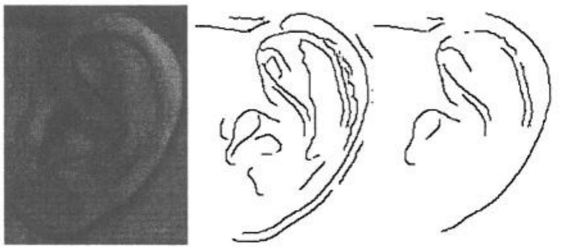


Fig. 4. Result of edge detection algorithm for two different values of *k*

3 Normalization

Given the binary image g(i, j), we obtain the centroid which later becomes the reference point for feature extraction. Because the features for a recognition algorithm should be invariant to ear translation and scale change, the coordinates normalization is performed. Therefore we normalize coordinates, such that the centroid becomes the centre of the image. Suppose that the image with pixel coordinates (i, j) undergoes geometric transformations to produce an invariant image with coordinates (x, y). This transformation may be expressed as:

$$[x, y, z] = [i, j, 1]\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -I & -J & 1 \end{bmatrix}\begin{bmatrix} \frac{1}{\sigma_i} & 0 & 0 \\ 0 & \frac{1}{\sigma_j} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

where:

- I, J – centroid,
- σ_i, σ_j – **standard** deviation of *i* and *j* respectively.

Furthermore, our method is also invariant to rotation as all the rotated images of the same object have the same centroid. That is the major reason that we chose the centroid of the image to be the reference point in the feature extraction algorithm. Such approach allows the successful processing ofRST queries.

4 Feature Extraction

There are many possible geometrical methods of feature extraction and shape description such as Fourier Descriptors, Delaunay Triangles and methods based on combination of angles and distances as parameters. We propose a 2 step-method that is based on number of pixels that have the same radius in a circle with the centre in the centroid and on the contour topology. The algorithm for the first step of feature extraction is presented below:

1. we create a set of circles with the centre in the centroid (Fig. 5.)
2. number of circles N_r is fixed and unchangeable
3. we create circles in such a manner that the corresponding radiuses are α pixels longer from the previous radius
4. since each circle is crossed by the contour image pixels we count the number of intersection pixels l_r
5. next we calculate all the distances d between neighbouring pixels, we proceed in the counter-clockwise direction
6. we build the feature vector that consists of all the radiuses with the corresponding number of pixels belonging to each radius and with the sum of all the distances between those pixels $\sum d$

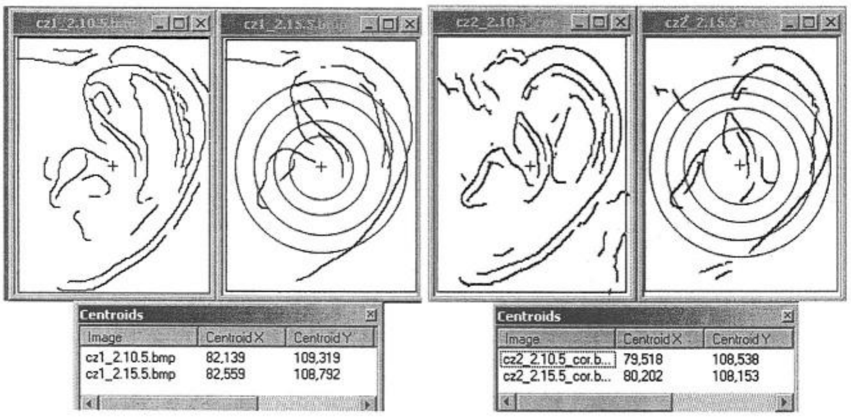


Image	Centroid X	Centroid Y
cz1_2.10.5.bmp	82.139	109.319
cz1_2.15.5.bmp	82.559	108.792

Image	Centroid X	Centroid Y
cz2_2.10.5_cor.b...	79.518	108.538
cz2_2.15.5_cor.b...	80.202	108.153

Fig. 5. Binary ear images with the extracted edges (2 values of *k*) and with the centroid marked with a cross. Circles represent the radius values for calculation of number of pixels intersecting each circle. The table below shows the centroid values for each binary image

The algorithm for *Nr=3* is symbolically presented in the Fig. 6.
The general rule for forming the first vector is presented below:

$$V = \left\{ \left[r_{min}, l_{r\,min}, \sum d_{r\,min} \right] \cdots \left[r_{max}, l_{r\,max}, \sum d_{r\,max} \right] \right\} \tag{7}$$

where:
r – radius length,
l_r – number of intersection points for each radius,
$\sum d$ – sum of all the distances between the intersection points for the considered radius.

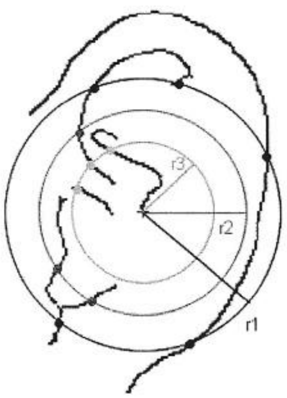


Fig. 6. The symbolic representation of our algorithm for $N_r = 3$

Then in order to enhance the distinctiveness of the extracted features, we build the second vector in the second step of feature extraction. Once again we base upon the created circles with the centre in the centroid. Hereby, we propose to extract the characteristic points for each contour in the normalized coordinates.
For each contour line the characteristic points are:

- contour endings,
- contour bifurcations,
- all the points that cross the created circles (those points are already extracted by the previous algorithm).

In each contour we check the topological properties of every pixel. For each contour pixel g_o we use 3×3 window as in Fig. 7. (left). When $g_0 = 1$, the connected number N_c^8 of g_0 is defined as:

$$N_c^8(g_o) = \sum_{k=S} \left(\bar{g}_k - \bar{g}_k \, \bar{g}_{k+1} \, \bar{g}_{k+2} \right), \text{ where } S = (1,3,5,7). \tag{8}$$

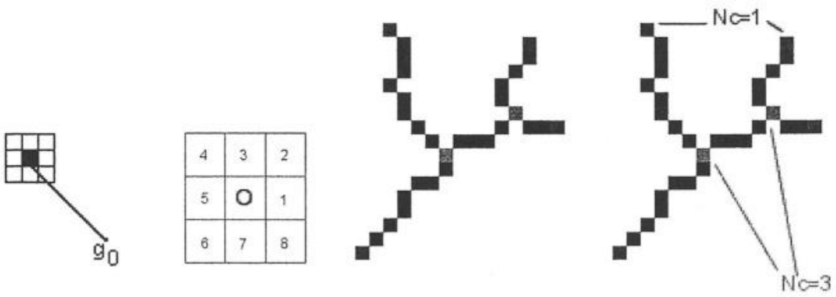


Fig. 7. Characteristic points detection algorithm

We search for the contour beginning in the area A, such that: $r_i \geq A > r_{i-1}$. We begin the search for $r_i = r_{max}$, which means that we start our algorithm in the most outer circle. Then we search in all other circles heading towards the centre in the centroid. If we come across any point with $N_c^8 = 1$, we check if it is already stored in the feature vector and if not, we store it as the ending point and we trace its contour. Points with $N_c^8 = 1$ and $N_c^8 > 2$ are the ear contour endings and the contour bifurcation points respectively. Those points are marked as E and B in the Fig. 8. For each contour we also extract the intersections with the circles created earlier. For each contour intersecting the circles we store all the intersections coordinates i and the number of such intersections N_I as presented in Fig. 8 (right) and Eq. 9

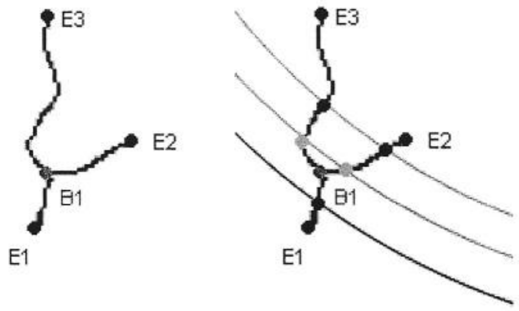


Fig. 8. The symbolic representation of the second step of feature extraction algorithm

The general rule for forming the second vector for each contour is presented below. First we store the number of endings, bifurcations and intersection points, and then we store all the coordinates of those points for all the extracted and traced contours. For C contours in a given image we obtain:

$$F = \{\left[(N_E, N_B, N_I)(e_1, \cdots, e_{N_E}, b_1, \cdots, b_{N_B}, i_1, \cdots, i_{N_I})\right]_1 \cdots$$
$$\cdots\left[(N_E, N_B, N_I)(e_1, \cdots, e_{N_E}, b_1, \cdots, b_{N_B}, i_1, \cdots, i_{N_I})\right]_C\} \quad (9)$$

where:

N_E – number of endings in each contour,

N_B – number of bifurcations in each contour,

N_I – number of points intersecting the circles,

e – coordinates of endings,

b – coordinates of bifurcations,

i – coordinates of intersections in each contour.

5 Classification

For each image stored in the database we have two vectors F_{ref} and V_{ref}. For each input ear, we acquire many images under different angles to the camera.
The algorithm for recognition of an input image is following:

1. for the fixed number of circles, the feature vectors V and F of the input image are obtained
2. for each radius, we search the database feature vectors V_{ref} that have the same number of intersections l_r for the corresponding radiuses
3. the vectors with number of intersections $(l_r \pm 1)$ are also accepted, allowing the difference of one pixel on each circle
4. in the next step we check if the difference within the distance sum $\sum d$ for all the extracted vectors is less than a certain threshold value
5. if none of the vectors V_{ref} are found for the input image, the input image is rejected
6. if the number of intersecting points l_r is accepted and the difference within the distance sum $\sum d$ is less than a certain threshold value we check the contour-topology vector F
7. we first search for the same triples (N_E, N_B, N_I) of the input contour-topology vector F with the reference contour vectors F_{ref}
8. then for the images with the same triples (N_E, N_B, N_I) we check if the coordinates of the stored points are the same
9. if the corresponding coordinates of those vectors refer to the same points , the algorithm finds the winner of classification.

6 Experimental Results and Future Work

We perform our experiments on our own database of collected ear images. At the moment of writing, the database consists of over 240 images, but we are still adding more ear images of different type. For each person included in the experiments, we collected 2 left ear images, first with the camera perpendicular to the head and the second, with the camera within the specified angle of 30 degrees. We divided the database to several sets of images concerning their quality and degree of complexity. So far we have only experimented with images of very high quality and with the ideal conditions of recognition, without illumination changes. For such "easy" images from our database we obtained error-free recognition.
In further experiments we work with the "difficult" images and with the changing conditions of the image acquisition. In order to achieve satisfactory results with such complex images (Fig. 2) we are improving the contour detection algorithm, so that long, straight line-contours of glasses and artificial contours of earrings and hair are eliminated before applying feature extraction algorithm. We also think that the feature

vectors should be enriched with more geometrical features in order to better distinguish ear identity. Moreover, we search for other than geometrical features describing ear images, such as energy and shape parameters. We try to discover, which features are the most significant in determining ear identity, so that we will be able to weight them properly in the process of building hybrid vectors of features of different type.

7 Conclusions

In the article we proposed a human identification method based on human ear images. We believe that human ear is unique and has many advantages over other biometrics methods. We proposed invariant geometrical method in order to extract features needed to classification. First we perform contour detection algorithm, then size normalization. Thanks to placing the centre of the new coordinates system in the centroid, our method is invariant to rotation, translation and scaling, which will allow RST queries. The centroid is also a key reference point in the feature extraction algorithm, which is divided into 2 steps. In the first step, we create circles centred in the centroid and we count the number of intersection points for each radius and the sum of all the distances between those points. All those points are stored in the first feature vector corresponding to the radiuses. In the second step, we use the created circles, but hereby we count the intersection points for each contour line. Moreover, while tracing the contour lines, we detect the characteristic points like endings and bifurcations. Together with the intersection points for each contour, we store them in the second feature vector corresponding to contour topology. Then we perform classification, basing on the simple comparison between the input image feature vectors, and all the vectors from the database. So far we have obtained very good results, however we still continue our research in order to improve our method and add more parameters to the feature vectors.

References

[1] Ashbourn J., Biometrics - Advanced Identity Verification, Springer-Verlag 2000.
[2] Beveridge J.R., She R., Draper B.A., Givens G.H., Parametric and Nonparametric Methods for the Statistical Evaluation of Human Id Algorithms, Workshop on Evaluation Methods in Computer Vision, 2001.
[3] Bowman E., Everything You Need to Know about Biometrics, Technical Report, Identix Corporation, 2000.
[4] Burge M., Burger W., Ear Biometrics, Johannes Kepler University, Linz, Austria 1999.
[5] Burge M., Burger W., Ear Biometrics for Machine Vision, 21 Workshop of the Austrian Association for Pattern Recognition, Hallstatt, 1997.
[6] Canny J., A Computational Approach to Edge Detection, IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 8, no. 6, 679-698, 1986.
[7] Choras M., Human Identification Based on Image Analysis – New Trends, Proc. Int. IEEE Workshop Signal Processing'03, pp. 111-116, Poznan 2003.
[8] Danielsson P. E., Ye Q. Z., Rotation-Invariant Operators Applied to Enhancement of Fingerprints, Proc. 8th ICPR, Rome 1988.

[9] Hoogstrate A.J., Heuvel van den H., Huyben E., Ear Identification Based on Surveillance Camera's Images, Netherlands Forensic Institute, 2000.
[10] Hurley D.J., Nixon M.S., Carter J.N., Force Field Energy Functionals for Image Feature Extraction, Image and Vision Computing Journal, vol. 20, no. 5-6, 311-318, 2002.
[11] Iannarelli A., Ear Identification, Forensic Identification Series, Paramont Publishing Company, California 1989.
[12] Jain A., Bolle R., Pankanti S., Biometrics: Personal Identification in Networked Society, Kluwer Academic Publishers, 1999.
[13] Jain L. C., Halici U., Hayashi I., Lee S. B., Tsutsui S., Intelligent Biometric Techniques in Fingerprint and Face Recognition, CRC Press International Series on Computational Intelligence, 1999.
[14] Kouzani A.Z., He F., Sammut K., Towards Invariant Face Recognition, Journal of Information Sciences 123, Elsevier 2000.
[15] Lai K., Chin R., Deformable Contours: Modeling and Extraction, IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 17, no. 11, 1084-1090, 1995.
[16] Safar M., Shahabi C., Sun X., Image Retrieval By Shape: A Comparative Study, University of Southern California, November 1999.
[17] Victor B., Bowyer K.W., Sarkar S., An Evaluation of Face and Ear Biometrics, Proc. of Intl. Conf. on Pattern Recognition, I: 429-432, 2002.
[18] Zhang D., Automated Biometrics – Technologies and Systems, Kluwer Academic Publishers, 2000.

A Probabilistic Framework for Articulated Shape Recognition

Abdullah A. Al-Shaher and Edwin. R. Hancock

University of York
York YO1 5DD, UK
{abdullah,erh}@minster.cs.york.ac.uk

Abstract. This paper describes a probabilistic framework for recognising 2D shapes with articulated components. The shapes are represented using both geometrical and a symbolic primitives, that are encapsulated in a two layer hierarchical architecture. Each primitive is modelled so as to allow a degree of articulated freedom using a polar point distribution model that captures how the primitive movement varies over a training set. Each segment is assigned a symbolic label to distinguish its identity, and the overall shape is represented by a configuration of labels. We demonstrate how both the point-distribution model and the symbolic labels can be combined to perform recognition using a probabilistic hierarchical algorithm. This involves recovering the parameters of the point distribution model that minimise an alignment error, and recovering symbol configurations that minimise a structural error. We apply the recognition method to human pose recognition.

1 Introduction

The task of recognising articulated shapes has attracted considerable interest in computer vision. The main problem is how to robustly recover correspondence when the object under study undergoes deformations and the detected feature points defining the object are subject to noise. One of the most effective ways of developing matching techniques is to draw on probabilistic and statistical methods. This approach has lead to the development of point distribution models [1], deformable templates [2] and condensation [3]. One of the most important modes of variation in a moving shape, especially biological forms, is that of articulation.

There are a number of ways in which object articulation can be modelled. Perhaps the simplest of these is to decompose the shape into a skeletal form, consisting of limbs or branches, and to model the articulation of the branches. The mechanical movement of the resulting shape can be captured by the rotation of the components. However, in order to constrain the change in shape to be physically realistic bounds, or distributions, must be imposed on the rotation angles [4, 5]. Hence, the mechanical constraints on articulation must be combined with a statistical model of limb movement. In addition to movement of the limbs, the articulated shape also has a structural composition, since the limbs can be assigned labels to distinguish them, and the arrangement of the labels can be used to provide further constraints for shape-recognition.

The aim in this paper is to develop a statistical framework that can be used to recognise articulated shapes using information concerning limb movement and symbolic constraints concerning the overall shape structure. To do this, we develop a hierarchical algorithm. Each shape is represented as an arrangement of articulated limbs. The movement of the limbs is represented by a polar point distribution model. The structural component of the model is represented by a configuration of limb-labels. The recognition architecture has two intercommunicating layers. The first of these is concerned with limb alignment, and this aims to recover the lengths and polar angles of the limbs. The second aims to assign limb-labels so that the overall structure is recovered consistently. The fitting of this two-level model to data is effected using a variant of the expectation-maximisation algorithm.

2 Point Distribution Models

The point distribution model of Cootes and Taylor commences from a set training patterns. Each training pattern is a configuration of labelled point co-ordinates or landmarks. The landmark patterns are collected as the the object in question undergoes representative changes in shape. To be more formal, each landmark pattern consists of L labelled points whose co-ordinates are represented by the set of position co-ordinates $\{(x_1, y_1), ...(x_L, y_L)\}$. Suppose that there are T landmark patterns. The t^{th} training pattern is represented using the long-vector of landmark co-ordinates $X_t = (x_1, y_1, x_2, y_2, \cdots, x_L, y_L)^T$, where the subscripts of the co-ordinates are the landmark labels. For each training pattern the labelled landmarks are identically ordered. The mean landmark pattern is represented by the average long-vector of co-ordinates $Y = \frac{1}{T}\sum_{t=1}^{T} X_t$. The covariance matrix for the landmark positions is

$$\Sigma = \frac{1}{T}\sum_{t=1}^{T}(X_t - Y)(X_t - Y)^T \tag{1}$$

The eigenmodes of the landmark covariance matrix are used to construct the point-distribution model. First, the unit eigenvalues E of the landmark covariance matrix are found by solving the eigenvalue equation $|\Sigma - \alpha I| = 0$ where I is the $2L \times 2L$ identity matrix. The eigen-vector ϕ_i corresponding to the eigenvalue α_i is found by solving the eigenvector equation $\Sigma\phi_i = \alpha_i\phi_i$. According to Cootes and Taylor [1], the landmark points are allowed to undergo displacements relative to the mean-shape in directions defined by the eigenvectors of the covariance matrix Σ. To compute the set of possible displacement directions, the M most significant eigenvectors are ordered according to the magnitudes of their corresponding eigenvalues to form the matrix of column-vectors $\Phi = (\phi_1|\phi_2|...|\phi_M)$, where $\alpha_1, \alpha_2, ..., \alpha_M$ is the order of the magnitudes of the eigenvectors. The landmark points are allowed to move in a direction which is a linear combination of the eigenvectors. The updated landmark positions are given by $\hat{X} = Y + \Phi\gamma$, where γ is a vector of modal co-efficients. This vector represents the free-parameters of the global shape-model. When fitted to an observed set of landmark measurements X_o, the least-squares estimate of the parameter vector is

$$\gamma = \frac{1}{2}(\Phi + \Phi^T)(X_o - Y)$$

3 Shape Representation

Our aim is to use point distribution models to account for shape deformations due to limb articulation. The model is a two component one. First, we have a limb-model. This accounts for the variations in shape of each of the individual limbs using a point distribution model to describe the modes of variation of the landmark points about a mean shape. Second, we have a limb arrangement model. This is an augmented point distribution model that describes the arrangement of the centre points of the limbs, and their polar angles.

We are concerned with recognising 2D shapes by modelling segment movement around the centre of the shape. The shape under study is assumed to be segmented into a set of K jointed and non-overlapping limbs. The k^{th} limb is represented by a long-vector of consecutive landmark points

$$X_k = (x_1^k, y_1^k, x_2^k, y_2^k, \dots x_{n_k}^k, y_{n_k}^k)^T$$

The centre-of-gravity of the limb indexed k is

$$c_k = \frac{1}{n_k}\sum_{i=1}^{n_k}(x_i^k, y_i^k)^T$$

The overall shape is represented by a long-vector of consecutive limb centres $C = (c_1^T, c_2^T, .., c_K^T)^T$. The centre of articulated shape is computed by averaging the centre of the limbs

$$U = \frac{1}{K}\sum_{k=1}^{K} c_k$$

To model the articulated shape, we use a polar variant of the standard point distribution model [6]. This model allows the primitives to move about the centre of articulation According to this model the shape is viewed as an arrangement of non-overlapping primitives. Each primitive is represented by mean point c_k. Limb articulation is represented by a set of limb-angles. For the k^{th} limb the angle is defined to be

$$\theta_k = \tan^{-1}\frac{U(y) - c_k(y)}{U(x) - c_k(x)}$$

and the angular arrangement of the limbs is represented by the vector $\Theta = (\theta_1, \theta_2, ..., \theta_K)^T$. The global movement of the limbs within a shape is specified by the concatenated long-vector of angles and the centres-of-articulation, i.e. by the vector $S = (\Theta^T, C^T)^T$.

To augment the geometric information, we assign symbols to the articulated components. Each training pattern is assigned to a shape class and each component primitive is assigned to a primitive class. The set of shape-labels is Ω_c and the set of articulated component or limb labels is Ω_s. The symbolic structure of each shape is represented by a permissible arrangement of limb-labels. For shapes of class $\omega \in \Omega_c$ the permissible arrangement of limbs is denoted by $\Lambda_\omega =< \lambda_1^\omega, \lambda_2^\omega, ... >$.

4 Learning Mixtures of PDM's

In Cootes and Taylor's method [7], learning involves extracting a single covariance matrix from the sets of landmark points. Hence, the method can only reproduce variations in shape which can be represented as linear deformations of the point positions. To reproduce more complex variations in shape either a non-linear deformation or a series of local piecewise linear deformations must be employed.

In this paper we adopt an approach based on mixtures of point-distributions. Our reasons for adopting this approach are twofold. First, we would like to be able to model more complex deformations by using multiple modes of shape deformation. The need to do this may arise in a number of situations. The first of these is when the set of training patterns contains examples from different classes of shape. In other words, we are confronted with an unsupervised learning problem and need to estimate both the mean shape and the modes of variation for each class of object. The second situation is where the shape variations in the training data can not be captured by a single covariance matrix, and a mixture is required.

Our approach is based on fitting a Gaussian model to the set of training examples. We commence by assuming that the individual examples in the training set are conditionally independent of one-another. We further assume that the training data can be represented by a set of shape-classes Ω. Each shape-class $\omega \in \Omega_s$ has its own mean point-pattern Y_ω and covariance matrix Σ_ω. With these ingredients, the likelihood function for the set of training patterns is

$$p(X_t, t = 1, ..., T) = \prod_{t=1}^{T} \sum_{\omega \in \Omega_s} p(X_t | Y_\omega, \Sigma_\omega) \tag{2}$$

where $p(X_t|Y_\omega, \Sigma_\omega)$ is the probability distribution for drawing the training pattern X_t from the shape-class ω. According to the EM algorithm, we can maximise the likelihood function above, by adopting a two-step iterative process. The process revolves around the expected log-likelihood function

$$Q_{n+1} = \sum_{t=1}^{T} \sum_{\omega \in \Omega_s} P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)}) \ln p(X_t | Y_\omega^{(n+1)}, \Sigma_\omega^{(n+1)}) \tag{3}$$

where $Y_\omega^{(n)}$ and $\Sigma_\omega^{(n)}$ are the estimates of the mean pattern-vector and the covariance matrix for class ω at iteration n of the algorithm. The quantity $P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)})$ is the *a posteriori* probability that the training pattern X_t belongs to the class ω at iteration n of the algorithm. The probability density for the pattern-vectors associated with the shape-class ω, specified by the estimates of the mean and covariance at iteration $n+1$ is $p(X_t|Y_\omega^{(n+1)}, \Sigma_\omega^{(n+1)})$. In the M, or maximisation, step of the algorithm the aim is to find revised estimates of the mean pattern-vector and covariance matrix which maximise the expected log-likelihood function. The update equations depend on the adopted model for the class-conditional probability distributions for the pattern-vectors.

In the E, or expectation, step the *a posteriori* class membership probabilities are updated. This is done by applying the Bayes formula to the class-conditional density. At iteration $n+1$, the revised estimate is

$$P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)}) = \frac{p(X_t|Y_\omega^{(n)}, \Sigma_\omega^{(n)})\pi_{t,\omega}^{(n)}}{\sum_{\omega \in \Omega} p(X_t|Y_\omega^{(n)}, \Sigma_\omega^{(n)})\pi_{t,\omega}^{(n)}} \tag{4}$$

where

$$\pi_{t,\omega}^{(n+1)} = \frac{1}{T}\sum_{t=1}^{T} P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)}) . \tag{5}$$

4.1 Mixtures of Gaussians

We now consider the case when the class conditional density for the training patterns is Gaussian. Here we assume that the pattern vectors are distributed according to the distribution

$$p(X_t|Y_\omega^{(n)}, \Sigma_\omega^{(n)}) = \frac{1}{(2\pi)^L\sqrt{|\Sigma_\omega^{(n)}|}} \exp\left[-\frac{1}{2}(X_t - Y_\omega^{(n)})^T(\Sigma_\omega^{(n)})^{-1}(X_t - Y_\omega^{(n)})\right] \tag{6}$$

At iteration $n+1$ of the EM algorithm the revised estimate of the mean pattern vector for class ω is

$$Y_\omega^{(n+1)} = \sum_{t=1}^{T} P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)}) X_t \tag{7}$$

while the revised estimate of the covariance matrix is

$$\Sigma_\omega^{(n+1)} = \sum_{t=1}^{T} P(t \in \omega | X_t, Y_\omega^{(n)}, \Sigma_\omega^{(n)})(X_t - Y_\omega^{(n)})(X_t - Y_\omega^{(n)})^T \tag{8}$$

When the algorithm has converged, then the point-distribution models for the different classes may be constructed off-line using the procedure outlined in Section 2.

We apply this learning procedure separately to the landmark data for the individual limbs, and to the combined limb angle and limb-centre data. For the limb with label λ, the estimated modal matrix is Φ_λ and the estimated parameter vector is γ_λ. For the shape-class with label ω, on the other hand, the combined modal matrix for the articulation angles and limb-centres is $\tilde{\Phi}_\omega$ and the result of fitting to data is a parameter vector Γ_ω. The first K rows of $\tilde{\Gamma}_\omega$ correspond to the limb angles, and the remaining $2K$ to the long-vectors of limbs centres. However, we need to constrain the parameters corresponding to the limb angles. Suppose that the mean-vector for the limb-angles is $\hat{\Theta}_w$ and the corresponding covariance matrix is Σ_w. The angular deformations are constrained to avoid flipping by limiting the deformation vector. We use the variance associated with the eigenmodes to constrain the deformation. The k^{th} component of the parameter vector is constrained to fall in the interval $-3\sqrt{\alpha_k} \leq \Gamma_{(k)} \leq 3\sqrt{\alpha_k}$ The articulation angles lie in the range –180° to 180° to avoid discontinuities associated with the flip from 0° to 360°. A similar procedure for learning is followed to learn the variation in the polar representation of the limb and limb classes.

5 Hierarchical Architecture

With the limb-articulation and limb-centre point distribution models to hand, our recognition method proceeds in a hierarchical manner. Our aim is to classify the set of limb landmark long-vectors $X = \{z_1, ..., z_k, ..., z_K\}$ representing a test-shape. To commence, we make maximum likelihood estimates of the best-fit parameters of each limb-model to each set of limb-points. The best-fit parameters γ_λ^k of the limb-model with class-label λ to the set of points constituting the limb indexed k is

$$\gamma_\lambda^k = \arg\max_{\gamma} p(z_k|\Phi_\lambda, \gamma) \tag{9}$$

We use the best-fit parameters to assign a label to each limb. The label is that which has maximum a posteriori probability given the limb parameters. The label assigned to the limb indexed k is

$$l_k = \arg\max_{l \in \Omega_s} P(l|z_k, \gamma_\lambda, \Phi_\lambda) \tag{10}$$

In practice, we assume that the fit error residuals follow a Gaussian distribution. As a result, the class label is that associated with the minimum squared error. This process is repeated for each limb in turn. The class identity of the set of limbs is summarised by the string of assigned limb-labels $L =< l_1, l_2, ... >$. Hence, the input layer is initialised using maximum likelihood limb parameters and maximum a posteriori probability limb labels. The shape-layer takes this information as input. The goal of computation in this second layer is to refine the configuration of limb labels using global constraints on the arrangement of limbs to form consistent shapes. The constraints come from both geometric and symbolic sources. The geometric constraints are provided by the fit of a polar limbs point distribution model. The symbolic constraints are provide by a dictionary of permissible limb-label strings for different shapes.

The parameters of the limb-centre point distribution model are found using the EM algorithm [8]. Here we borrow ideas from the hierarchical mixture of experts algorithm [9], and pose the recovery of parameters as that of maximising a gated expected log-likelihood function for the distribution of limb-centre alignment errors $p(X|\Phi_\omega, \Gamma_\omega)$. The likelihood function is gated by two sets of probabilities. The first of these are the a posteriori probabilities $P(\lambda_k^\omega | z_k, \gamma_{\lambda_k^\omega}, \Phi_{\lambda_k^\omega})$ of the individual limbs. The second are the conditional probabilities $P(L|\Lambda_\omega)$ of the assigned limb-label string given the dictionary of permissible configurations for shapes of class ω. The expected log-likelihood function is given by

$$\mathcal{L} = \sum_{\omega \in \Omega_c} P(L|\Lambda_\omega) \left\{ \prod_k P(\lambda_k^\omega | z_k, \gamma_{\lambda_k^\omega}, \tilde{\Phi}_{\lambda_k^\omega}) \right\} \ln p(X|\tilde{\Phi}_\omega, \Gamma_\omega) \tag{11}$$

The optimal set of polar limb arrangement parameters satisfies the condition

$$\Gamma_\omega^* = \arg\max_{\Gamma} P(L|\Lambda_\omega) \left\{ \prod_k P(\lambda_k^\omega | z_k, \gamma_{\lambda_k^\omega}, \tilde{\Phi}_{\lambda_k^\omega}) \right\} \ln p(X|\tilde{\Phi}_\omega, \Gamma_\omega) \tag{12}$$

From the maximum likelihood alignment parameters we identify the shape-class of maximum *a posteriori* probability. The class is the one for which

$$\omega^* = \arg\max_{\omega \in \Omega_c} P(\omega | X, \tilde{\Phi}_\omega, \Gamma_\omega^*) \tag{13}$$

The class identity of the maximum *a posteriori* probability shape is passed back to the limb-layer of the architecture. The limb labels can then be refined in the light of the consistent assignments for the limb-label configuration associated with the shape-class ω

$$l_k = \arg\max_{\lambda \in \Omega_s} P(\lambda | z_k, \gamma_l^k, \tilde{\Phi}_\lambda) P(L(\lambda, k)|\Lambda_\omega) \tag{14}$$

Finally, the maximum likelihood parameters for the limbs are refined

$$\gamma_k = \arg\max_{\gamma} p(z_k | \tilde{\Phi}_{l_k}, \gamma, \Gamma_\omega^*) \tag{15}$$

These labels are passed to the shape-layer and the process is iterated to convergence.

6 Models

To apply the model to shape-recognition, we require models of the alignment error process and the label error process.

6.1 Alignment Errors

To develop a useful alignment algorithm we require a model for the measurement process. Here we assume that the observed position vectors, i.e. $\boldsymbol{z}_k$ are derived from the model points through a Gaussian error process. According to our Gaussian model of the alignment errors,

$$p(\boldsymbol{z}_k|\tilde{\Phi}_\lambda, \gamma_\lambda) = \frac{1}{2\pi\sigma} \exp\left[-\frac{1}{2\sigma^2}(\boldsymbol{z}_k - \hat{X}_\lambda - \tilde{\Phi}_\lambda\gamma_\lambda)^T(\boldsymbol{z}_k - \hat{X}_\lambda - \tilde{\Phi}_\lambda\gamma_\lambda)\right] \quad (16)$$

where σ^2 is the variance of the point-position errors which for simplicity are assumed to be isotropic. A similar procedure may be applied to estimate the parameters of the polar limb-angle distribution model.

6.2 Label Assignment

The distribution of label errors is modelled using the method developed by Hancock and Kittler [10]. To measure the degree of error we measure the Hamming distance between the assigned string of labels L and the dictionary item Λ. The Hamming distance is given by

$$H(L, \Lambda_\omega) = \sum_{i=1}^{K} \delta_{l_i, \lambda_i^\omega} \quad (17)$$

where δ is the Dirac delta function. With the Hamming distance to hand, the probability of the assigned string of labels L given the dictionary item Λ is

$$P(L|\Lambda_\omega) = K_p \exp[-k_p H(L, \Lambda_\omega)] \quad (18)$$

where $K_p = (1-p)^K$ and $k_p = \ln \frac{1-p}{p}$ are constants determined by the label-error probability p.

7 Experiment

We have evaluated our method on sets of images of people in different poses. Figure 1 shows an exampled of the images used in our experiments, and illustrates the steps used to abstract the human subject for the purposes of recognition. In Figure 1a, we show the original subject, who is wearing white markers to distinguish the main control points, or limb-centres, of our model. Figure 1b shows the subject " skeleton", and Figure 1c the set of strokes constituting the shape. In Figure 2 we show sets of example "skeletons" from a number of distinct body poses which we have used for the purposes of training. In total we have collected images of 14 distinct poses, and there are 20 examples of each pose. Each example is composed of 13 limb segments, and each limb-segment centre is identified by a marker. In Figure 3 we show four examples of the mean-shapes recovered as the output of the learning stage.

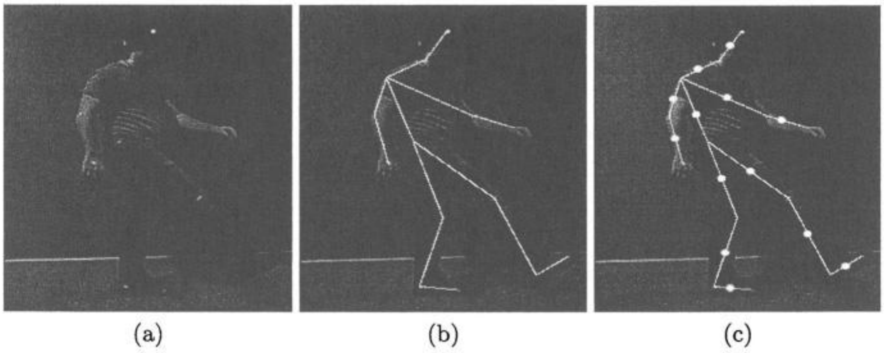


Fig. 1. Shape Representation. (a) shooting; (b) extracted shape limbs; (c) shape limbs center

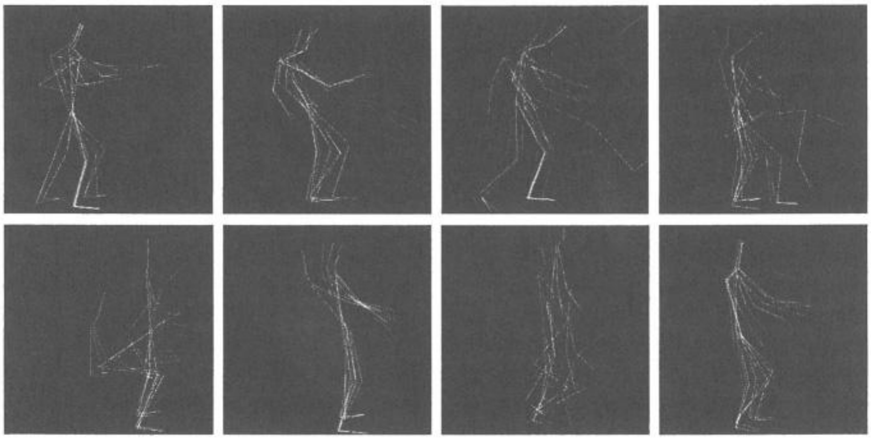


Fig. 2. Training Sets

To evaluate recognition performance, we have used 1200 images corresponding to different pose classes for testing. Figures 4 and 5 show the learned shape models iteratively aligning to the "Boxing" and "Stretching" pose samples. The figure illustrates how the model adapts in a flexible manner to fit the example shape in a relatively small number of iterations. In the figure the test-shape is shown in black lines while the model is shown in red.

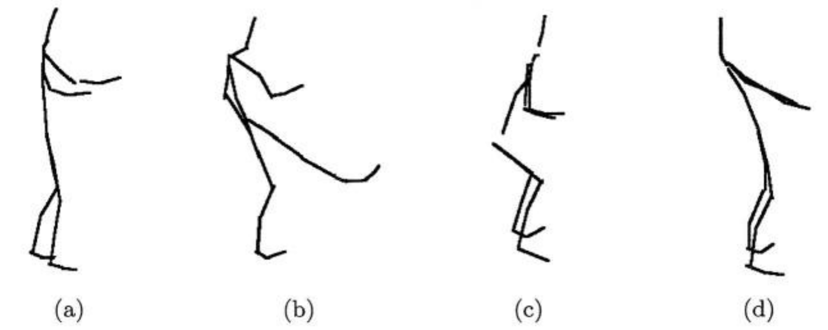


Fig. 3. Learnt Shapes:(a) Boxing, (b) Kicking, (c) Relaxing, (d) Stretching

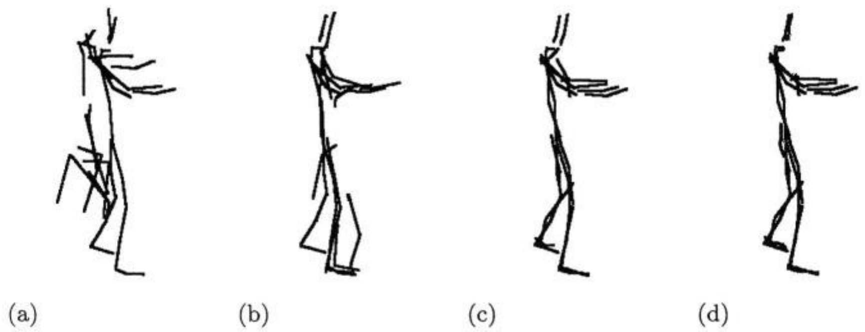


Fig. 4. Boxing Alignment: (a) iteration 1, (b) iteration 2, (c) iteration 3, (d) iteration 4

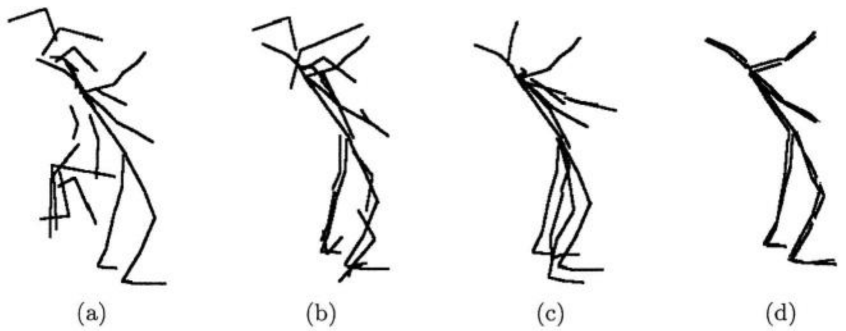


Fig. 5. Stretching Alignment: (a) iteration 1,(b) iteration 2, (c) iteration 3, (d) iteration 5

To illustrate the effectiveness of the recognition method when the input is confused, we have experimented with a test shape that overlaps two model shapes. In this example the test shape can fall into either the "Kicking" or "Shooting" classes. Figure 6 shows the alignment process. Initially, both the

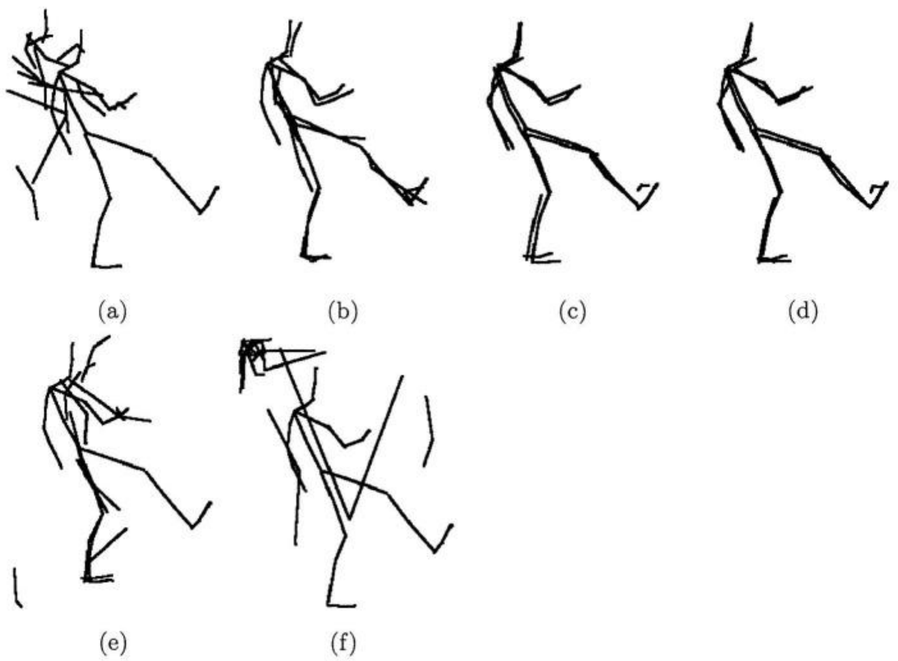


Fig. 6. Alignment: Kicking: (a) iteration 1, (b) iteration 2, (c) iteration 3, (d) iteration 4; Shooting: (e) iteration 1, (f) iteration 2

"Kicking" and "Shooting" hypotheses have low a *posteriori* probability. However, after several iterations the "Kicking" model dominates the alternative hypothesis.

To explore the capacity of the method to identify clusters of poses, we have applied principal components analysis and multidimensional scaling to the vectors of fitted model parameters extracted using our recognition method. Figures 7 and 8 respectively show the result of applying MDS and PCA to the alignment parameters for 54 sample shapes drawn at random from the set of shape-classes. In the top row of the two figures, we show the result of projecting the data onto the leading three eigenvectors. In the bottom row of the two plots, we show the matrix of pairwise distances computed from the projection of the data into the three dimensional eigenspace. In the left-hand column of each plot, we show the results obtained with the initial parameter vectors, while the second column shows the results with the parameter vectors at convergence. The main effect of iterating the recognition method is to improve the clusters of shapes. This is reflected both by the distribution of data in the eigenspace, and the block structure in the pairwise distance matrices. Moreover, a better cluster structure emerges when MDS is used rather than PCA. However, a deeper analysis of the data reveals that PCA gives a better cluster structure when the shapes are subject to scaling.

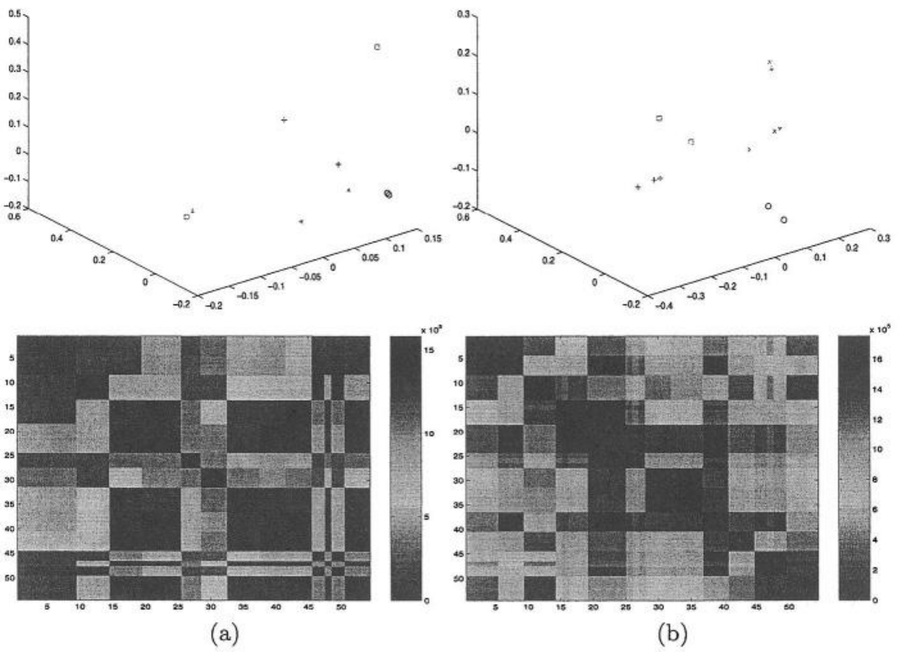


Fig. 7. MDS Classification: top row shows the embedding and the second row represent distance matrix; (a) iteration 1, (b) final iteration

To take this investigation one step further, Figure 9 shows the effect of adding random noise to the patterns. Here we apply MDS to the Euclidean distances between the final reconstructed shapes using the model parameters obtained at the final iteration of the fitting algorithm. The left-hand panel shows the MDS embedding of 6 shapes with 15 examples each. Here the different shape classes are shown in different colours. The right-panel shows the pairwise distance matrix. It is clear that the method produces good shape clusters even under conditions of noise.

Table 1 shows the recognition rates for six shape classes. The overall correct recognition rate is 93.16%. The poorest recognition occurs for the kicking, the picking and the shooting classes. Since these classes share similar limb configuration, we can conclude that recognition is reasonably high.

8 Conclusion

In this paper, we have described a method for fitting articulated shape-models to landmark point data. The shape deformation process adopted is based on point distribution models. The model representation is a hierarchical one. There

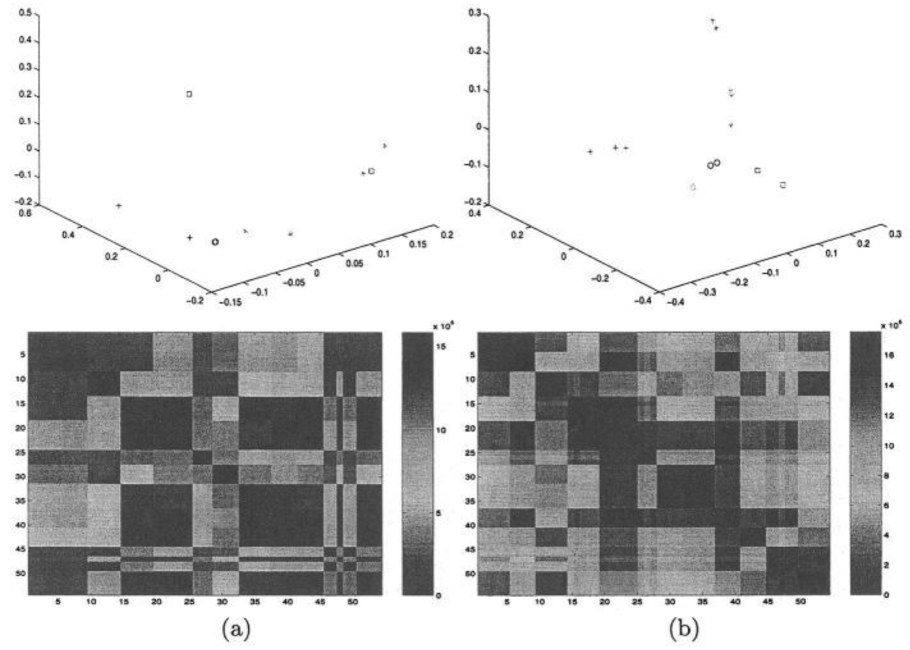


Fig. 8. PCA Classification: top row shows the embedding and the second row shows the distance matrix; (a) iteration 1, (b) final iteration

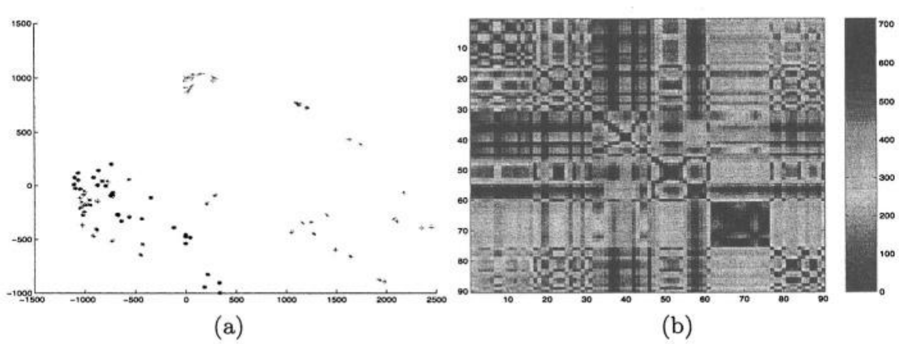


Fig. 9. MDS data-model distances. (a) 6 clusters of 90 shapes, (b) dissimilarity matrix

is a Cartesian deformation model for the limbs and the limb-centres, together with a polar model which represents limb articulation. We develop a probabilistic framework for fitting a mixture of articulated models to data. The method delivers good results of human shape modelling.

Table 1. Recognition rate for shape classes

Shape	Samples	Correct	Wrong
Boxing	200	198	2
Kicking	200	163	37
Relaxing	200	197	3
Picking	200	181	19
Stretching	200	200	0
Shooting	200	179	21
Recognition Rate		93.16%	6.83%

References

[1] Cootes T.; Taylor C. Combining point distribution models with shape models based on finite element analysis. *IVC*, 13(5):403–409, 1995.

[2] Duta N.; Jain A.; Dubuisson P. Learning 2d shape models. *International Conference on Computer Vision and pattern Recognition*, 2:8–14, 1999.

[3] Michael Isard; Andrew Blake. Contour tracking by stochastic propagation of conditional density. In *Proc. ECCV*, pages 343–356, 1996.

[4] J. Gonzales; J. Varona; F. X. Roca; and J. Villanueva. aspace:Action space for recognition and synthesis of human actions. *2 IWAMDO*, *Spain*, pp 189–200, 2002.

[5] James M. Rehg and Takeo Kanade. Visual tracking of high dof articulated structures: an application to human hand tracking. *3 ECCV*, *Sweden*, pp 35–46, 1994.

[6] Heap T.; Hogg D. Extending the point distribution model using polar coordinates. *Image and Vision Computing*, 14:589–599, 1996.

[7] Cootes T.; Taylor C. A mixture models for representing shape variation. *Image and Vision Computing,* 17:403–409, 1999.

[8] Dempster A.; Laird N.; Rubin D. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Soc. Ser.,* 39:1–38, 1977.

[9] Jordan M.; Jacobs R. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6:181–214, 1994.

[10] Edwin R. Hancock; Josef Kittler. Edge-labelling using dictionary-based relaxation. *IEEE Transaction on PAMI*, 12(2):165–181, 1990.

Neuroanatomy Registration: An Algebraic-Topology Based Approach

Layachi Bentabet[1] and Djemel Ziou[2]

[1] Bishop's University, Computer Science Department
Lennoxville, J1M1Z7 Québec, Canada
lbentabe@ubishops.ca
[2] Université de Sherbrooke, Département d'Informatique
Sherbrooke, J1K2R1 Québec, Canada
ziou@usherbrooke.ca

Abstract. In this paper, a method for image deformation is presented. It is based upon decomposition of the deformation problem into basic physical laws. Unlike other methods that solve a differential or an energetic formulation of the physical laws involved, we encode the basic laws using computational algebraic topology. Conservative laws are translated into exact global values and constitutive laws are judiciously approximated. In order to illustrate the effectiveness of our model, we utilize the viscous fluid model to achieve neuroanatomy image registration

1 Introduction

Physics-based deformation has been gaining in popularity since the introduction of active contours by Kass et al. [1]. The physics-based deformation problem is solved by combining the basic physical laws that constitute the fundamental equations of continuum mechanics. These equations may be developed in two separate but essentially equivalent formulations. One, the integral or global form, derives from a consideration of the basic principles being applied to a finite volume of the material. The tessellation of space into finite volumes using the finite elements method (FEM) or the finite volumes method (FVM) gives rise to discrete equations that can be directly implemented if the physical space is discrete in nature, as in the case of an image. The other, differential or field approach, involves equations resulting from the basic principles being applied to a very small (infinitesimal) element of volume. As a result of the continuum assumption, field quantities such as velocity and displacement which reflect the mechanical or cinematic properties of continuum bodies are expressed mathematically as continuous functions, or at worst as piecewise continuous functions, of the space and time variables. Moreover, the derivatives of such functions, if they are considered at all, should be continuous. In practice, it is often proposed to derive the field equations from their global counterparts instead of using the global forms directly. The convenience of the differential forms often argues in favour of this. However, the major drawback of such an approach when dealing with the deformation problem is that the description of the material will be accurate only if the displacement and velocity fields vary slowly over the size of the elements used [3]. This drawback arises directly from the use of a differential formulation of the physics-based laws involved. Indeed, field equations in a differential formulation are subject to restrictions imposed by derivability, restrictions that are not related to the physical phenomenon investigated.

In this paper we introduce a new formulation of physics-based image deformation. The proposed formulation is based on the computational algebraic topology (CAT) based image model introduced by Ziou and Allili [4]. In our model, we propose to derive the equations governing the viscous fluid based deformation directly for a discrete volume, not for an infinitesimal point, using the basic laws instead of the differential equations. In fact, we will show that there is no need to write the equilibrium balance for an infinitesimal point, thereby introducing a differentiability restriction, when equilibrium holds for a whole region. The proposed approach is validated through a series of tests on neuroanatomy registration.

2 The CAT-Based Image Model

In the CAT model [4], an image is composed of two distinctive parts: the image support and some field quantity associated with it. The image support is a complex of unit cells, usually called *pixels.* A pixel of dimension q is called a q-pixel. Hence, the pixel $\gamma \subset \Re^2$ in figure 1 is a 2-pixel. The boundaries of γ are 1-pixels, referred to by us as the 1-faces of γ. Similarly, the boundaries of each 1-pixel are 0-pixels, which we refer to as 0-faces. A natural orientation is associated with each of these faces as indicated in figure 1. We define a *cubical complex* in $\Re^n$ as a finite collection K of q-pixels. A *q-chain* is defined as a linear combination of q-pixels. A formal expression for a q-chain c_q is $c_q = \sum_{\gamma_i \in K} \lambda_i \gamma_i$, where $\lambda_i \in \mathrm{Z}$. The last step needed for the description of the image support is the introduction of the concept of a *boundary of a chain.* Given a q-pixel γ, we define its boundary, $\partial\gamma$, as the (q-1)-chain corresponding to the alternating sum of its (q-1)-faces. The sum is taken according to the orientation of the (q-1)-faces relative to the orientation of the q-pixel. In order to model the pixel quantity over the image plane, we look for an application f_q, which associates a global quantity with all q-pixels. The resulting application f_q is called a *q-cochain* and may be any mathematical entity, such as a scalar or a vector. Finally, let's define a generic operation that links a (q+1)-cochain and a q-cochain defined on its boundaries. This relationship is described by the coboundary operator δ_q. Given a (q+1)-chain γ, this operator is defined by $\delta_q f_q(\gamma) = f_q(\partial_{q+1}\gamma)$. The coboundary is defined as the signed sum of the physical quantities associated with the q-faces of γ. The sum is taken according to the relative orientation of the q-faces of the (q+*1*)-pixels of γ.

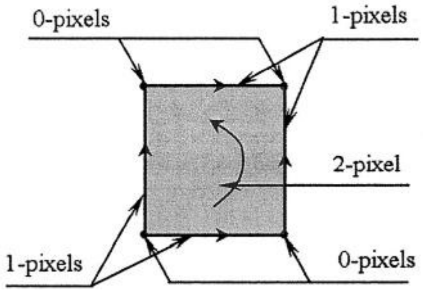


Fig. 1. A 2-pixel γ and its boundaries

3 Physical Modeling of the Deformation Problem

The first step consists of deriving the *conservation of momentum* law, which establishes the relationship between the external forces applied to a material and the resulting deformation. Consider a material, *M,* subject to a system of *external forces,* denoted by $\vec{F}^{ext}$. In the case of viscous fluids, *internal forces,* $\vec{F}^{int} = \iiint_{\Omega} \rho\vec{b}\, dV$, will be developed to counterbalance the external forces. In order to describe the dynamic behaviour of the material, we use Newton's second law, which states that the resultant force acting on a body, with a density of material ρ, moving at velocity $\vec{v}$ is equal to the rate of change over time of the linear momentum $\rho\vec{v}$. Hence

$$\frac{d}{dt}\iiint_{\Omega}\rho\vec{v}\,dV = \vec{F}^{int} + \vec{F}^{ext} = \iiint_{\Omega}\vec{\nabla}\cdot\sigma\,dV + \iiint_{\Omega}\rho\vec{b}\,dV, \tag{1}$$

where $\sigma = \lfloor\sigma_{ij}\rfloor$ is the 3 by 3 stress tensor [3]. In what follows, we will focus on the equilibrium state of the material. Hence, solving the deformation problem will be equivalent to determining the material state after the deformation has been achieved. A viscous fluid is said to be in equilibrium if $\frac{d\vec{v}}{dt} = \vec{0}$. Hence, the conservation of momentum equation for a viscous fluid can be rewritten at equilibrium as follows:

$$\vec{F}^{int} + \vec{F}^{ext} = \iiint_{\Omega}\left(\vec{\nabla}\cdot\sigma + \rho\vec{b}\right)dV = \vec{0}. \tag{2}$$

Note that this equation is a conservative equation, which provides a direct link between the external forces $\iiint_{\Omega}\rho\vec{b}\,dV$ and the internal forces $\iiint_{\Omega}\vec{\nabla}\cdot\sigma\,dV$. The stress tensor is related to the strain tensor by the constitutive law which is local by nature and which defines the behaviour of the viscous fluid when subject to stress. This relationship is given by the Hooke's law [3]:

$$\sigma = [\lambda(tr(\varepsilon)) - p]Id + 2\mu\varepsilon, \tag{3}$$

where λ, and μ are the Lame's constants, p is the pressure, tr is the trace operator, Id is the identity tensor, and ε is the strain tensor. Finally, the strain is defined by the well-known strain-velocity relationship [5]:

$$\varepsilon_{ij} = \frac{1}{2}\left[\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right], \qquad i, j = 1,2,3. \tag{4}$$

In the reminder of this paper, we will consider another tensor d, where

$$d = \vec{\nabla}\vec{v} = \left(\left(\vec{\nabla}v_1\right)^T, \left(\vec{\nabla}v_2\right)^T, \left(\vec{\nabla}v_3\right)^T\right). \tag{5}$$

It is easy to see that ε is the symmetric part of d, that is: $\varepsilon = \dfrac{d + d^T}{2}$.

4 CAT-Based Formulation of the Deformation Problem

The conservative laws involved are: the conservation of momentum in equation (2) linking the external forces to stress; and the strain-velocity relation in equation (5). In the context of CAT, the velocity and the strain quantities are associated with a complex describing the image support configuration. This complex is called K^P and it is positioned in such a way that its 0-pixels are centered on the image pixels (Fig. 2.-a). Besides, the external forces and stress which describe the dynamic state of the fluid are associated with a dual complex K^S. This complex is positioned in such a way that its 2-pixels coincide with the image pixels (figure 2-a). A 2-pixel γ_F from K^S intersects four 2-pixels of K^P. This construction allows us to write the equilibrium equation for each 2-pixel γ_F. This involves examining and collecting the contributions of each portion of 2-pixels from K^P surrounding γ_F. In this way, equilibrium relations are established over γ_F directly in a discrete form, without approximation. In order to encode the strain-velocity relation in equation (5), consider a 2-pixel γ_P of K^P as shown in figure 2-b. The strain tensor d can be modeled as a 1-cochain $\mathbf{D}_1$ positioned on the 1-faces γ_{Di} of γ_P with $\partial\gamma_{Di} = x_* - x_\#$ (figure 2-c). Thus

$$\mathbf{D}_1^i\left(\gamma_{D_i}\right) = \int_{\gamma_{Di}} d\, dl = \int_{x_\#}^{x_*} \vec{\nabla}\vec{v}\, dl. \tag{6}$$

Since the velocity is known only at the 0-faces of γ_P, it is modeled as a 0-cohain, noted $\mathbf{\Psi}_0$, positioned on the 0-faces $\partial\gamma_{Di}$. By evaluating the line integral in equation (6), it could be shown that the cochain $\mathbf{D}_1$ is the coboundary of $\mathbf{\Psi}_0$. So that

$$\mathbf{D}_1^i(\gamma_D) = \delta\mathbf{\Psi}_0^i\left(\gamma_{D_i}\right) = \mathbf{\Psi}_0^i\left(\partial\gamma_{D_i}\right) = \mathbf{\Psi}_0^i(x_*) - \mathbf{\Psi}_0^i(x_\#), \tag{7}$$

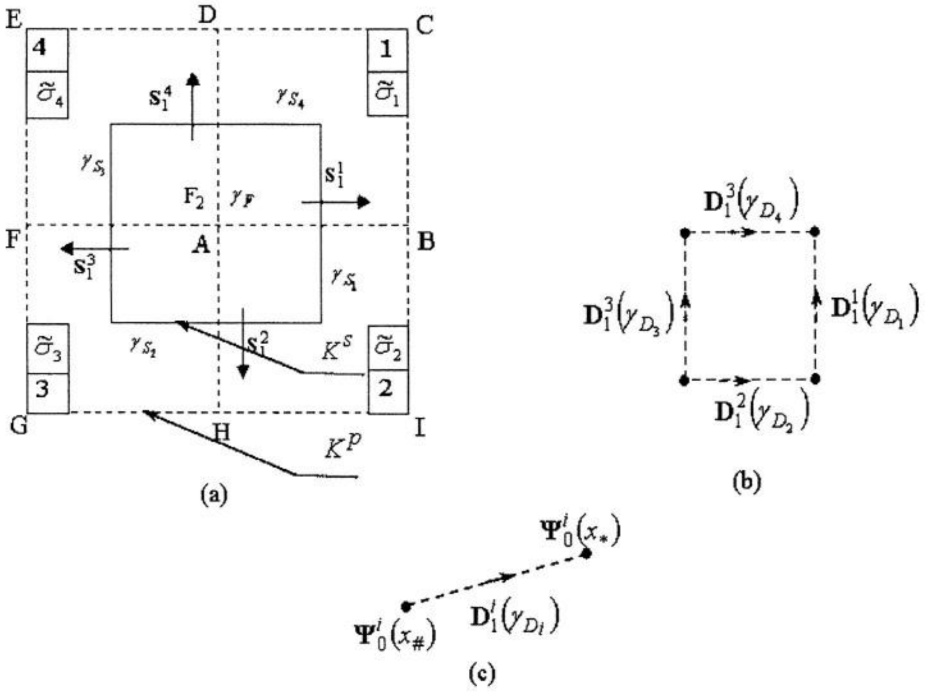


Fig. 2. a) Complexes K^p and K^s, b) cochain $\mathbf{D}_1$, c) cochain $\mathbf{\Psi}_0$

which is a discrete representation of equation (5).

In order to express the conservation of momentum in equation (2), consider a 2-pixel γ_F from K^s as shown in figure 2-a. The external forces over the surface of γ_F are expressed as a 2-cochain $\mathbf{F_2}(\gamma_F)$. In the equilibrium state, this cochain is calculated from equation (2) by:

$$\mathbf{F}_2(\gamma_F) = \iint_{\gamma_F} \rho\vec{b}\, dS = -\iint_{\gamma_F} \vec{\nabla}\cdot\sigma\, dS\,. \tag{8}$$

Applying the divergence theorem on equation (8), we have

$$\mathbf{F}_2(\gamma_F) = -\int_{\partial\gamma_F} \sigma\,\vec{n}\, dl = -\sum_{i=1}^{4}\int_{\gamma_{S_i}} \sigma\,\vec{n}_{S_i}\, dl\,, \tag{9}$$

where γ_{S_i} is the i^{th} 1-face of γ_F and $\vec{n}_{S_i}$ is the normal vector to γ_{S_i}. This allows us to define the stress tensor in equation (9) as a 1-cochain, as follows:

$$\mathbf{S}_1^i(\gamma_{S_i}) = \int_{\gamma_{S_i}} \sigma\,\vec{n}_{S_i}\, dl\,. \tag{10}$$

The 1-cochain $\mathbf{S}_1^i$ is positioned on the 1-face γ_{S_i}. The cochain $\mathbf{F}_2$ is the coboundary of $\mathbf{S}_1$, so that

$$\mathbf{F}_2(\gamma_F) = -\delta\mathbf{S}_1(\gamma_F) = -\mathbf{S}_1(\partial\gamma_F) = -\sum_{i=1}^{4}\mathbf{S}_1^i(\gamma_{S_i}), \tag{11}$$

which is the discrete representation of equation (2). Since γ_F intersects four 2-pixels of K^P, we use four approximation functions $\tilde{\sigma}^1, \tilde{\sigma}^2, \tilde{\sigma}^3$ and $\tilde{\sigma}^4$, which describe the stress tensors on each 2-pixel surrounding γ_F. These functions are used to build the stress cochains in equation (11):

$$\mathbf{S}_1^1(\gamma_{S_i}) = \int_0^{\frac{\overline{AD}}{2}} \tilde{\sigma}^1\left(\frac{\overline{AB}}{2}, x_2\right)\vec{n}_1\,dx_2 + \int_{-\frac{\overline{AH}}{2}}^{0} \tilde{\sigma}^2\left(\frac{\overline{AB}}{2}, x_2\right)\vec{n}_1\,dx_2,$$

$$\mathbf{S}_1^2(\gamma_{S_i}) = \int_0^{\frac{\overline{AB}}{2}} \tilde{\sigma}^2\left(x_1, -\frac{\overline{AH}}{2}\right)\vec{n}_2\,dx_1 + \int_{-\frac{\overline{AF}}{2}}^{0} \tilde{\sigma}^3\left(x_1, -\frac{\overline{AH}}{2}\right)\vec{n}_2\,dx_1,$$

$$\mathbf{S}_1^3(\gamma_{S_i}) = \int_0^{\frac{\overline{AD}}{2}} \tilde{\sigma}^4\left(-\frac{\overline{AF}}{2}, x_2\right)\vec{n}_3\,dx_2 + \int_{-\frac{\overline{AH}}{2}}^{0} \tilde{\sigma}^3\left(-\frac{\overline{AF}}{2}, x_2\right)\vec{n}_3\,dx_2,$$

$$\mathbf{S}_1^4(\gamma_{S_i}) = \int_0^{\frac{\overline{AB}}{2}} \tilde{\sigma}^1\left(x_1, \frac{\overline{AD}}{2}\right)\vec{n}_4\,dx_1 + \int_{-\frac{\overline{AF}}{2}}^{0} \tilde{\sigma}^4\left(x_1, \frac{\overline{AD}}{2}\right)\vec{n}_4\,dx_1 .$$

The strain tensor in the Hooke's law is replaced by a local piecewise approximation, so that for each 1-face γ_{D_i} of γ_P, equation (3) becomes:

$$\tilde{\sigma} \approx \left[\lambda\left(tr\left(\frac{\tilde{d} + \tilde{d}^T}{2}\right)\right) - \kappa\varphi\right]I + \mu\left(\tilde{d} + \tilde{d}^T\right), \tag{12}$$

where the "~" sign stands for a local approximation of a given variable. In order to express equation (12) at a local level, the strain tensor $\tilde{d}$ must be calculated for each location within γ_P. Since $\tilde{d}$ is derived from the velocity $\vec{v}$ (see equation (5)), the latter must be locally known. In the previous sections, the 0-cochain $\mathbf{\Psi}_0$ was associated with the 0-pixels of K^p ; hence, $\vec{v}$ is derived using a bilinear interpolation of order 1 over γ_p. The bilinear interpolation of Ψ_0 over γ_p can be expressed as follows [6]:

$$\vec{v}(x_1,x_2)=\sum_{i=1}^{2}\left(v_{i,00}+\begin{pmatrix} \frac{v_{i,00}-v_{i,10}}{\overline{AB}} \\ \frac{v_{i,00}-v_{i,01}}{\overline{AD}} \\ \frac{v_{i,00}+v_{i,11}-v_{i,01}-v_{i,10}}{\overline{AB}\,\overline{AD}} \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} -x_2 \\ -x_1 \\ x_1x_2 \end{pmatrix} \right)\vec{e}_i , \tag{13}$$

where the velocity at the pq^{th} 0-pixel is given by $\vec{v}_{pq}=\left(v_{1,pq},v_{2,pq}\right)^T$. Finally, the external forces cochain can be rewritten in terms of the velocity on the 0-pixels surrounding γ_F , as follows:

$$\mathbf{F}_2(\gamma_F)=-\sum_{i=1}^{4}\int_{\gamma_{S_i}}\tilde{\sigma}\,\vec{n}_i\,dl . \tag{14}$$

Equation (14) can be considered as a linear algebraic system that can be rewritten as the product of a matrix encoding the basic physical laws and a vector containing the velocity values:

$$\mathbf{F}_2(\gamma_F)=\begin{pmatrix} F_1^{ext} \\ F_2^{ext} \end{pmatrix}=\begin{bmatrix} k_{1,1} & k_{1,2} & \cdots & k_{1,18} \\ k_{2,1} & k_{2,2} & \cdots & k_{2,18} \end{bmatrix}\begin{bmatrix} v_{1,00}^r \\ v_{2,00}^r \\ v_{1,01}^r \\ v_{2,01}^r \\ \vdots \\ v_{1,-1-1}^r \\ v_{2,-1-1}^r \end{bmatrix}, \tag{15}$$

where

$$k_{1,1}=k_{2,2}=\left(-\frac{9}{2}\mu+\frac{3}{2}\lambda\right),\ k_{1,3}=k_{1,9}=k_{2,4}=k_{2,10}=\left(\frac{1}{4}\mu-\frac{1}{4}\lambda\right),$$

$$k_{1,5}=k_{1,11}=k_{2,6}=k_{2,12}=\left(\frac{5}{4}\mu-\frac{3}{4}\lambda\right),$$

$$k_{1,7}=k_{1,13}=k_{1,15}=k_{1,17}=k_{2,8}=k_{2,14}=k_{2,16}=k_{2,18}=\left(\frac{3}{8}\mu+\frac{1}{8}\lambda\right),$$

$$k_{1,8} = k_{1,14} = k_{1,16} = k_{1,18} = k_{2,7} = k_{2,13} = k_{2,15} = k_{2,17} = \left(\frac{1}{4}\mu + \frac{1}{4}\lambda\right),$$

$$k_{1,2} = k_{1,4} = k_{1,6} = k_{1,10} = k_{1,12} = k_{2,1} = k_{2,3} = k_{2,5} = k_{2,9} = k_{2,11} = 0.$$

Once the velocity at a given time t is determined, the resulting displacement at time $t+1$, $\vec{u}_{t+1}$,can be calculated using the discrete material equation given by [2]:

$$\vec{u}_{t+1} = \vec{u}_t + \Delta t\left(I - \left(\vec{\nabla}\vec{u}_t\right)^T\right)\vec{v}_t. \tag{16}$$

5 Experimental Results

In neuroanatomy registration, the main objective is to match an image I_1 called the *template* to another image I_2 called the *study*. The study is usually a set of images representing a healthy brain. The registration of a patient's brain onto an atlas consists of finding the set of geometrical transformations that must be applied to the atlas to fit the patient's brain images. Once the transformations are known, all of the information contained in the atlas, such as structure names, sizes, locations, and atlas segmentation, is mapped onto the patient's brain images. One of the most cited work in the literature is the viscous fluid model proposed by Christensen et al. [2]. In their work, Christensen et al. proposed to solve the following PDE:

$$\mu\nabla^2\vec{v} + (\lambda + \mu)\vec{\nabla}\left(\vec{\nabla}\cdot\vec{v}\right) = \vec{F}^{ext}. \tag{17}$$

An iterative scheme is then derived to solve equation (17) using the successive over-relaxation (SOR) with checkerboard update method [2]. This approach will be compared to the CAT-based image deformation as described in section 4. Hence, the brain images will be considered as viscous fluids. The external force in equation (15), which deforms the template onto the study, is derived from the derivative of a Gaussian sensor model and defined as follows:

$$\vec{F}_n^{ext} = -\left(I_1(x_1,x_2) - I_2(x_1,x_2)\right)\ \vec{\nabla}I_1, \tag{18}$$

where $\vec{\nabla}I$ is the gradient of I. The external force in the equation above attempt to make edges in the template and the study fit. Note that the use of such a force requires the template to be initially aligned with the study so that they overlap. Experiments on brain images will be presented below. The resulting registration and time processing are compared with those obtained using the SOR algorithm. In all experiments, we assume that the template and the study contain the same structures, λ and μ are set at 1 and 5, respectively. The experiments were performed on a sample of CT scan images measuring 256 × 256 pixels. The left column in figure 3 shows the templates and the right column the studies. Pairs (A) and (B), are not equivalent since they do not contain the same configuration of grey/white matter. The deformed templates obtained using the CAT-based algorithm are shown in the middle column. Notice that the algorithm accommodates local shape variations and large-scale deformations well, with a good level of detail. However, there are some situations where the algorithm cannot recover the study shape, such as the grey matter in pair (A). This is mainly due to violation of the initial assumptions concerning similarity of the white/grey matter configuration. A 3D registration can overcome this problem, since the information from other slices will make the template and the study match. The second column in table 1 indicates the mean and standard deviation of the grey-level difference between the deformed templates and the studies for the CAT-based algorithm. The third column indicates the same measures when the SOR algorithm is employed. The obtained measures indicate that the CAT-based algorithm yields the best performance in terms of accuracy. Moreover, it is important to mention that the CAT-based algorithm takes only 10 iterations to perform the deformation, whereas the SOR requires 250 iterations to solve equation (17) and 100 iterations to accomplish the registration.

Table 1. Mean and standard deviation of the difference between the deformed template and the study

Experiment	Our method	SOR method
A	$\mu = 6.0$ $\sigma = 12.6$	$\mu = 10.9$ $\sigma = 19.2$
B	$\mu = 6.8$ $\sigma = 13.8$	$\mu = 9.3$ $\sigma = 17.2$
C	$\mu = 6.2$ $\sigma = 13.2$	$\mu = 9.1$ $\sigma = 16.6$

6 Conclusion

In this paper, we have presented a new approach for image deformation utilizing viscous fluid model. In the proposed approach, the image model is based on algebraic topology. This allows us to decompose the image deformation problem into one of basic physical laws. Cochains encode these laws over complexes and are linked together using coboundaries and codual operators. The major advantage of such an approach lies in the possibility of solving the deformation problem directly from the exact global forms rather than from discrete differential forms. Consequently, errors resulting from the approximation of continuous fields (i.e., displacement and velocity) and derivative operators by discrete forms can be reduced. Furthermore, the idea of expressing the physics-based deformation in a modular way is very interesting since it can be applied to solve other pattern-recognition problems based on a physical interpretation. CAT-based deformation under the viscous fluid assumption was tested successfully for the neuroanatomy registration.

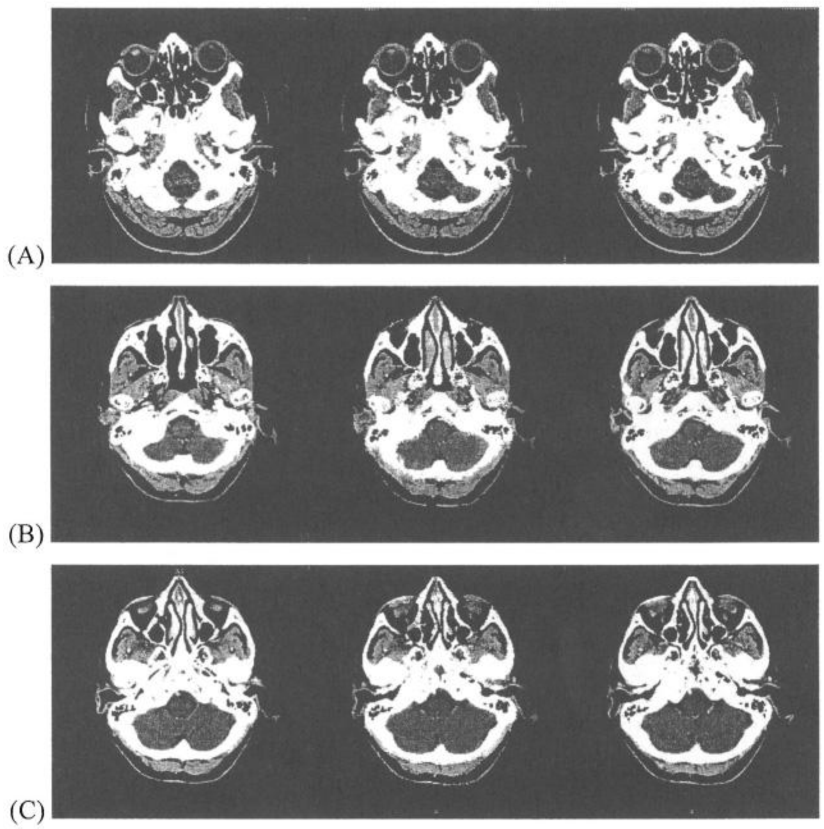


Fig. 3. Left: template; middle: after viscous fluid deformation; right: study

References

[1] Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active Contour Models. Int. J. of Comp. Vision 1, Vol. 4 (1988) 321–331

[2] Christensen, G. E., Joshi, S., Miller, M. I.: Volumetric transformation of brain anatomy. IEEE Tran. on Med. Imag., 16 Vol. 6 (1997) 864-877

[3] Cosmi, F.: Material response with the cell method. The 10th International Conference on Fracture, Hawaii (2001)

[4] Ziou, D., Allili, M.: Generating cubical complexes from image data and computation of the Euler number. Pattern Recognition, 35 Vol. 12, (2002) 2833-2939

[5] Boresi, A. P.: Elasticity in Engineering Mechanics. Prentice-Hall (1965)

[6] Auclair-Fortier, A. –F., Poulin, P., Ziou, D., Allili, M.: A Computational Algebraic Topology Model for the Deformation of Curves. The Second International Workshop on Articulated Motion and Deformable Objects, Spain, Vol. 2492 (2002) 56-67

Recognition and Tracking of the Members of a Moving Human Body*

Costas Panagiotakis and Georgios Tziritas

Department of Computer Science, University of Crete
P.O. Box 2208, Heraklion, Greece

Abstract. We present a method to solve the human silhouette tracking problem using 18 major human points. We used: a simple 2D model for the human silhouette, a linear prediction technique for initializing major points search, geometry anthropometric constraints for determining the search area and color measures for matching human body parts. In addition, we propose a method to solve the problem of human members recognition and 18 major human points detection using silhouette. This result can be used to initialize a human tracking algorithm for real time applications. Our main purpose is to develop a low computation cost algorithm, which can be used independently of camera motion. The output of the tracking algorithm is the position of 18 major human points and a 2D human body extraction. In cases of low quality imaging conditions or low background contrast, the result may be worst. For these cases we defined an appropriate criterion concerning tracking ability.

1 Introduction

The analysis of human motion using computer vision techniques attempts to detect, track and identify people and recognize their activity. There are many applications in many areas such as analysis of athletic events, 3D virtual reality based on real motion, video conferencing, content-based image storage and retrieval, etc.

There has been a significant number of recent papers on human tracking. The following works give an overview of the various task involved in motion analysis of the human body. Aggarwal and Cai [1] focus on major areas relating to interpreting human motion, like motion analysis, body parts tracking and recognition of human activities using image sequences. In a more recent work, Wang, Hu and Tan [6] emphasize on three major issues of human motion analysis systems, namely human detection, tracking and activity understanding. According of them, there are 2D, with or without explicit shape models, and 3D approaches.

Firstly, we consider 2D approaches. The system called W^4 [3] uses a statistical-background model to locate people using stable cameras. W^4 uses a statistical-background model to locate people using stable cameras. W^4 allows multiple person groups and detects and tracks six main body parts of each

* This work was supported by the DXH project.

person using a static-shape model. In our silhouete analysis method we use silhouette statistics like silhouette projections as in the W^4. Cheng and Moura [2] represent a system that tracks a walking human with monocular video using a 2D model, motion and texture. The human walker can be shaped by a stick model with 12 cone-shaped body parts. The above 2D models are similar to our 2D model mainly in that they both use human joints. Wang, Ning and Tan [7] propose a method to recognize and track a walker using 2D human model and both static and dynamic cues of body biometrics. Kakadiaris and Metaxas [4] present a 3D model-based method for motion estimation of human movement from multiple cameras.

1.1 Objectives

The main objectives of the silhouette analysis method is the the 18 major human points detection using silhouette image. In this problem, there are many solutions because of the projection from 3D in 2D image plane. Our method provides only one of them without using color information. The method is executed automatically, indepedent of silhouette rotation, scaling, in complex (with overlapping parts) - noisy silhouette, and without any initial information about the silhouette. The 2D human silhouette tracking in the whole sequence, tracking the set of predefined major human points and constructing a 2D segmentation map from them in every image of the sequence. As input, we use color images from a single, uncalibrated and maybe moving camera. The acquisition of realistic human motion by combining anthropometric geometrical constraints, color segmentation using an initial learning method, and time series information using a prediction method. The efficient searching that reduces the complexity and speeds up the algorithm without accuracy loss, is an important feature of our algorithm.

The main hypothesis of our method is that the human head and main body are almost completely visible in the whole sequence. Also, the color of any separate human part should not be changed too much. Colors are represented in the *Lab* color system, because it is closest to human color dissimilarity perception and the distance between colors can be modeled by Euclidean distance. The EMD is based on a distance between histograms. The EMD allows partial matching with very good performance [5].

The paper is organized as follows. The section 2 describes the human silhouette analysis, containing the human members recognition and major human points estimation method. In section 3 the human tracking algorithm is examined. Finally, sections 4, 5 provide experiments results and the discusion.

1.2 Overview

In this section we are going to specify the data collection methods, the main algorithmic steps, the model adopted and the hypothesis used. The silhouette analysis method can be divided into two stages. In the first stage, human members are recognized. In the second stage, the 18 major human points are estimated using the result of the stage one. Both of the stages have been developed according to the following rule. First, the head is searched, because it has a well defined and static shape. Next, we compute the position of the main body, because it can be easily determined using the head position. Finally, we can compute the position of the legs and the arms. The main steps of the human tracking method are the following.

- Initialization: Dominant colors extraction for every body part based on joint points position. It is executed only in the first image of the sequence.
- Tracking loop for every part sequentially: head, main body, legs and arms. For each subpart it is executed: Color Distances Computation, Visibility Computation, Position Prediction, Search by best matching.

The human tracking method is executed automatically apart from the initialization step. In order to minimize the complexity cost, a sequential algorithm is used for each part or subpart. This is required because the complexity would be $O(N^{36})$ for 18 points in the image plane, where N is the number of image points. Using an appropriate sequential algorithm, we can reduce the complexity to $O(N^2)$. First, we search for the head, because it is the most visible part, and it usually has specific and distinctive color and texture. If we estimate the head position correctly, then we can predict with sufficient accuracy the position of the main body, and finally we can compute the position of the legs and the arms. Following this sequential search, the search area of the body is significantly reduced.

Our method uses color images, because we want to use the whole color information to get better results. We use a 2D human silhouette model, which was created by the position of 18 major human points. Moreover, the human body is divided into 6 parts (Figure 3): head, main body, left arm, right arm, left leg and right leg. Some parts, like legs and arms, have articulated motion. For this reason we divide these parts into subparts. We use two subparts for every arm and three subparts for every leg. The 2D model that we use comprises an ellipse model for the head, and polygon models for the other parts.

2 Human Silhouette Analysis

2.1 Human Body Parts Recognition

In this section we examine the human body members recognition method. The human body is divided into the following six members: head, main body, left leg, right leg, left arm and right arm. The human silhouette pixels will be classified to one of the above members. An example of silhouette image is shown in Figure 1, the white pixels correspond to the human body, while the gray pixels correspond to the background. The member recognition algorithm is sequential. The more visible members are computed first decreasing the search space of the others.

First, the major human body axis is determined using central moments. The silhouette is rotated according to the major axis. The rotation center is the mass

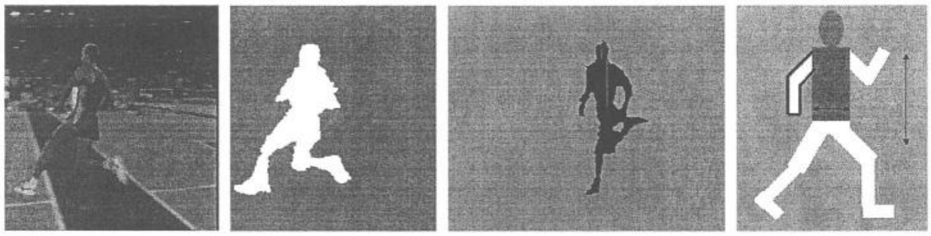


Fig. 1. The second figure shows the silhouette of the human of the first figure. An example of successful execution of the end of head (blue point) localization method (third figure). The left arm, of the right figure, can be distinguished from the main body because the proportion of red boundary pixels is high (fourth figure)

center of the human. Now, the human is located vertically. Next, we estimate the end point of the head (X_d, Y_d) using the following iterative method (Figure 1). The (X_d, Y_d) point is initialized as the human body mass center (m_c, n_c). The (X_d, Y_d) is changed dynamically having always the meaning of the mean of human points that belongs to the same line. So, the above method can be characterized as dynamic mean method.

```
Dynamic Mean Method
 $X_d = m_c, \quad Y_d = n_c$
 repeat {
     $X_d = X_d - 1, \quad Y_d = E_{(X_d, y) \in Man}(y)$
 } until $(X_d - 1, Y_d) \in Background$
```

The head region will be placed in a rectangle defined by the point (X_d, Y_d) as the end of the head. The maximum height and width of the head are proportional to the human height which can be easily estimated by the silhouette. The two shoulder positions, that will define the down limits of the head region, are determined by the first local minimum of the left and right horizontal silhouette projections.

The main body region is computed using an iterative algorithm similar to the end point of head estimation method. The maximum height and width of the main body are proportional to the human height. Using the human boundary, it can be determined if the arms can be distinguished from the main body (visible arm) and if the legs are distinguished. The rule is the following: if the proportion of the boundary pixels whose the closest background pixel is on the right, computed in an area where is located the left arm, exceeds a threshold, then the left arm can be distinguished from the main body. The above value is the ratio between the red pixels and the red plus green pixels of Figure 1. This knowledge helps in definition of the main body limits.

The legs and arms regions estimation can be done in the same time. An initial approximation is computed using the mass center of the human and the border point D that discriminates the left and right leg. Figure 2 shows an example of

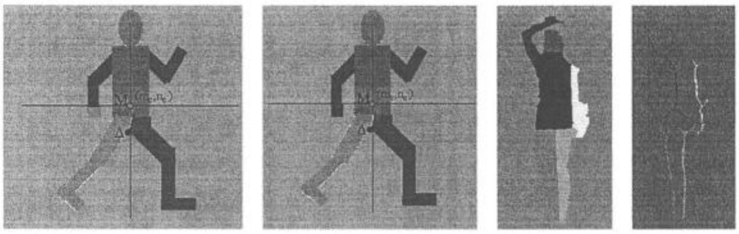


Fig. 2. The initial approximation and the final segmentation result (figures on the left). Human parts and their skeletons (figures on the right)

the initial segmentation. It is possible that more regions than six, that is the number of the human members, will be created. For these cases, we determine first which are the fault regions using the surface and the mass center of the regions. Next, we join the faults regions to the regions which have the most common boundary points with the faults regions. In the Figure 2, the false left leg region is correctly classified to left arm member.

2.2 Major Human Points Estimation

In the second stage, the 18 major human points are estimated using the segmented to human members silhouette. The method uses the skeleton of each human member in order to reduce the computation complexity, as the joint points are located to skeleton points. The skeleton is defined as the set of points whose distance from the nearest boundary is locally maximum. An example of human members skeletons are shown in the Figure 2. The points are computed sequentially. The easier defined points are computed first decreasing the search space of the others.

First, the center of the head is computed as the mass center of the head region. The neck point is defined as the mean of the boundary between head region and main body region. The two shoulders points are computed by minimizing an appropriate function F. The function domain is an isosceles triangle whose vertex is the neck point and its base vertexes are the two shoulders points. The function is minimized when the triangle base is maximized and the triangle height is minimized at the same time. The 18 major points formulations are defined in Figure 3. The points (9), (10) of the main body are computed using the main body height and width. The points (11), (15) of the main body are defined by the mean of boundary between main body region and left or right leg region respectively. The ankle point is computed first. We compute the farthest point K of skeleton points from point (9) using one line segment that should belongs to silhouette. The ankle point A is defined as the farthest point, of the not visited skeleton points, from K using one line segment. The knee point K is estimated by minimizing the function $G(X)$ which is defined by the following equation. Let F be the point (9) of the main body. Let the function $d(X, AF)$ be the minimum distance of point p from the line segment AF. $G(X) =$

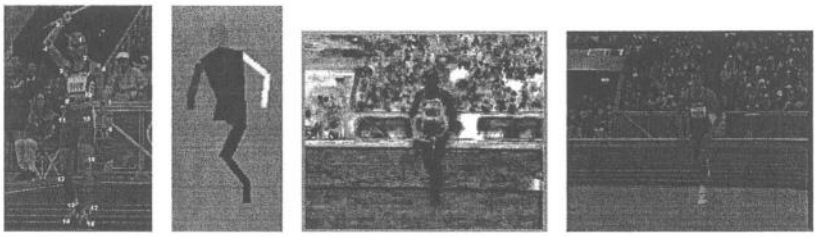


Fig. 3. The 18 major human points, the 2D human model, the left leg distance image of the right image

$(|XF|-|XA|)^2 - 0.2\cdot d^2(X,AF)$. If the point X *is* located close to the middle of AF and close to the knee angle at the same time, then the function G will be minimized. Finally, the end of leg point E is computed using the knee and ankle points. The E point search area is the left leg skeleton points which are close to the ankle point. The E point is defined as the skeleton point, whose distance from the point K is maximum. In each arm, we have to compute two points, the elbow point and the end of arm point. The computation of the left arm points is similar to the computation of the right arm points.

3 Human Tracking Method

3.1 Color Distance Images

The color distance image is an important measure used in human tracking method. We compute them for every part and subpart. The gray value at a given pixel of the color distance image corresponds to the minimum distance between the color of the considered pixel and the nearest color among the dominant colors of the part or subpart, where the point is assumed to belong. The dominant colors are computed once during the initialization process, where the segmentation of the body parts can be user guided. In Figure 3 an example of color distance image is illustrated.

3.2 Threshold – Visibility Computation

Our method is based on color segmentation using the color distance image. Hence a threshold is needed. In addition we define a visibility measure for evaluating the ability of a given part detection in its changing background. In this section, we examine the automatic threshold and visibility measure computation. These computations are executed for every new frame of the sequence. The algorithm is related to color information, so the color distance images are used. We need to update these values, because the image colors probably could be changed (shadings, color effects). The threshold and the visibility information computation are unsupervised. These methods use the dominant colors and the color distance images of the part or subpart. The threshold corresponds to the maximum value of the color distance image for a pixel belonging to the considered part. First, the histogram (H) of the color distance image is computed. The threshold T_g and the visibility measure V_g for a part or subpart g depend on the distance distribution.

Let I_g be the Color Distance Image of part g and S be the part surface in pixels. Let us define two thresholds W_1 and W_2, so that the pixels of the color distance image that have lower value than W_1, probably correspond to pixels of the foreground, else, if they have higher value than W_2, probably correspond to pixels of the background. So, the maximum of T_g is W_2 and its minimum value is W_1. A constant c_g, with values between 2 and 4, depending on the part size, is used in the definition of T_g. Let us define first the cumulative empirical probability: $f(n) = \sum_{k=0}^{n} H(k)$. Then the two limit values are defined as follows:

$$W_1 = min_t(t : f(t) > S) \quad W_2 = min_t(t : f(t) > c_g \cdot S) \tag{1}$$

Finally, we define the threshold value T_g automatically as the mean of W_1 and W_2. The visibility measure (V_g) is relevant to the ability of a human part detection (foreground/background contrast). It is defined separately for every part. The visibility measure could take any real positive value. If a part has null visibility measure, then it will be almost impossible to extract it.

The main steps of the visibility measure (V_g) computation are the following. At first two average values, v_1, v_2, are computed,

$$v_1 = E(n : f(n) < S) \quad v_2 = E(n : ((c_g - 1) \cdot S < f(n) < c_g \cdot S)) \tag{2}$$

These values play the same role as the W_1 and W_2 for the threshold determination. The visibility measure of a part is proportional to the difference between v_1, v_2 (Equation (3)). When v_1 takes values higher than 6, then the part in the current image will probably differ from its dominant colors. So, the part visibility measure should be decreased (third factor in Equation (3)). The second factor in Equation (3) has similar meaning as the third.

$$V_g = (v_2 - v_1) \cdot (1 - e^{-(\frac{v_2}{9})^2}) \cdot e^{-(\frac{v_1}{9})^2} \tag{3}$$

If a part has visibility measure more than 2, from our experience we conclude that it will have high contrast against the background.

3.3 Initialization, Prediction

The initialization step is executed only in the first image of the sequence. The user has to give the position of the predefined 18 major human joint points (Figure 3) in the first image of the sequence.

The algorithm is the following. Using these positions in the first image we construct a 2D shape of the human to get the color information of every body part and subpart. The 30 dominant colors of each part and the 15 dominant colors of every subpart are computed using the Self-Organizing Map (SOM) neural network. An appropriate coordinate system is used in order to utilize the

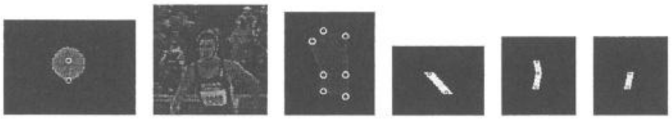


Fig. 4. The head mask, a head tracking result, the body mask and the 3 masks that are used in arm tracking

human motion, with the center of the head as the coordinate center. Relative coordinates are used for the main body points (7 points). We use angle and amplitude, which determine the positions of joints (10 points). The angle and the amplitude of each joint are computed by using its neighbor joint which is closer to the main body.

The prediction model is described following. This model is used in every image of the sequence. The prediction of the position for the 18 major points is computed using the positions of these points in the two previous images of the sequence. The prediction of the 7 main body points (in the relative coordinate system of the head) is given by the positions of these points in the previous image. For the other 10 joint points, we use a fixed coefficient first-order filter to predict the angle in a relative coordinate system. Concerning the prediction of the part length it is likely to be near that in the previous frame.

3.4 Head Tracking

At first, we search for the head because it is the most visible part and it usually has specific color and texture. The head is defined in the 2D model by two points, the center of the head and the neck point. Because of the 2 degrees of freedom of every point, the complexity of the head tracking would be $O(N^4)$. We use a sequential algorithm to reduce this complexity to $O(N^2)$.

According to the 2D model the position of the center of the head determines the translation and the position of the neck point determines the rotation and the size of the head. In most of the cases the motion due to translation is stronger than that due to rotation and scaling, so the points could be computed sequentially. First, we compute the center of the head and then the neck point. We use the same minimization criterion for the two cases.

Instead of minimization we use a matching function which is maximized. This function is defined in the color distance image of head using the head mask (Figure 4). The mask is defined by the center head point and the neck point, as in the 2D model. The mask consisted of an ellipse, covering a semi-ellipse. The semi-ellipse corresponds to the head background and the ellipse corresponds to the head.

First, we shift the mask in order to find the center of the head. The matching function is maximized when the head pixels belong to the ellipse (points with low color distance) and the background head pixels belong to the semi-ellipse (points with high color distance). This position will give us the center of the head. The I_h is the head color distance image. Let F_h be the set of the head points and B_h be the set of head background points according to the head mask.

The function h is defined by the percentage of the head points according to the mask, which have color distance lower than T_{head}. The function b is defined by the percentage of the background points according to the mask, which have color distance lower than T_{head}. When h or $1 - b$ becomes low (close to 0) then the matching is extremely wrong. Otherwise, when h and $1 - b$ are close to 1 the matching is satisfactory.

$$f_h = h \cdot (1 - b) \tag{4}$$

By experiments, the best results are achieved when h is high and b is low at the same time. This is done when function f_h is maximized. If we assume that the head model and background model are independent, then the function f_h is proportional to the possibility of matching both of them.

The neck point is computed following the same procedure. The mask is rotated and scaled in order to maximize the matching function. The point that maximizes the matching function will be the neck point.

3.5 Main Body Tracking

In this section we examine the method of 6 main body points tracking (the neck point was computed in the head tracking method). We use relative coordinates towards to the center of the head because the relative coordinates change slowly. We use a sequential-iterative algorithm in order to reduce complexity. The algorithm is described in the following. A point p is chosen among the 6 main body points. The point p is moved in a small region around its position. The new position of p is the position which maximizes the matching function f_b. The algorithm is completed when the points stop oscillating or the number of iterations exceeds a prefixed number of iterations.

The mask that we use is shown in Figure 4. The heptagon points correspond to the main body pixels. The two triangles correspond to the main body background pixels. The heptagon and the two orthogonal triangles are defined by the 7 points of the main body. The matching function (f_b) that we use is similar with the matching function in the head method.

3.6 Leg Tracking

In this section we are going to examine the method for tracking the three leg points. The algorithm is the same for the left and the right leg. We use a sequential algorithm because the normal computation would have $O(N^6)$ complexity. The steps of the algorithm are the following. First, the thigh angles are tracked. Next, shoes and knee angles are tracked. Finally, knee points are tracked.

Thigh Angles Tracking The first step of the method is similar to that of the previous methods. We search the left and the right thigh angle in a search space that is defined by the predictions of the angles. We use the masks of

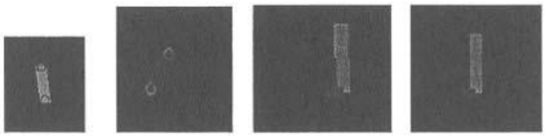


Fig. 5. The mask of the left thigh angle, the right thigh angle, the left leg, the right leg and the 3 masks that are used in arm tracking

Figure 5 where the thigh point and knee point are represented as circles. The Color matching error is minimized when yellow pixels of the mask correspond to pixels with low left leg Color Distance (left leg) and the red pixels correspond to pixels with high left leg Color Distance (left leg background).

Shoes and Knee Angles Tracking The second step of the method uses the visibility information to combine the shoe color information and the second leg subpart color information. In this section we compute the position of the 2 points of the shoes and the angle of the knee using the masks of Figure 5. The color matching error is minimized when the green pixels of the mask correspond to pixels with low Left leg Color Distance (left leg), the blue pixels of the mask correspond to pixels with low Left shoe Color Distance (left shoe) and the red pixels correspond to pixels with high left leg Color Distance (background). First, we search for the position of the ankle point and the angle in a long area with high searching step. The next step is to search the above features with a lower searching step using the previous method results. Finally, we search for the leg member (the angle and the amplitude of the last leg point), which is changed slowly. If the minimum matching error is lower than a threshold, this means that the shoe is hidden. Let f_s be the matching function (f_s). The I_s is the shoe distance image and the I_l is the second subpart of leg color distance image. Let F_l be the set of the leg points and F_s be the set of shoe points and B_l be the set of leg background points, according to the leg mask. Let l_2 be the second subpart of leg.

The function S is defined by the percentage of the shoe points, according to the mask, which have color distance lower than T_{shoes}. The function L_2 is defined by the percentage of the leg subpart (left or right) points, according to the mask, which have color distance lower than T_{l_2}. The function B is defined by the percentage of the background points, according to the mask, which have shoe color distance lower than T_{shoes} and second subpart of leg color distance lower than T_{l_2}. When B becomes low (close to 0) then the matching is satisfactory. Otherwise, when B is close to 1, the matching is extremely wrong. Function H is defined by the linear combination of S and L_2.

$$H = \frac{V_{shoes} \cdot S + V_{l_2} \cdot L_2}{V_{shoes} + V_{l_2}} \tag{5}$$

So, H contains the matching information from two models synthesis, while B contains the matching information from one model (background model). A func-

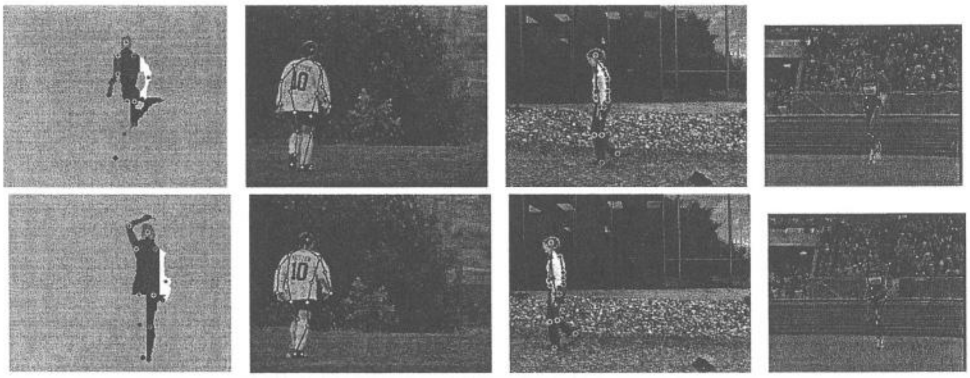


Fig. 6. Final results of human silhouette analysis and of 2D human tracking

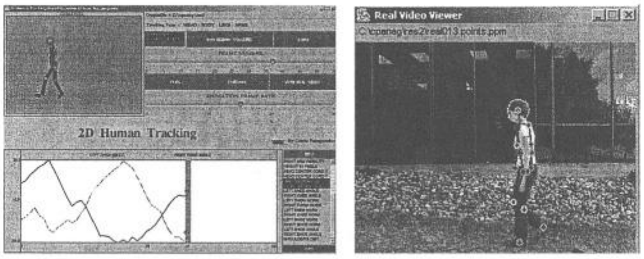


Fig. 7. The Java interface and the Java Real Video Viewer

tion f_s, similar to f_h (Eq. (4)) is maximized, $f_s = H^2 \cdot (1 - B)$. The last step is the knee points tracking. This step is trivial because the knee angle and the thigh angle have been estimated.

3.7 Arm Tracking

In this section, we examine the arm tracking algorithm. The method is the same for left and right arm. We have to compute two points in every arm.

The method will track the arm, if the arm and the main body are not superposed. If the arm is hidden, the method will detect that by high matching error. The algorithm has the following steps. First, the shoulder angle is tracked. Next, the elbow angle and the shoulder point - elbow point distance are estimated. Finally, the elbow point - arm member point distance is estimated. Figure 5 shows the three masks used. The Color matching error is minimized when white pixels of the masks correspond to pixels with low arm Color Distance (arm) and red pixels correspond to pixels with high arm Color Distance (arm background).

4 Experimental Results

The human body members recognition and 18 major human points estimation methods have been tested in more than 100 noisy silhouettes. The algorithm succeeded to recognize the human body members in all the images with high accuracy. The mean error in head, main body major points estimation is about 2% of the total human height, while the arms - legs major points are estimated with less than 200% of the previous error. In Figure 6, some results of human silhouette analysis are shown. The method was implemented in C. The complexity of the total algorithm is $O(N)$ (N = # pixels of the human). If we did not use skeletons we would have a complexity of $O(N\sqrt{N})$.

The human tracking algorithm has been tested in many video sequences of walking or running people. The quality and analysis of images that we used as input varied from CIF format images with low quality (MPEG1) to high quality images (MPEG2). The complexity of the algorithm is $O(N^2)$ (N = # pixels of the human). The frame processing rate was about 1 frame per second in Pentium III 800 MHz, using 288 × 320 images. The results include the position of the 18 major human points in every image of the sequence and 2D segmentation into human parts and subparts using the 2D human model. We examined how the unstable background, the video quality and the image size can affect the final result.

In Figure 6, some results for three sequences are shown. In two of these sequences, the person walked in an unstable background (the camera was moving). In the third sequence, where the person was running and the camera was moving to track him, the arms were not tracked because of their low imaging quality. The algorithm accurately tracked the human in the above sequences.

The method was implemented in C. Also, a Java based interface has been developed for our experiments (Figure 7). The user has many options, like load old tracking results, execute human tracking algorithm, selection of features that will be plotted, viewing of real sequence images, etc.

5 Discussion

In this paper, we have proposed an effective sequential algorithm for human body members recognition and 18 major human points estimation using silhouette. If the estimation of a major point is wrong then the other major points, that use the false point, will be computed wrong. As the results have shown, this is impossible to happen because the points, that are computed first, are the more visible points and its estimation decrease the searching area and the computation cost of the other points.

Also, we have proposed a fast and effective algorithm for human silhouette tracking in video sequence. The method needs the position of the 18 major human points in the first image. The method also detects if any human part is hidden (arm - body, shoes) by using high matching errors. The head tracking is always correct, because of the head's simple shape and motion. The head is usually the most visible human part. The main body is more difficult to be tracked than the head, because, usually, there are more colors in the main body or the arms may cover it. The most difficult parts to be tracked are the arms and legs, because of their complex motion. In many frames they are hidden and their pixels are smoothed because of motion.

The developed method could be expanded to track more than one person. We can also avoid the initialization step by using background subtraction methods and independent motion detectors. The above methods give us a bitmap image of a moving object (silhouette). Using the extraction method of the 18 major human points from the silhouette we can get the initial position of 18 major human points. One possible extension of the developed method is 3D tracking and the extraction of motion parameters using a stereoscopic system. Another extension can be human-activity recognition (walking, sitting, running, etc). Security system and statistics analysis of human motion systems could be based on our method.

References

[1] J. K. Aggarwal, Q. Cai. Human Motion Analysis: A Review, *Computer Vision and Image understanding*, vol. 73, no. 3, pp. 428-440, March 1999.

[2] J. C. Cheng, M. F. Moura. Capture and Representation of Human Walking in Live Video Sequences, *IEEE Trans. on Multimedia*, vol. 1, no. 2, pp. 144-156, June 1999.

[3] I. Haritaoglu, D. Harwood, D. Davis. W^4: Real-Time Surveillance of People and Their Activities, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809-830, Aug. 2000.

[4] I. Kakadiaris, D. Metaxas. Model-Based Estimation of 3D Human Motion, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1453-1459, Dec. 2000.

[5] J. Puzicha, J. Buhmann, Y. Rubner, C. Tomazi. Empirical Evaluation of Dissimilarity Measures for Color and Texture, *1996 IEEE*, 1999.

[6] L. Wang, W. Hu, T. Tan. Recent developments in human motion analysis, *Pattern Recognition* , vol. 36, no. 3, pp. 585-601, 2003.

[7] L. Wang, H. Ning, T. Tan. Fusion of Static and Dynamic Body Biometrics for Gait Recognition, in *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 14, no. 2, pp. 149-158, Feb. 2004.

Image Cues Fusion for Object Tracking Based on Particle Filter

Peihua Li and François Chaumette

IRISA/INRIA Rennes, Campus Universitaire de Beaulieu
35042 Rennes Cedex, France
pli@irisa.fr

Abstract. Particle filter is a powerful algorithm to deal with non-linear and non-Gaussian tracking problems. However the algorithm relying only upon one image cue often fails in challenging scenarios. To overcome this, the paper first presents a color likelihood to capture color distribution of the object based on Bhattacharry coefficient, and a structure likelihood representing high level knowledge regarding the object. Together with the widely used edge likelihood, the paper further proposes a straightforward image cues fusion for object tracking in the framework of particle filter, under assumption that the visual measurement of each image cue is independent of each other. The experiments on real image sequences have shown that the method is effective, robust to illumination changes, pose variations and complex background.

1 Introduction

Probabilistic object tracking in image sequences has widespread applications in human-computer interaction, surveillance, visual servoing and biomedical image analysis. It has therefore been an active research topic in computer vision for over a decade.

Among probabilistic object tracking algorithms, particle filter has attracted considerable attention in recent years, because of its powerful ability to deal with general non-linear and non-Gaussian problems [1, 12]. In the framework of particle filter, one of the most important parts is the likelihood function (i.e., the measurement model). Some researchers are devoted to present different likelihoods for effective tracking, including those based on edge [1], or color [2, 3, 4]. Although particle filter has proven successful in dealing with object tracking, visual measurement dependent only on one image cue is not sufficient, and tracking failures often occur in complex scenarios [5, 6, 7]. Several factors can result in this consequence, such as significant illumination changes in the environment, pose variations of the object and non-linear deformations of shapes, in addition to noise and dense clutters in complex background.

Some researchers have recently made efforts to try to solve the problem. Wu et al. [6] present a novel particle filter, as an approximation of a factorized graphical model, in which shape and color samples are interactively drawn from each others' measurements based on importance sampling. The algorithm is novel and tested on many real image sequences, but the effects are not so much satisfying. Pérez et al. [7] combine color information and motion information into tracker, in addition, they also consider the problem of multi-modality fusion, such as that of sound and image. While combining multiple information into particle filter, what one should consider carefully is how to integrate them in an effective way. Both Wu et al. and Pérez et al. employ methods similar to importance sampling introduced in [8]. Wu et al. first draw samples from color information based on importance sampling and evaluate shape probability. After that they make importance sampling on shape information and evaluate the color likelihood. Pérez et al. adopt similar method, either first drawing samples from color cue and then evaluating motion cue, or sampling from motion cue and then performing evaluation of color cues. This way, different image cues are applied to the algorithm sequentially, instead of simultaneously. While it may help improve efficiency, the inaccurateness, or, the most worst of all, failure of one cue, will heavily affect the others. In general, it is not desirable that evaluation of one cue will affect that of others.

We present in the paper a multiple cues fusion method in the framework of particle filter, in which all of the image cues are evaluated simultaneously on each discrete sample (particle). Three different likelihood functions, representing respectively three different image cues, are considered in the paper. We assume that the measurement of each image cue is independent of each other, and they are integrated to contribute to the overall measurement density. This way, different image cues are fused simultaneously and the failure of one image cue will not affect the evaluation of the other cues. The image cues we are interested in are edge information, color distribution and/or structural information of a specific class of objects, e.g., human faces. Note that our method of image cues integration is similar to that introduced in [9]. There exist, however, great differences between theirs and ours. In [9], Spengler et. directly extend the approach introduced in [10] from single hypothesis to multiple hypotheses, in which the image cues are evaluated on the whole image map (so they will have to confine their algorithm in the small image, say, 90*72, just like in [10]), and then different cues are weighted and added. In contrast, our image cues are only evaluated on particles. In addition, the modelling of image cues are also quite different.

The remainder of the paper is structured as follows. Section 2 introduces the generative model for object tracking, and then presents the three likelihood functions involved in the algorithm. Section 3 describes the particle filter based tracking algorithm. Section 4 makes experiments to validate the algorithm. Section 5 contains concluding marks.

2 Generative Model for Object Tracking

2.1 Shape Model

Following that described in [11], the contour is parameterized as a B-spline curve, for a set of B-spline basis is general and flexible in representing different (complex) shapes, and in controlling the degree of continuity. Specifically, the tracked objects are modelled as follows

$$\mathbf{r}(s,t) = \begin{bmatrix} x(s,t) \\ y(s,t) \end{bmatrix} = \begin{bmatrix} \mathbf{B}(s)^{\mathrm{T}} & 0 \\ 0 & \mathbf{B}(s)^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{Q}^x(t) \\ \mathbf{Q}^y(t) \end{bmatrix} \tag{1}$$

where $\mathbf{B}(s) = [b_0(s) \quad \cdots \quad b_{J-1}]^T$, for$0 \leq s \leq S$, $b_i(s)(0 \leq i \leq J-1)$ is the *i*th B-spline basis function, $\mathbf{Q}^x$ is a column vector whose unit consists of x coordinates of all the control points and similarly with $\mathbf{Q}^y$ (the time index t is omitted hereafter for simplicity), and L is the number of spans. The configuration of the spline is restricted to a shape-space of vectors $\mathbf{X}$ defined by

$$\begin{bmatrix} \mathbf{Q}^x \\ \mathbf{Q}^y \end{bmatrix} = \mathbf{W}\mathbf{X} + \begin{bmatrix} \bar{\mathbf{Q}}^x \\ \bar{\mathbf{Q}}^y \end{bmatrix} \tag{2}$$

where $\mathbf{W}$ is a shape matrix whose rank is less than $2J$, and $\bar{\mathbf{Q}} = [\bar{\mathbf{Q}}^x \; \bar{\mathbf{Q}}^y]^T$ is a template of the object. Below are two kinds of shape spaces used in the paper, the first for head tracking and the second for person tracking

$$\mathbf{W} = \begin{bmatrix} 1 & 0 & \bar{\mathbf{Q}}^x \\ 0 & 1 & \bar{\mathbf{Q}}^y \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} 1 & 0 & \bar{\mathbf{Q}}^x & -\bar{\mathbf{Q}}^y \\ 0 & 1 & \bar{\mathbf{Q}}^y & \bar{\mathbf{Q}}^x \end{bmatrix} \tag{3}$$

2.2 Dynamical Model

The motion equation of the state in the shape space is modelled as the multi-dimensional second order auto-regression (AR) process, which generally can be seen as the discretized form of a continuous stochastic second order dynamic system [11]. This multi-dimensional AR process may be regarded as the direct extension of a 1D case. Define

$$a_1 = -\exp(-2\beta\tau)$$
$$a_0 = 2\exp(-\beta\tau)\cos(\omega\tau)$$
$$b_0 = \sqrt{1 - a_1^2 - a_0^2 - 2\frac{a_1 a_0^2}{1 - a_1}}$$

Define also the damping coefficient β, the oscillation period ω, and the sampling period of the system τ. Then the 1D AR process has the following form:

$$x_k = a_1 x_{k-1} + a_0 x_{k-2} + b_0 \nu \tag{4}$$

where ν is one dimensional Gaussian i.i.d. noise. It is desirable, in practice, to model the translation and the shape variations of the contour separately, so the 1D AR process is extended respectively to two complementary subspaces of the shape space: translation subspace and deformation subspace. Then the multi-dimensional motion model can be represented as below

$$\mathbf{X}_k = \mathbf{A}_1 \mathbf{X}_{k-1} + \mathbf{A}_0 \mathbf{X}_{k-2} + \mathbf{B}_0 \mathcal{V} \tag{5}$$

2.3 Observation Model

The observation model $p(\mathbf{Y}_k|\mathbf{X}_k)$ concerns in the paper three kinds of image cues: edge information of the contour, weighted color histogram and structural information outputted from boosted detector, which are represented respectively, $p_c(\mathbf{Y}_k|\mathbf{X}_k)$, $p_s(\mathbf{Y}_k|\mathbf{X}_k)$ and $p_e(\mathbf{Y}_k|\mathbf{X}_k)$. Given the predicted target state, we assume that the observation procedure of the three cues are independent of one another, so the overall measurement density has the following form

$$p(\mathbf{Y}_k|\mathbf{X}_k) = p_c(\mathbf{Y}_k|\mathbf{X}_k)p_s(\mathbf{Y}_k|\mathbf{X}_k)p_e(\mathbf{Y}_k|\mathbf{X}_k) \tag{6}$$

In practice, the log-likelihood of Equ. (6) is evaluated, and the multiplications thus become sum on the right side of the above equation. It can be seen that if one cue fails, its contribution to the overall measurement density becomes negligible.

Color Likelihood We define a likelihood to measure the confidence of the color similarity of a candidate to the target, which is based on the metric introduced in [13]. In the likelihood function, both target and candidate distribution are represented by weighted multi-channel color histogram: $\hat{\mathbf{q}} = \{\hat{q}_u\}$ with $\sum_{u=1}^{N_c} \hat{q}_u = 1$ for target, and $\hat{\mathbf{p}}(\mathbf{x}) = \{\hat{p}_u(\mathbf{x})\}$ with $\sum_{u=1}^{N_c} p_u = 1$ for the candidate, where $\mathbf{x}$ is the center of the candidate ellipse, and $u = 1, \ldots, N_c$ denote the bins of the histogram. Denote $\mathbf{z}_i$ $i = 1, \ldots, M$ the pixel locations of one candidate (the ellipse which best fits the discrete sample), and $\mathcal{K}(\cdot)$ the weighting function that has the following form

$$\mathcal{K}(\mathbf{z}) = \begin{cases} c(1 - \|\mathbf{z}\|^2) & \text{if } \|\mathbf{z}\| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The value of each weighted histogram bin u for the candidate distribution can be expressed as

$$\hat{p}_u(\mathbf{x}) = \frac{\sum_{i=1}^{M} \mathcal{K}(\|\frac{\mathbf{x}-\mathbf{z}_i}{\mathbf{h}}\|^2)\delta(b(\mathbf{z}_i) - u)}{\sum_{i=1}^{M} \mathcal{K}(\|\frac{\mathbf{x}-\mathbf{z}_i}{\mathbf{h}}\|^2)} \tag{8}$$

where $\mathbf{h}$ is the radius of a candidate region, $b(\mathbf{z}_i)$ is a function which associates to the pixel at location $\mathbf{z}_i$ the index $b(\mathbf{z}_i)$ of the histogram, and $\delta(\cdot)$ is the Kronecker delta function. The metric to measure the similarity of the target and candidate is

$$d(\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{x}) = \sqrt{1 - \rho(\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{x}))} \tag{9}$$

where $\rho(\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{x}))$ is the Bhattacharyya coefficient that has the following form

$$\rho(\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{x})) = \Sigma_{u=1}^{N_c} \sqrt{\hat{p}_u(\mathbf{x})}\sqrt{\hat{q}_u} \tag{10}$$

Upon this, we define the corresponding likelihood as follows

$$p_c(\mathbf{Y}_k|\mathbf{X}_k) = \frac{1}{\sqrt{2\pi}\sigma_c} \exp - \frac{1 - \rho(\hat{\mathbf{q}}, \hat{\mathbf{p}}(\mathbf{x}_k))}{2\sigma_c^2} \tag{11}$$

In practice we get the ellipse that best fits the contour of each particle, on which the likelihood is evaluated. The target distribution is achieved via a face detector when for head tracking. Otherwise, it is manually set.

Structure Likelihood The structure likelihood is aimed at representing the high level knowledge of the object and is achieved via machine learning algorithm. In the paper we are interested in boosted multi-view face detector [14,15]. A total of five different views are considered in the paper: frontal view, left and half-left profiles, and right and half-right profiles. For efficiency, two levels are involved: the first level concerns a cascaded face detector trained on all training examples, which contains non-face examples and face examples in five different views. The test image region which only pass the first level will continue to try to pass the second level. Five different cascaded detectors are in the second level which are responsible for detections of faces which may be in different views.

A cascaded detector implicitly assumes a certain form for the underlying probability distribution [16]. Define N_s the total number of layers in the detector, and $1, \cdots, n_s$ the layers the detection process passed, in which the output is above the relevant threshold for the input from the test rectangle corresponding to the particle. In our implementation, we assume for simplicity that the likelihood of the particle is related to n_s/N_s. More precisely, we define the structure likelihood as

$$p_s(\mathbf{Y}_k|\mathbf{X}_k) = \frac{1}{\sqrt{2\pi}\sigma_s} \exp - \frac{1 - n_s/N_s}{2\sigma_s^2} \tag{12}$$

The face detection is performed within the circumscribed rectangle of the minimal area for each particle. Because of the level and the cascade structure, the fast computation of features used in the detector, and the search being constrained in a small image rectangle, the evaluation of the likelihood is efficient. Furthermore, when the face is not present, most of regions will fail to pass the first level. This further reduces the computational load.

Edge Likelihood The model introduced by MacCormick is adopted for the observation of edges [12]. The measurement as regards edge is made at a finite number of points along the contours modelled as B-spline curve, and the normals to the contour at these sample points are searched for features. These normals have fixed length L and are termed measurement lines. A Canny edge detector is applied to each measurement line and the points of local maximum adopted as features. In general, a set of n_l features are detected on the measurement line indexed by $l = 1, \ldots, N_e$. The distances of these features from the contour constitute a set $z_j^{(l)}, j = 1, \ldots, n_l$. Each feature at distance $z_j^{(l)}$ from the contour could correspond to the true boundary of the object (in which case it is called an edge feature) or random visual clutter (in which case it is called a clutter feature).

We assume that only one edge feature can be detected on the measurement line. To model the observation density, some further assumptions are made

- The number of clutter features on the measurement lines of length L obeys a Poisson law with density λ.
- The density of the clutter features is uniform on the measurement line.
- The probability that the edge feature is not detected is P_0 and the probability that it is detected is $P_1 = 1 - P_0$.
- The distribution of the distance between the edge feature and the true contour location is Gaussian, with zero mean and variance σ_e^2.

From these assumptions, we obtain the following equation for the likelihood of the observation at a sample point, given the state $\mathbf{X}_k$:

$$p(l|\mathbf{X}_k) \propto P_0 + \frac{1-P_0}{\lambda} \sum_{j=1}^{n_l} \frac{1}{\sqrt{2\pi}\sigma_e} \exp - \frac{(z_j^l)^2}{2\sigma_e^2} \tag{13}$$

Assuming that the feature outputs on distinct normal line are statistically independent, the overall edge likelihood becomes

$$p_e(\mathbf{Y}_k|\mathbf{X}_k) = \prod_{l=1}^{N_e} p(l|\mathbf{X}_k) \tag{14}$$

3 Image Cues Fusion for Contour Tracking Based on Particle Filter

Target tracking can be characterized as the problem of estimating the state $\mathbf{X}_k$ of a system at (discrete) time k, as a set of observations $\mathbf{Y}_k$ become available over time. The Bayesian filtering framework is based on the densities $p(\mathbf{X}_k|\mathbf{X}_{k-1})$ and $p(\mathbf{Y}_k|\mathbf{X}_k)$. The transition prior $p(\mathbf{X}_k|\mathbf{X}_{k-1})$ indicates that the evolution of the state is a Markov process, and $p(\mathbf{Y}_k|\mathbf{X}_k)$ denotes the observation density (likelihood function) in the dynamical system, in which the measurements are conditionally independent of each other given the states. The aim is to estimate recursively in time the filtering density $p(\mathbf{X}_k|\mathbf{Y}_{1:k})$, where $\mathbf{Y}_{1:k}$ denotes measurements from the beginning to the current time step k, which is described as follows:

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k}) = \frac{p(\mathbf{Y}_k|\mathbf{X}_k)p(\mathbf{X}_k|\mathbf{Y}_{1:k-1})}{p(\mathbf{Y}_k|\mathbf{Y}_{1:k-1})} \tag{15}$$

where the prediction density $p(\mathbf{X}_k|\mathbf{Y}_{1:k-1})$ is

$$p(\mathbf{X}_k|\mathbf{Y}_{1:k-1}) = \int p(\mathbf{X}_k|\mathbf{X}_{k-1})p(\mathbf{X}_{k-1}|\mathbf{Y}_{1:k-1})\mathrm{d}\mathbf{X}_{k-1} \tag{16}$$

Eq. (15) provides an optimal solution of the tracking problem, which, unfortunately, involves high-dimensional integration. In most cases involving non-Gaussianity and nonlinearity, analytical solutions do not exist, leading to the use of Monte Carlo methods.

3.1 Tracking Algorithm Based on Particle Filter

The basic principle of particle filtering (also known as the Sequential Monte Carlo algorithm) is that the posterior density is approximated by a set of discrete samples (called particles) with associated weights. For each discrete time step, particle filtering generally involves three steps for sampling and weighting the particles, plus one output step. In the sampling step, particles are drawn from the transition prior. In the weighting step, particle weights are set equal to the measurement likelihood. The outputs of the filter are the particle states and weights, used as an approximation to the probability density in state space. In the last step, particles are re-sampled, to obtain a uniform weight distribution. The detailed algorithm is presented in Fig. 1.

```
1. Initialisation
   Draw particles from the prior $p(\mathbf{X}_0^{(i)})$ to obtain a set $\{(\tilde{\mathbf{X}}_0^{(i)},\ 1/N_w),\ i = 1, \cdots, N_w\}$, let $k = 1$
2. Sampling step:
   For $i = 1, \ldots, N_w$: Sample $\mathbf{X}_k^{(i)}$ from the transition prior Eq. (5).
3. Sampling and updating step
   a. Given the particle $\mathbf{X}_k^{(i)}$, evaluate the color likelihood $p_c(\mathbf{Y}_k|\mathbf{X}_k^{(i)})$, and the
      structure likelihood $p_s(\mathbf{Y}_k|\mathbf{X}_k^{(i)})$ and the edge likelihood $p_e(\mathbf{Y}_k|\mathbf{X}_k^{(i)})$. Set
      the weight to the overall measurement density
         $\tilde{w}_k^{(i)} = p(\mathbf{Y}_k|\mathbf{X}_k) = p_c(\mathbf{Y}_k|\mathbf{X}_k^{(i)})p_s(\mathbf{Y}_k|\mathbf{X}_k^{(i)})p_e(\mathbf{Y}_k|\mathbf{X}_k^{(i)}),\ i = 1, \ldots, N_w$
   b. Normalize the particle weights:
         $w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{j=1}^{N_w}},\quad i = 1, \ldots, N_w$
4. Output step and mode update
   Output a set $\{(\mathbf{X}_k^{(i)},\ w_k^{(i)}),\ i = 1, \cdots, N_w\}$ of particles that can be used
   to approximate the posterior distribution as $p(\mathbf{X}_k|\mathbf{Y}_{1:k}) \approx \sum_{i=1}^{N_w} w_k^{(i)}\delta(\mathbf{X}_k - \mathbf{X}_k^{(i)})$,
   and the system mean as the tracking result $\bar{\mathbf{X}}_k \approx \sum_{i=1}^{N_w} w_k^{(i)}\mathbf{X}_k^{(i)}$
5. Selection (resampling) step
   Resample the particles $\{(\mathbf{X}_k^{(i)},\ w_k^{(i)})\}$ with probability $w_k^{(i)}$ to obtain $N$
   i.i.d random particles $\{\tilde{\mathbf{X}}_k^{(i)}, 1/N_w\}$, approximately distributed according to
   $p(\mathbf{X}_k|\mathbf{Y}_{1:k})$
6. $k = k + 1$, go to step 2.
```

Fig. 1. The tracking algorithm based on particle filter

Table 1. Summary of parameters used in the paper

Symbol	Meaning	Value
β	Damping coefficient	2.0 and 10.0 [a]
ω	Oscillation period	0.0
P_0	Probability that edge feature is not detected	0.2
σ_e	Std variace of edge likelihood	$\sqrt{2}$
σ_c	Std variance of color likelihood	5
σ_s	Std variance of structure likelihood	$\sqrt{15}$
N_e	Number of measurement lines in edge detection	40 and 70 [b]
N_c	Number of histogram bins	$32 \times 32 \times 32$ and $16 \times 16 \times 16$[c]
N_s	Number of face detector layers	25

[a] The damping coefficient of translation shape space is 2.0, while that of deformation space is 10.0.
[b] The number of measurement lines is 40 for head tracking, and 70 for person tracking.
[c] The number of histogram bins is $32 \times 32 \times 32$ for head tracking, and $16 \times 16 \times 16$ for person tracking.

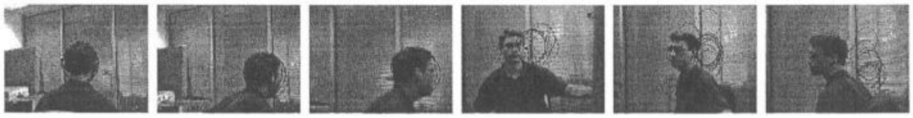


Fig. 2. Some of tracking results using edge information only. The tracking results are satisfying when strong edge can be observed, but it fails when edge information becomes weak

4 Experiments

The program is implemented with C++ on a laptop with 2.0GHz mobile Pentium CPU and 256M Memory. Table 1 summarizes parameters used in the paper. The standard variances of color and structure likelihoods are set empirically. It doesn't need to tune them very carefully: once they are set, they are favorable to all of our experiments. Unless indicated explicitly, the image sequence for head tracking is of size 256×192, and for person tracking is of size 640×240. The particles whose probability are greater than 0.1 and the mean as the tracking result, are plotted, with dashed blue color and solid red color respectively.

Fig. 2 demonstrates head tracking exploiting only edge information in a typical office environment. The tracking results are satisfying when strong edge can be observed, but it fails when edge information becomes weak.

Particle filtering based tracking dependent merely on color likelihood works well, even when illumination changes, but not significantly, as show in Fig. 3. But the algorithm collapses while encountered with significant lighting variations. Fig. 3 shows the relevant results.

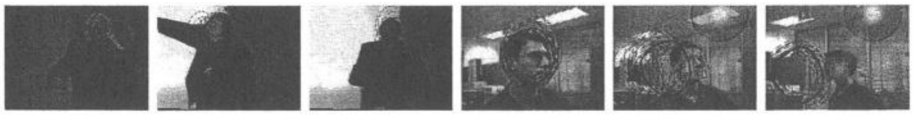


Fig. 3. Some of tracking results using color information only. Particle filter dependent merely on color likelihood works well, even when illumination changes, but not significantly, as show by the first three figures (from left to right, image size: 320 × 240). But the algorithm collapses while encountered with significant lighting variations, as shown by the last three images

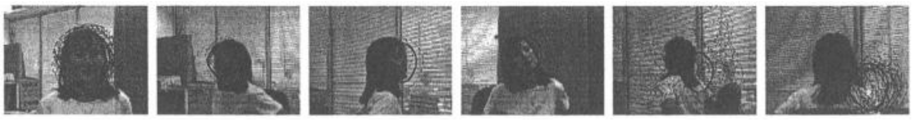


Fig. 4. Some of tracking results using color information only. The algorithm can successfully track face which undergoes pose variations and a small in-plane rotation, but will collapse when the subject turns back

Fig. 4 presents the result with algorithm which relies solely on structure information. The algorithm can successfully track face which undergoes pose variations and a small in-plane rotation, but will fail when the subject turns back.

We test our tracking algorithm fusing multiple image cues on two image sequences. The first image sequence concerns head tracking, which involves considerable lighting changes, distraction similar to skin color, agile motion of the object and complex background. Tracking depending only on one image cue or two image cues simultaneously will fail without exception. The tracking algorithm , running at about 10Hz, can well localize the target throughout the whole image sequence, making use of the three image cues simultaneously. Some tracking results are shown in Fig. 5. Note that from frame 29 to 80 there occurs considerable lighting change, which affects heavily both the color and edge of the object, as such importance sampling depending either will fail in this case.

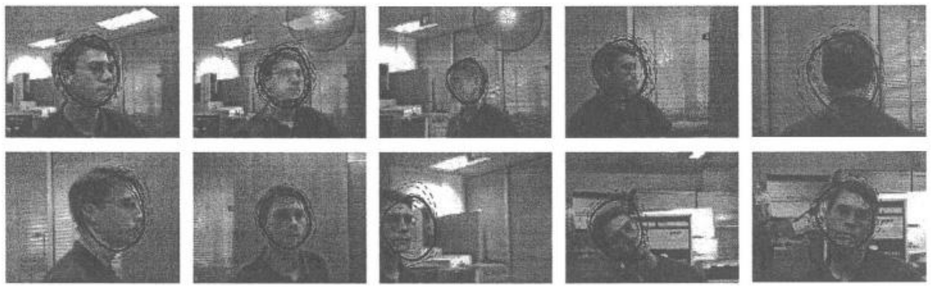


Fig. 5. Some of tracking results using three image cues

The second image sequence, recorded with a wide-len camera, is concerned with a person walking in front of the shop window [17]. The target appears at one end and walk to the other end. The following factors existed that make tracking difficult in this case: the reflections from the ground and from the opposing window; occlusions from the text on the window; complex non-linear deformation due to the walking behavior and the wide-len; the similarity of the color between the subject's clothes and the background. Tracking would have been more simple by subtracting the background image from the image sequence since the camera is fixed. We do not make use of this advantage, in order to test our algorithm in this complex scenario. The tracking results as illustrated in Fig. 6 are satisfactory, thanks to the algorithm (running at about 9Hz) that fuses of edge and color information. Notice that in the vicinity of frame 95 the tracker are heavily disturbed by the clutter in the background, but it succeeds to overcome it several frames later.

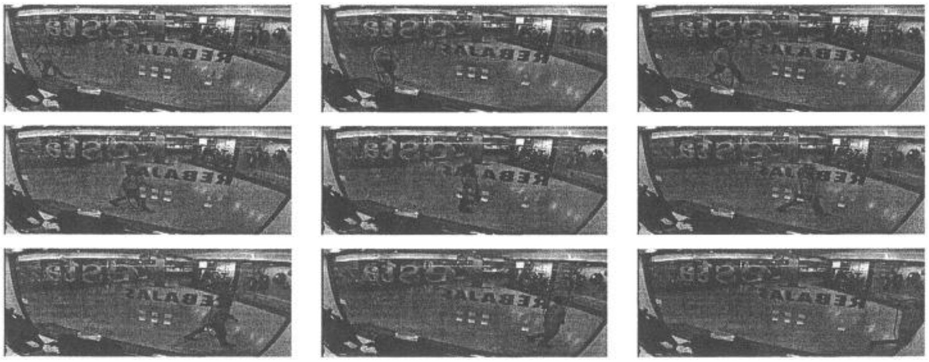


Fig. 6. Some of tracking results using edge and color information

5 Conclusions

In the paper, a tracking algorithm based on particle filter is presented which integrates multiple image cues in a probabilistic way. We first presents a color likelihood to capture color distribution of the object based on Bhattacharry coefficient, and a structure likelihood to represent high level knowledge regarding the object based on AdaBoost learning. We also consider a widely used edge likelihood. Then, under the assumption that the measurement processes related to the above three likelihoods are independent of one another, they are combined to contribute to an overall observation density. The experiments show that, in challenging environment, particle filter based tracking algorithms making use of only one image often fails. With fusion of multiple image cues, the tracker becomes much more robust to significant lighting changes, pose variations and complex background. The paper also demonstrates that the high level knowledge itself, through a likelihood built upon a boosted multi-view face detector, can be used for object tracking. It is straightforward to extend this to tracking of other class of objects, such as pedestrians and cars, which will be our future work.

Acknowledgements

Some test video streams are downloaded from the following websites:

http://vision.stanford.edu/˜birch/headtracker/

and

http://www.ece.northwestern.edu/˜yingwu/research/Tracking/.

The respective owner of the video clips is acknowledged. The first author would also like to thank Dr. Arthur E.C. Pece who helped improve the structure as well as the readability of the paper.

References

[1] Isard, M., Blake, A.: Contour Tracking by Stochastic Propagation of Conditional Density. Proc. Eur. Conf. on Comp. Vis. Cambridge UK (1996) 343–356.

[2] Nummiaro, K., Koller-Meier, E.,Van Gool, L.: An Adaptive Color-based Particle Filter. Image and Vision Computing **21**(1) (2003) 99–110.

[3] Vermaak, J. , Pérez, P., Gangnet, M., Blake, A.: Towards Improved Observation Models for Visual Tracking: Selective Adaptation. Proc. Eur. Conf. on Comp. Vis. Copenhagen Denmark (2002).

[4] Pérez, P., Hue, C., Vermaak, J., Gangnet, M.: Color-based Probabilistic Tracking. Proc. Eur. Conf. on Comp. Vis. Copenhagen Denmark (2002).

[5] Vermaak, J., Gangnet, M., Blake, A., Pérez, P.: Sequential Monte Carlo Fusion of Sound and Vision for Speaker Tracking. IEEE Int'l Conf. Comp. Vis. (2001) 741–746.

[6] Wu, Y., Huang, T. S.:A Co-inference Approach to Robust Visual Tracking. IEEE Int'l Conf. Comp. Vis. Vancouver Canada (2001) 26–33.

[7] Pérez, P., Vermaak, J., Blake, A.: Data Fusion for Visual Tracking with Particles. Proceedings of IEEE (issue on State Estimation) (2004).

[8] Isard, M., Blake, A.: ICondensation: Unifying Low-level and High-level Tracking in a Stochastic Framework. Proc. Eur. Conf. on Comp. Vis. Freiburg Germany (1998) 893–908.

[9] Spengler, M., Schiele, B.: Towards Robust Multi-Cue Integration for Visual Tracking. Machine Vision and Applications. **14**(1) (2003) 50–58.

[10] Triesch, J., von der Malsburg, C.: Democratic Integration: Self-Organized Integration of Adaptive Cues. Neural Computation **13**(9) (2001) 2049–2074.

[11] Blake, A., Isard, M.: Active contours. Springe-Verlag, Berlin Heidelberg (1998).

[12] MacCormick, M.: Probabilistic Models and Stochastic Algorithms of Visual Tracking. PhD thesis, University of Oxford (2000).

[13] Comaniciu, D., Ramesh, V., Meer, P.: Real-time Tracking of Non-rigid Objects Using Mean Shift. IEEE Int. Conf. on Comp. Vis. and Pat. Rec. Hilton Head Island South Carolina (2000) 142–149.

[14] Viola, P., Jones, M. J.: Robust Real-time Object Detection. IEEE Workshop on Statistical and Computational Theories of Vision. Vancouver Canda (2001).

[15] Li, S. Z., Zhu, L., Zhang, Z.,Blake, A., Zhang, H., Shum, H.: Statistical Learning of Multi-view Face Detection. Proc. Eur. Conf. on Comp. Vis. Copenhagen Denmark (2002).

[16] Friedman, J., Hastie, T., Tibshirani, R.:Additive Logistic Regression: A Statistical View of Boosting. The Annals of Statistics, **38**(2)(2000) 337-374.

[17] http://pets2002.visualsurveillance.org

3D Human Walking Modeling

Angel D. Sappa[1], Niki Aifanti[2], Sotiris Malassiotis[2], and Michael G. Strintzis[2]

[1] Computer Vision Center, Edifici O, Campus UAB,
08193 Bellaterra - Barcelona, Spain
angel.sappa@cvc.uab.es
[2] Informatics & Telematics Institute, 1st Km Thermi-Panorama Road,
57001 Thermi-Thessaloniki, Greece
{naif,malasiot}@iti.gr
strintzi@eng.auth.gr

Abstract. This paper presents a new approach for 3D human walking modeling from monocular image sequences. An efficient feature point selection and tracking approach has been used to compute feature points' trajectories. Peaks and valleys of these trajectories are used to detect key frames—frames where both legs are in contact with the floor. These frames, together with prior knowledge of body kinematics and a motion model, are the basis for the 3D reconstruction of human walking. The legs' configuration at each key frame contributes to tune the amplitude of the motion model. Differently than previous approaches, this tuning process is not performed at every frame, reducing CPU time. In addition, the movement's frequency is defined by the elapsed time between two consecutive key frames, which allows handling walking displacement at different speed. Experimental results with different video sequences are presented.[1]

1 Introduction

Walking motion is the most common type of human locomotion. 3D motion models are required for applications such as: intelligent video surveillance, pedestrian detection for traffic applications, gait recognition, medical diagnosis and rehabilitation, human-machine interface [1]. Due to the wide interest it has generated, 3D human motion modeling is one of the most active areas within the computer vision community.

Vision-based human motion modeling approaches usually combine several computer vision processing techniques (e.g. video sequence segmentation, object tracking, motion prediction, 3D object representation, model fitting, etc.). Different techniques have been proposed to find a model that matches a walking displacement. These approaches can be broadly classified into monocular or multi camera approaches.

[1] This work has been carried out as part of the ATTEST project (Advanced Three-dimensional TElevision System Technologies, IST-2001-34396). The first author has been supported by *The Ramón y Cajal Program.*

A multicamera system was proposed by [2]. It consists of a stereoscopic technique able to cope not only with self-occlusions but also with fast movements and poor quality images. This approach incorporates physical forces to each rigid part of a kinematics 3D human body model consisting of truncated cones. These forces guide each 3D model's part towards a convergence with the body posture in the image. The model's projections are compared with the silhouettes extracted from the image by means of a novel approach, which combines the Maxwell's demons algorithm with the classical ICP algorithm. Although stereoscopic systems provide us with more information for the scanned scenes, 3D human motion systems with only one camera-view available is the most frequent case.

Motion modeling using monocular image sequences constitutes a complex and challenging problem. Similarly to approach [2], but in a 2D space and assuming a segmented video sequence is given as an input, [3] proposes a system that fits a projected body model with the contour of a segmented image. This boundary matching technique consists of an error minimization between the pose of the projected model and the pose of the real body—all in a 2D space. The main disadvantage of this technique is that it finds the correspondence between the projected body parts and the silhouette contour, before starting the matching approach. This means that it looks for the point of the silhouette contour that corresponds to a given projected body part, assuming that the model posture is not initialized. This problem is still more difficult to handle in those frames where self-occlusions appear or edges cannot be properly computed.

Differently than the previous approaches, the aspect ratio of the bounding box of the moving silhouette has been used in [4]. This approach is able to cope with both lateral and frontal views. In this case the contour is studied as a whole and body parts do not need to be detected. The aspect ratio is used to encode the pedestrian's walking way. However, although shapes are one of the most important semantic attributes of an image, problems appear in those cases where the pedestrian wears clothes not so tight or carries objects such as a suitcase, handbag or backpack. Carried objects distort the human body silhouette and therefore the aspect ratio of the corresponding bounding box.

In order to be able to tackle some of the problems mentioned above, some authors propose simplifying assumptions. In [5] for example, tight-fitting clothes with sleeves of contrasting colors have been used. Thus, the right arm is depicted with a different color than the left arm and edge detection is simplified especially in case of self-occlusions. [6] proposes an approach where the user selects some points on the image, which mainly correspond to the joints of the human body. Points of interest are also marked in [7] using infrared diode markers. The authors present a physics-based framework for 3D shape and non-rigid motion estimation based on the use of a non-contact 3D motion digitizing system. Unfortunately, when a 2D video sequence is given, it is not likely to affect its content afterwards in such a way. Therefore, the usefulness of these approaches is restricted to cases where access in making the sequence is possible.

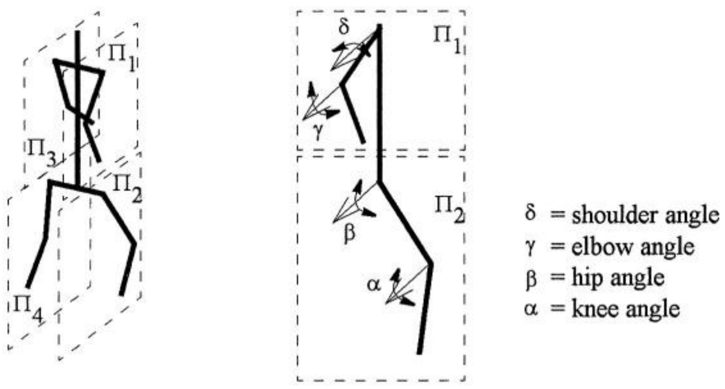


Fig. 1. Simplified articulated structure defined by 12 DOFs, arms and legs rotations are contained in planes parallel to the walking's direction

Recently, a novel approach based on feature point selection and tracking was proposed in [8]. This approach is closely related to the technique proposed in this work. However, a main difference is that in [8] feature points are triangulated together and similarity between triangles and body parts is studied, while in the current work, feature point's trajectories are plotted on the image plane and used to detect key frames. Robustness in feature point based approaches is considerably better than in those techniques based on silhouette, since silhouette does not only depend on walking style or direction but also on other external factors such as those mentioned above. Walking attitude is easier captured by studying the spatio-temporal motion of feature points.

In this paper a new approach to cope with the problem of human walking modeling is presented. The main idea is to search for a particular kinematics configuration throughout the frames of the given video sequence, and then to use the extracted information in order to tune a general motion model. Walking displacement involves the synchronized movements of each body part—the same is valid for any cyclic human body displacement (e.g., running, jogging). In this work, a set of curves, obtained from anthropometric studies [9], is used as a coarse walking model. These curves need to be individually tuned according to the walking attitude of each pedestrian. This tuning process is based on the observation that although each person walks with a particular style, there is an instant in which every human body structure achieves the same configuration. This instant happens when both legs are in contact with the floor. Then, the open articulated structure becomes a closed structure. This closed structure is a rich source of information useful to tune most of the motion model's parameters. The outline of this work is as follows. The proposed technique is described in section 2. Experimental results using different video sequences are presented in section 3. Conclusions and further improvements are given in section 4.

2 The Proposed Approach

The proposed approach consists of two stages. In the first stage feature points are selected and tracked throughout the whole video sequence in order to find key frames' positions. In the second stage a generic motion model is locally tuned by using kinematics information extracted from the key frames. The main advantage comparing with previous approaches is that matching between the projection of the 3D model and the body silhouette image features is not performed at every frame (e.g., hip tuning is performed twice per walking cycle). The algorithm's stages are fully described below together with a brief description of the 3D representation used to model the human body.

2.1 Body Modeling

Similarly to [10], an articulated structure defined by 16 links is used. However, in order to reduce the complexity, motion model is simplified and consists of 12 DOF. This simplification assumes that in walking, legs' and arms' movements are contained in parallel planes (see illustration in Fig. 1). In addition, the body orientation is always orthogonal to the floor. Thus the orientation is described by only one DOF (this degree of freedom allows us to model trajectories that are not orthogonal to the camera direction, see Fig. 11). Hence, the final model is defined by two DOF for each arm and leg and four for the torso (three for the position plus one for the orientation).

The articulated structure is represented by 16 superquadrics (Fig. 2); a complete mathematical description of this volumetric model can be found in [10].

2.2 Feature Point Selection and Tracking

Feature point selection and tracking approaches were chosen because they allow capturing the motion's parameters by using as prior knowledge the kinematics of the body structure. In addition, point-based approaches seem to be more robust in comparison with silhouette based approaches. Next, a brief description of the techniques used is given.

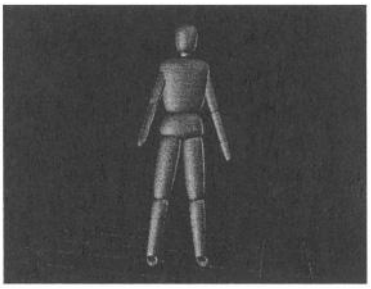


Fig. 2. Illustration of a 22 DOF model built with superquadric

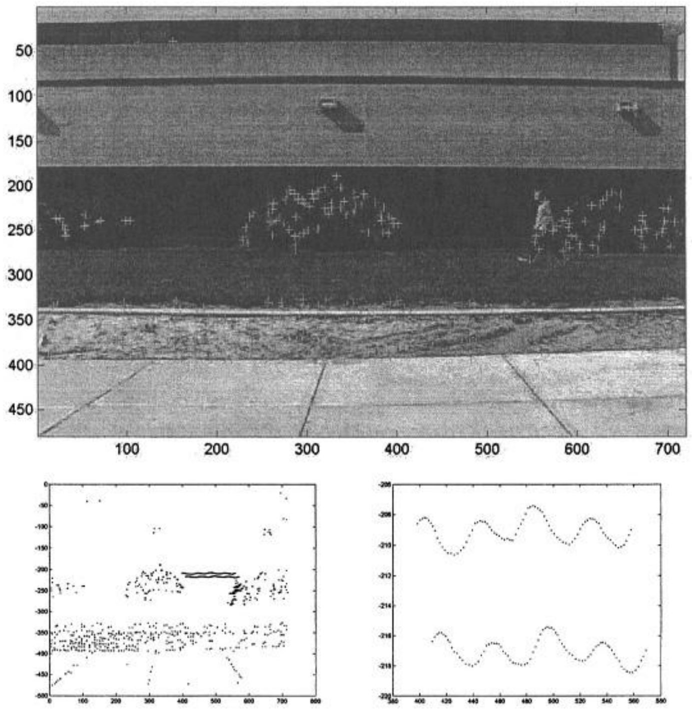
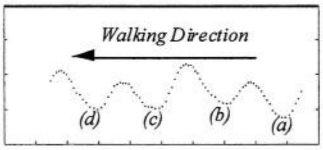


Fig. 3. *(top)* Feature points from the first frame of the video sequence used in [13]. *(bottom-left)* Feature points' trajectories. *(bottom-right)* Feature points' trajectories after removing static points

Feature Point Selection. In this work, the feature points are used to capture human body movements and are selected by using a corner detector algorithm. Let *I(x,y)* be the first frame of a given video sequence. Then, a pixel *(x,y)* is a corner feature if at all pixels in a window W_S around *(x,y)* the smallest singular value of *G* is bigger than a predefined σ; in the current implementation W_S was set to *5x5* and $\sigma = 0.05$. *G* is defined as:

$$G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

and (I_x, I_y) are the gradients obtained by convolving the image *I* with the derivatives of a pair of Gaussian filters. More details about corner detection can be found in [11]. Assuming that at the beginning there is no information about the pedestrian's position in the given frame, and in order to enforce a homogeneous feature sampling, input frames are partitioned into 4 regular tiles (*2x2* regions of *240x360* pixels each in the illustration presented in Fig. 3).



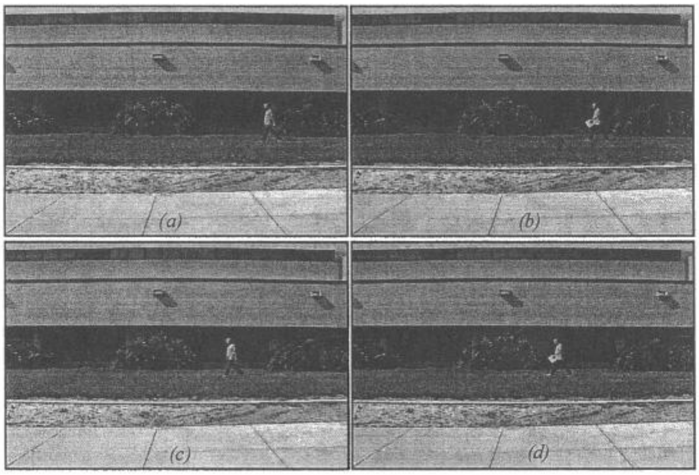


Fig. 4. *(top)* A single feature point's trajectory. *(middle* and *bottom)* Key frames associated with the valleys of a feature point's trajectory

Feature Point Tracking. After selecting a set of feature points and setting a tracking window W_T (*3x3* in the current implementation) an iterative feature tracking algorithm has been used [11]. Assuming a small interframe motion, feature points are tracked by minimizing the sum of squared differences between two consecutive frames.

Points, lying on the head or shoulders, are the best candidates to satisfy the aforementioned assumption. Most of the other points (e.g. points over the legs, arms or hands, are missed after a couple of frames). Fig. 3(*top*) illustrates feature points detected in the first frame of the video sequence used in [13]. Fig. 3*(bottom-left)* depicts the trajectories of the feature points when all frames are considered. On the contrary, Fig. 3*(bottom-right)* shows the trajectories after removing static points. In the current implementation we only use one feature point's trajectory. Further improvements could be to merge feature points' trajectories in order to generate a more robust approach.

2.3 Motion Model Tuning

The outcome of the previous stage is the trajectory of a feature point (Fig. 4(*top*)) consisting of peaks and valleys. Firstly, the first-order derivative of the curve is computed to find peaks' and valleys' positions by seeking the positive-to-negative zero-crossing points. Peaks correspond to those frames where the pedestrian reaches the maximum height, which happens in that moment of the half walking cycle when the hip angles are minimum. On the contrary, the valleys correspond to those frames where the two legs are in contact with the floor and then, the hip angles are maximum. So, the valleys are used to find key frames, while the peaks are used for footprint detection. The frames corresponding to each valley of Fig. 4*(top)* are presented in Fig. 4*(middle)* and *(bottom)*. An interesting point of the proposed approach is that in this video sequence, in spite of the fact that the pedestrian is carrying a folder, key frames are correctly detected; as a result, legs and torso correspondence are easily computed. On the contrary, an approach based on shape, such as [3], will face difficulties, since it will try to minimize the matching error based on the whole shape (including folder).

After detecting key frames corresponding to the valleys of a trajectory, it is necessary to define also the footprints of the pedestrian throughout the sequence. In order to achieve this, body silhouettes were computed by an image segmentation algorithm [12]. Additionally, other segmented video sequences were provided by [14]. Footprint positions are computed as follow.

Throughout a walking displacement sequence, there is always, at least, one foot in contact with the ground, with null velocity (pivot foot). In addition, there is one instant per walking cycle in which both feet are in contact with the floor (both with null velocity). The foot that is in contact with the floor can be easily detected by extracting its defining *static points*. A point is considered as a static point $spt^{F}_{(i,j)}$ in frame *F,* if it remains as a boundary point $bp^{F}_{(i,j)}$ (silhouette point, Fig. 5(*left*)) in at least three consecutive frames—value computed experimentally $stp^{F}_{(i,j)} \Rightarrow (bp^{F-1}_{(i,j)}, bp^{F}_{(i,j)}, bp^{F+1}_{(i,j)})$.

The result of the previous stage is a set of static points distributed along the pedestrian's path. Now, the problem is to cluster those points belonging to the same foot. Static points defining a single footprint are easily clustered by studying the peaks' positions in the feature point's trajectory. All those static points in a neighborhood of $F\pm3$ from the frame corresponding to a peak position *(F)* will be clustered together and will define the same footprint (fp_i). Fig. 5*(right)* shows the footprints detected after processing the video sequence of Fig. 4.

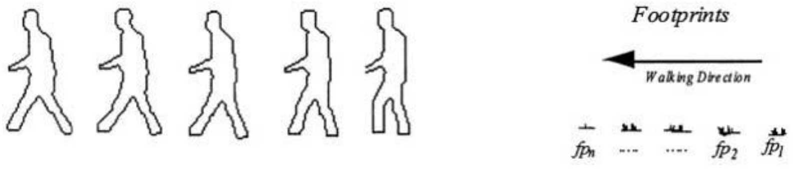


Fig. 5. (*left*) Five consecutive frames used to detect static points. *(right)* Footprints computed after clustering static points generated by the same foot (picks in a feature point's trajectory (Fig. 4(*top*))

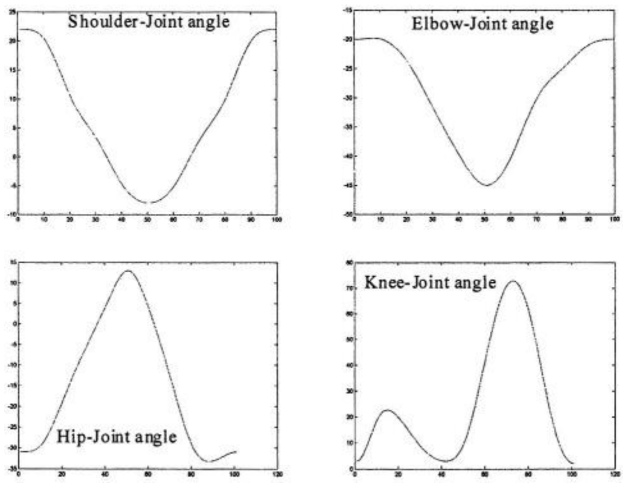


Fig. 6. Motion curves of the joints at the shoulder, elbow, hip and knee (computed from [9])

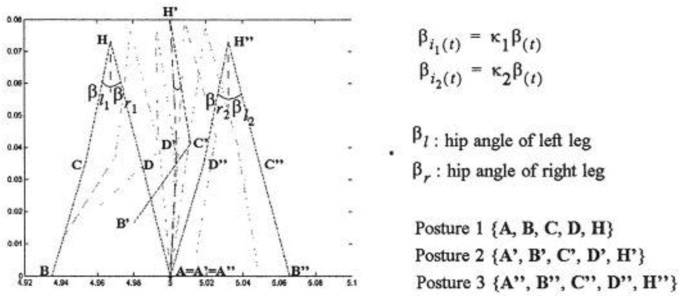

H'
H
H''
C
D
D'
C'
D''
C''
B'
B
A=A'=A''
B''
$\beta_{i_1(t)} = \kappa_1\beta_{(t)}$
$\beta_{i_2(t)} = \kappa_2\beta_{(t)}$
β_l : hip angle of left leg
β_r : hip angle of right leg
Posture 1 {A, B, C, D, H}
Posture 2 {A', B', C', D', H'}
Posture 3 {A'', B'', C'', D'', H''}

Fig. 7. Half walking cycle executed by using scale factors (κ_1, κ_2) over the hip motion curve presented in Fig. 6 (knee motion curve is not tuned at this stage). Spatial positions of points **(D, H, C** and **B)** are computed by using angles from the motion curves and trigonometric relationships

As it was introduced above, key frames are defined as those frames where both feet are in contact with the floor. At every key frame, the articulated human body structure reaches a posture with maximum hip angles. In the current implementation, hip angles are defined by the legs and the vertical axis containing the hip joints. This maximum value, together with the maximum value of the hip motion model (Fig. 6) are used to compute a scale factor κ. This factor is utilized to adjust the hip motion model to the current pedestrian's walking. Actually, it is used for half the walking cycle, which does not start from the current key frame but from a quarter of the walking cycle before the current key frame until halfway to the next one. The maximum hip angle in the next key frame is used to update this scale factor.

This local tuning, within a half walking cycle, is illustrated with the 2D articulated structure shown in Fig. 7, from Posture 1 to Posture 3. A 2D articulated structure was chosen in order to make the understanding easier, however the tuning process is carried out in a 3D space, estimated by using an average pedestrian's height. The two footprints of the first key frame are represented by the points **A** and **B,** while the footprints of the next key frame are the corresponding points **A"** and **B"**. During this half walking cycle one foot is always in contact with the floor (so points **A = A'= A"),** while the other leg is moving from point **B** to point **B".** In halfway to **B",** the moving leg crosses the other one (null hip angle values). Points **C, C', C"** and **D, D', D"** represent the left and right knee, while the points **H, H', H"** represent the hip joints.

Given the first key frame, the scale factor κ_1 is computed and used to perform the motion ($\beta_{i_1(t)}$) through the first quarter of the walking cycle. The second key frame **(A", B")** is used to compute the scale factor κ_2. At each iteration of this half walking cycle, the spatial positions of the points **B, C, D** and **H** are calculated using the position of point **A,** which remains static, the hip angles of Fig. 6 scaled by the corresponding factor κ_i is and the knee angles of Fig. 6. The number of frames in between the two key frames defines the sampling rate of the motion curves presented on Fig. 6. This allows handling variations in the walking speed.

As aforementioned, the computed factors κ_i are used to scale the hip angles. The difference in walking between people implies that all the motion curves should be modified by using an appropriate scale factor for each one. In order to estimate these factors an error measurement (registration quality index: *RQI*) is introduced. The proposed *RQI* measures the quality of the matching between the projected 3D model and the corresponding human silhouette. It is defined as: *RQI = overlappedArea/totalArea,* where total area consists of the surface of the projected 3D model plus the surface of the walking human Fig. less the overlapped area, while the overlapped area is defined by the overlap of these two surfaces. Firstly, the algorithm computes the knee scale factor that maximizes the *RQI* values. In every iteration, an average *RQI* is computed for all the sequence. In order to speed up the process the number of frames was subsampled. Afterwards, the elbow and shoulder scale factors are estimated similarly. They are computed simultaneously using an efficient search method.

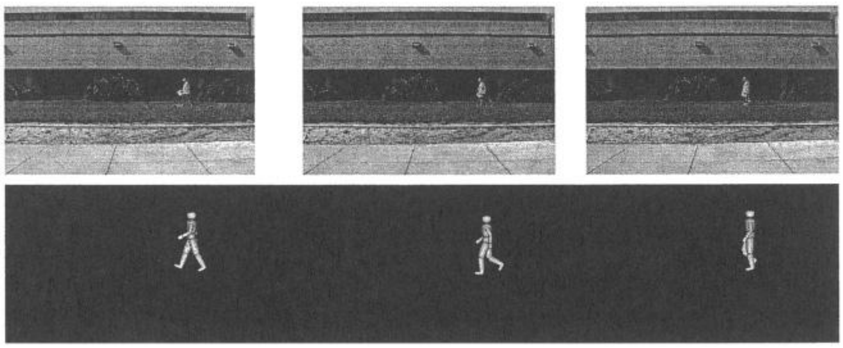


Fig. 8. *(top)* Three different frames of the video sequence used in [13]. *(bottom)* The corresponding 3D walking models

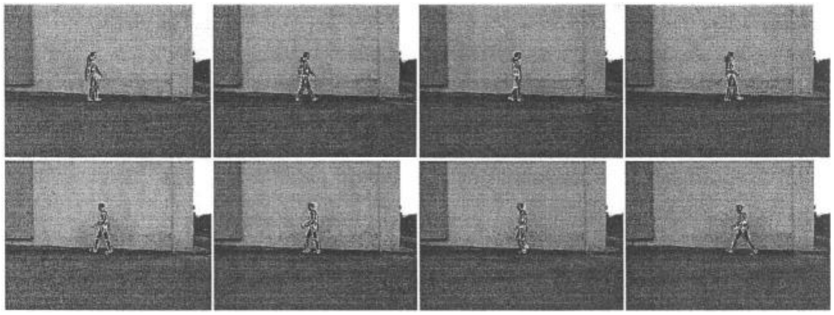


Fig. 9. Input frames of two video sequences (*240x320* each)

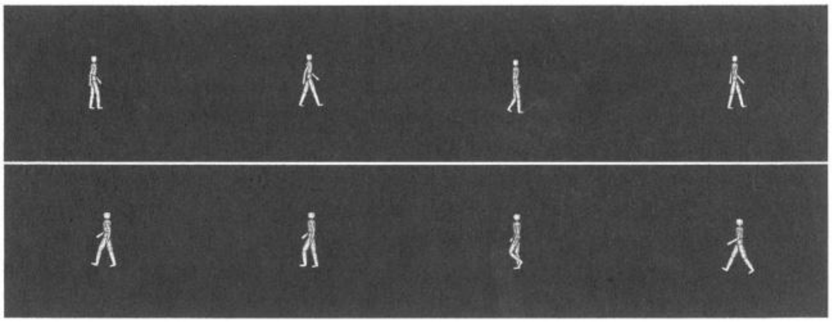


Fig. 10. 3D models corresponding to the frames presented in Fig. 9 *(top)* and *(bottom)* respectively

3 Experimental Results

The proposed technique has been tested with video sequences used in [13] and [14], together with our own video sequences. In spite that the current approach has been developed to handle sequences with a pedestrian walking over a planar surface, in a plane orthogonal to the camera direction, the technique has been also tested with an oblique walking direction (see Fig. 11) showing encouraging results. The video sequence used as an illustration throughout this work consists of 85 frames of *480x720* pixels each, which have been segmented using the technique presented in [15]. Some of the computed 3D walking models are presented in Fig. 8*(bottom)*, while the original frames together with the projected boundaries are presented in Fig. 8*(top)*.

Fig. 9*(top)* presents frames of a video sequence defined by 103 frames (*240x320* pixels each), while Fig. 9*(bottom)* correspond to a video sequence defined by 70 frames (*240x320* pixels each). Although the speed and walking style is considerably different, the proposed technique can handle both situations. The corresponding 3D models are presented in Fig. 10 *(top)* and *(bottom)* respectively.

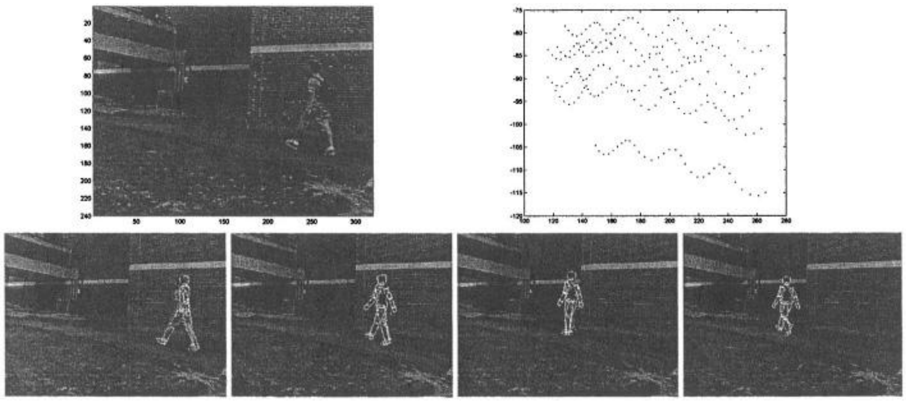


Fig. 11. *(top-left)* Feature points of the first frame. *(top-right)* Feature points' trajectories. *(bottom)* Some frames illustrating the final result (segmented input has been provided by [14])

Finally, the proposed algorithm was also tested on a video sequence, consisting of 70 frames of *240x320* pixels each, containing a diagonal walking displacement (Fig. 11). The segmented input frames have been provided by the authors of [14]. Although the trajectory was not on a plane orthogonal to the camera direction, feature point information was enough to capture the pedestrian attitude.

4 Conclusions and Future Work

A new approach towards human motion modeling and recovery has been presented. It exploits prior knowledge regarding a person's movement as well as human body kinematics constraints. At this paper only walking has been modeled. Although constraints about walking direction and planar surfaces have been imposed, we expect to find results similar to the ones presented in [17] (frontal and oblique walking direction).

The extension of the proposed technique to model other kinds of human body cyclic movements (such as running or going up/down stairs) will be studied by using the same technique (point features detection merged with the corresponding motion model). In addition, the use of a similar approach to model the displacement of other articulated bodies (animals in general [16]) will be studied. Animal motion modeling (i.e. cyclic movement) can be understood as an open articulated structure, however, when more than one extremity is in contact with the floor, that structure becomes a closed kinematics chain with a reduced set of DOFs. Therefore, a motion model could be computed by exploiting these particular features.

Further work will also include the tuning of not only motion model's parameters but also geometric model's parameters in order to find a better fitting. In this way, external objects attached to the body (like a handbag or backpack) could be added to the body and considered as a part of it.

References

[1] Sappa, A, Aifanti, N., Grammalidis, N and Malassiotis, S: Advances in Vision-Based Human Body Modeling, Chapter book in: 3D Modeling and Animation: Synthesis and Analysis Techniques for the Human Body, N. Sarris and M.G. Strintzis (Eds.), Idea-Group Inc., 2004 (in press).

[2] Delamarre, Q and Faugeras, O.: 3D Articulated Models and Multi-View Tracking with Physical Forces, Special Issue on Modelling People, Computer Vision and Image Understanding, Vol. 81, 328-357, 2001.

[3] Ning, H. Tan, T., Wang, L. and Hu, W.: Kinematics-Based Tracking of Human Walking in Monocular Video Sequences, Image and Vision Computing, Vol. 22, 2004, 429-441.

[4] Wang, L., Tan, T., Hu, W. and Ning, H.: Automatic Gait Recognition Based on Statistical Shape Analysis, IEEE Trans. on Image Processing, Vol. 12 (9), September 2003, 1-13.

[5] Gavrila, D. and Davis, L.: 3-D Model-Based Tracking of Humans in Action: a Multi-View Approach, IEEE Int. Conf. on Computer Vision and Pattern Recognition, San Francisco, USA, 1996.

[6] Barron, C. and Kakadiaris, I.: Estimating Anthropometry and Pose from a Single Camera, IEEE Int. Conf. on Computer Vision and Pattern Recognition, Hilton Head Island, USA, 2000.

[7] Metaxas, D. and Terzopoulos, D.: Shape and Nonrigid Motion Estimation through Physics-Based Synthesis, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 10 (6), June 1993, 580-591.

[8] Song, Y., Goncalves, L. and Perona, P.: Unsupervised Learning of Human Motion, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 25 (7), July 2003, 1-14.

[9] Rohr, K.: Human Movement Analysis Based on Explicit Motion Models, Chapter 8 in Motion-Based Recognition, M. Shah and R. Jain (Eds.), Kluwer Academic Publisher, Dordrecht Boston 1997, pp. 171-198.

[10] Sappa, A., Aifanti, N., Malassiotis, S. and Strintzis, M.: Monocular 3D Human Body Reconstruction Towards Depth Augmentation of TelefVision Sequences, IEEE Int. Conf. on Image Processing, Barcelona, Spain, September 2003.

[11] Ma, Y., Soatto, S., Kosecká, J. and Sastry, S.: An Invitation to 3-D Vision: From Images to Geometric Models, Springer-Verlag New York, 2004.

[12] Kim, C. and Hwang, J.: Fast and Automatic Video Object Segmentation and Tracking for Content-Based Applications, IEEE Trans. on Circuits and Systems for Video Technology, Vol. 12 (2), February 2002, 122-129.

[13] Phillips, P., Sarkar, S., Robledo, I., Grother, P. and Bowyer, K.: Baseline Results for the Challenge Problem of Human ID Using Gait Analysis, IEEE Int. Conf. on Automatic Face and Gesture Recognition, Washington, USA, May 2002.

[14] Jabri, S., Duric, Z., Wechsler, H. and Rosenfeld, A.: Detection and Location of People in Video Images Using Adaptive Fusion of Color and Edge Information, 15th. Int. Conf. on Pattern Recognition, Barcelona, Spain, September 2000.

[15] Kim, C. and Hwang, J.: Fast and Automatic Video Object Segmentation and Tracking for Content-Based Applications, IEEE Trans. on Circuits and Systems for Video Technology, Vol. 12 (2), February 2002, 122-129.

[16] Schneider, P. and Wilhelms, J.: Hybrid Anatomically Based Modeling of Animals, IEEE Computer Animation'98, Philadelphia, USA, June 1998.

[17] Wang, L, Tan, T., Ning, H. and Hu, W.: Silhouette Analysis-Based Gait Recognition for Human Identification, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 25 (12), December 2003, 781-796.

Variant Design in Immersive Virtual Reality: A Markup Language for Scalable CSG Parts

Peter Biermann[1] and Bernhard Jung[2]

[1] Laboratory for Artificial Intelligence and Virtual Reality, Faculty of Technology
University of Bielefeld
[2] Media Informatics, ISNM International School of New Media
University of Lübeck

Abstract. In many product areas, a growing trend can be observed towards variant design, i.e. the development of customized designs based on variations of mature product models. We have developed a Virtual-Reality (VR) system for variant design that supports the real-time scaling and subsequent simulated assembly of hierarchical, CSG-like parts. An XML-based format, VPML, serves as description for the scalable CSG parts. VPML part descriptions determine how the scaling behavior of the whole part affects the scaling of its subparts, constrain the translation and rotation of subparts w.r.t. their parent parts, and define the scalable parts' dynamic mating properties. The part descriptions are utilized by several submodules of the overall VR system including: a) algorithms for real-time CSG visualization, b), the updating of part geometry using the ACIS CAD kernel, and c), the assembly simulation engine. The VR system runs in a CAVE-like large screen installation and enables interactive variant design using gesture and speech interactions.

1 Introduction

Visions of future CAD/CAM systems include the application of Virtual Reality (VR) techniques for real-time interactive design, variation, assembly, and evaluation of three dimensional product models ("virtual prototypes") [1], [11], [5]. The central idea is to combine unique features of VR such as intuitive human-computer interaction based on gesture and speech with realistic graphical simulations to result in new, real-time capable tools supporting the early phases of the product development process. The increased utilization of virtual prototypes instead of physical ones holds the promise of shortened development cycles, increased product quality, and, not least, reduced cost: "80% of development costs and 70% of life cycle costs of a product are determined during its conceptual phase" [9].

Another trend in CAD/CAM concerns the movement towards the development of customized designs based on variations of mature product models, usually referred to as *variant design* [4] [10]. A prominent example, numerous others being listed e.g. in [4], is the automotive industry where a wide variety of product models is derived from a relatively small number of product platforms (e.g. shared platforms of Volkswagen/Skoda/Seat, Chrysler/Mitsubishi, Ford/Mazda). One of our goals in the project "Virtuelle Werkstatt" is to extend previous research results in VR-based virtual prototyping – mainly centered on assembly simulation involving non-deformable parts from standardized construction kits using gesture and speech [2] [6] – to develop a VR platform for interactive variant design.

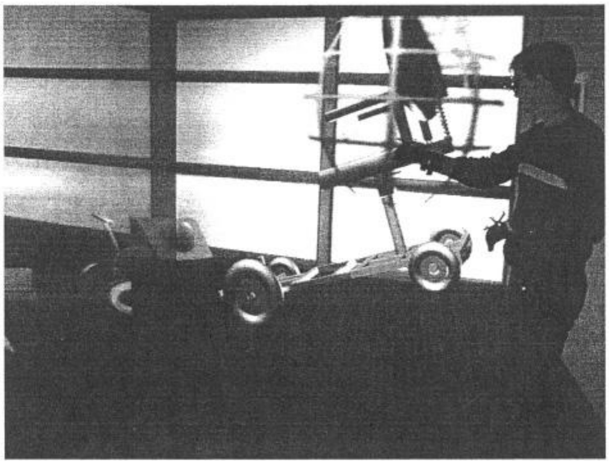


Fig. 1. Variant design of a "Citymobile" vehicle in a CAVE-like VR installation

Of particular interest are changes in the size of the variant parts by means of special scaling operations that are capable of preserving the shape of certain sub-elements of a part. E.g., the scaling of a plate with several holes should preserve the roundness of the holes, no matter in which dimension the plate is scaled (see Figure 2). Fundamental to our solution to this scaling problem is the definition of a hierarchical, CSG-like part model. For example the plate from figure 2 would be modeled as a box-like object from which three cylindrical shapes are subtracted. The part model also defines and constrains the scaling behavior of the part and its sub-shapes.

The rest of this paper is organized as follows: The next section introduces VPML, an XML-based language for hierarchically structured and scalable variant parts. Section 3 shows how various sub-modules of a prototypical VR system for variant design utilize the information contained in VPML models. Section 4 presents an extended example of variant design in the context of "Citymobile" vehicles (motorized one-seated carts typically used by the elderly or walking impaired). Finally we conclude and address future work.

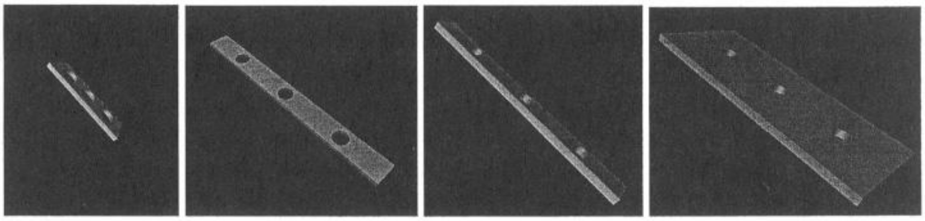
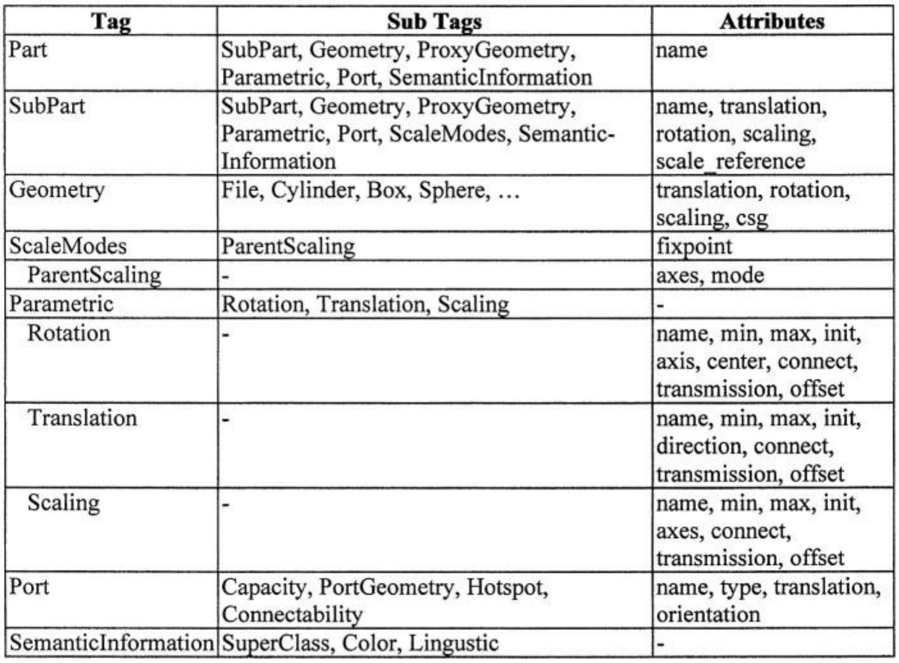


Fig. 2. A plate is scaled in different dimensions. The size of the holes remains unchanged

2 VPML – A Markup Language for Describing Variant Parts

VPML (“Variant Part Markup Language”) is an XML-based markup language for describing hierarchical, CSG-like parts, their scaling behavior and dynamic mating properties. Fig. 3 gives an overview of the mostly used tags in VPML and their attributes. The tags are explained below.



Tag	Sub Tags	Attributes
Part	SubPart, Geometry, ProxyGeometry, Parametric, Port, SemanticInformation	name
SubPart	SubPart, Geometry, ProxyGeometry, Parametric, Port, ScaleModes, Semantic-Information	name, translation, rotation, scaling, scale_reference
Geometry	File, Cylinder, Box, Sphere, ...	translation, rotation, scaling, csg
ScaleModes	ParentScaling	fixpoint
ParentScaling	-	axes, mode
Parametric	Rotation, Translation, Scaling	-
Rotation	-	name, min, max, init, axis, center, connect, transmission, offset
Translation	-	name, min, max, init, direction, connect, transmission, offset
Scaling	-	name, min, max, init, axes, connect, transmission, offset
Port	Capacity, PortGeometry, Hotspot, Connectability	name, type, translation, orientation
SemanticInformation	SuperClass, Color, Lingustic	-

Fig. 3. A choice of the tags used in VPML

2.1 Describing Part Hierarchy

XML documents are a means to describe tree structures. In VPML - an XML dialect - the document’s tree structure corresponds to the CSG-tree defining the variant parts: The `<Part>` element denotes the whole part (the root of the XML-tree), recursively nested `<Subpart>` elements are used to describe the part hierarchy. Each part or subpart can have an associated `<Geometry>` which may either be primitive, e.g. box, sphere, etc, or link to an external Inventor file. The `csg` attribute specifies whether a geometry is positive or negative, e.g. holes can be represented as cylinders with a negative `csg` attribute. Subparts include further attributes specifying their position, orientation and initial scaling w.r.t. their parent part.

2.2 Describing Scaling Behavior

If a part were scaled in one direction only by simply using a scaling matrix in a scene graph the subparts would be deformed the same way the part itself is scaled.

However, if e.g. holes in the scaled part should remain their size and roundness then a different scaling behavior has to be defined.

Therefore each subpart can define their scaling behavior in relation to its parents. For each of the three possible scaling axes, a specific `<ParentScaling>` behavior can be defined:

• `None:`	Keep the scaling for the subpart for that axis no matter of the scaling of the parent.
• `Parent:`	Use the same scaling as the parent – This is the normal behavior for objects which are hierarchically ordered in a scene graph. It is therefore defined as default.
• `Reference:`	Use the same scaling as one of the parents which was specially marked as a reference part.

If the scaling behavior is set to "`None`" the local transformation of the subpart will be set to the inverse scaling of the accumulated scaling of the parents. A fix-point can be set where the subpart is fixated when the parent is scaled.

The last scaling behavior "`Reference`" allows the definition of combined scaling of two or more parts, which are at different positions in the scene graph. If the direct parents of these objects should not be scaled, the "local" scale mode is not useable. In this case the scaling parameter is attached to a common parent node of the graph and it is marked as the reference node for these sub-nodes. Their scaling will be adapted to the scaling of the reference part, independent of their direct parents.

Parametric changeable scaling factors can be defined for the whole variant part and for each subpart. These scaling factors might combine two or three axes which can only be altered together.

2.3 Describing Parametric Translation, Rotation and Scaling of Subparts

Subparts of the variant part, which should be modifiable in the Virtual Environment, can be tagged with a special `<Parameter>`-Tag. With this tag it is possible to describe parametric scaling as described in the paragraph above, but it is also possible to define rotational and translational changes.

Each transformation described by a Parameter-Tag, results in a single floating point value for adjusting the part. In the case of parametric scaling, it will result in a parameter for scaling a single axis - respectively two or three axes combined to remain the proportion of these axes while scaling. The parametric rotation is defined by the rotation axis and center, while the resulting value will affect the angle of the rotation. The translation is defined by a normalized vector, which defines the direction in which the part is translated by the value of the parameter.

In the application the value of each Parameter-Tag can be queried and set in real-time and two parameters can be coupled with a fixed factor to realize constrained movements of subparts (see the Example in Fig. 11).

2.4 Describing Mating Properties

To enable interactive assembly simulations in VR, VPML models are further equipped with a means to describe the variant parts' mating properties. This is achieved by linking the VPML models to a knowledge-based framework for task-level assembly modeling [6] [7]. This framework utilizes a knowledge-based model of connection-sensitive part surfaces or *ports,* e.g. the shaft of a screw or the inside thread of a nut, and constraints imposed by different kinds of connections, e.g. screwed, inserted, or welded. Taxonomies with predefined port and connections concepts capture the most common mating properties of CAD-based parts. In VPML, a special `<Port>` tag can be used to attach port information to a part or subpart. The required port information concerns e.g. the type, position, orientation, size (capacity). More detailed information like port geometry may also be provided. The port information is used for assembly and disassembly of parts, e.g. by snap-to-fit enhanced direct manipulation using data-gloves or by spoken natural language instructions. Adjustment operators enable the relative repositioning of connected parts along the degrees of freedom induced by the specific type of connection; e.g. a screw could be only halfway inserted into a nut at same stage of the simulation and become fully fixated later. For further details of the modeling of ports see [7].

2.5 Other Information

For each subpart, additional information for the application can be provided. E.g. color and material information can be specified to modify the shading of the subparts. A further tag, `<SemanticInformation>` concerns type information, which can be used to link the part model to an externally defined taxonomy of part concepts as well as linguistic information. The conceptual and linguistic information is e.g. exploited in the processing of natural language commands to the virtual environment.

3 Utilizing VPML Models in Immersive VR

We have developed an operational immersive VR system for variant design that enables an interactive, non-uniform scaling and assembly of CSG-based variant parts using gesture and speech interactions. This section describes how VPML models of variant parts are used by the various sub-modules of the CAVE-based VR system. In particular, VPML supports two complementary methods concerned with the non-uniform scaling of VPML parts: A real-time, image-based CSG which is used during interactive part modifications; and a CAD-based CSG that is used for the generation of exact polygonal part models after the interaction is completed.

3.1 Image-Based CSG for Real-Time Visualization

VR systems for interactive variant design should provide real-time visual feedback during all manipulations of the variant parts, including scaling operations. Since a step-keeping computation of polygonal object models during scaling operations is not possible, a real-time, image-based CSG visualization was developed, which builds upon the Goldfeather-Algorithm [3]. This algorithm produces the visual effect of CSG operations, without generating the polygonal model itself, by using the OpenGL [12] stencil buffer to mask out the parts of the CSG primitives, which are not visible.

The Goldfeather algorithm was adapted to be compatible with the scene graph structure and the limited instruction set of the distributed OpenGL available in the PC-cluster serving our CAVE-like Virtual Environment. In contrast to the original algorithm which needs to save and restore the depth-buffer often and thus is not suitable for the distributed rendering architecture, our adaptation is mainly based on frame-buffer operations but also uses a compressed Stencil-Buffer. With the graphic hardware available today, this algorithm, which uses multi-pass rendering techniques, is fast enough to meet the requirements for real-time interaction with the modifiable parts.

3.2 Generating Polygonal Models

While the image-based CSG is well-suited for real-time visualization of the parts' scaling behavior - synchronized e.g. with user gestures in the VR environment - it does not produce any polygonal models. Polygonal models are needed, however, for different simulations of part behavior such as assembly and collision handling. Furthermore, the computational complexity of the multi-pass rendering involved in image-based CSG can be reduced when polygonal models of the scaled parts are available. Therefore, upon the completion of a user interaction in the VR environment, a polygonal model of the scaled part is generated.

The ACIS CAD kernel is used for the computation of the polygonal models. In the polygonal CSG subsystem, VPML models are represented as trees of ACIS bodies where each body, among other information, has a transformation matrix associated with it. When the new scaling factors of a part or subpart are transmitted to the polygonal CSG system, the transformation matrix of that part or subpart is updated according to the new scaling factors and the scale modes defined in the VPML model. This process is recursively repeated for the part's children parts. Then, set operations (union and subtract) are applied in a bottom-up fashion, starting at the leaves up to the root of the part tree, as governed by the subparts' CSG attributes (i.e. positive or negative geometry). Finally, the whole part is triangulated and an Inventor model is generated for inclusion in the VR scene graph. The appearance of the Inventor models also accounts for the colors defined in the VPML part definitions. Depending on part complexity, the construction of the polygonal model takes on the order of several tenths of a second to some ten seconds time. The updated polygonal part model is loaded into the VR scene graph in an asynchronous fashion such that a possibly on-going user interaction is not disrupted.

3.3 Assembly Simulation with Variant Parts

Our VR system for variant design contains an assembly simulation component which accomplishes part matings based on port descriptions as described in Section 2.4. The assembly simulation component extends previous work [6] to operate also with scalable variant parts.

In general, scaling operations may affect the size (capacity) but also the position of ports w.r.t. the complete part; e.g. the scaling operations in Figure 2 affect the position of the two outer holes (although in this case, the size of the holes is not changed). To support further parametric rotation and translation operations, capacity, position, and orientation of ports need to be updated after such user interactions in VR.

The assembly simulation maintains a hierarchically structured part representation which mirrors the part hierarchy in the VPML file. Ports can be associated with parts or subparts at any level of this hierarchy. When a scaling operation in VR is completed, the part-subpart transformations are updated according to the new scale factors and part scale modes defined in the VPML model in the same way as the polygonal CSG component (Sec. 3.2). The core algorithms for the assembly simulation expect the port positions, orientations, and capacities directly w.r.t. the root part. These data can be directly computed from the hierarchical model as the combined transformation from the root part to the port. The updates of the port representation ensure, e.g. that the wheel of the Citymobile is attached to a scaled axle at the correct position or that a prolonged screw can be used for attachment of thicker parts.

3.4 VPML for Interpretation of Natural Language Instructions

To facilitate the interpretation of natural language instructions to the virtual environment, the semantic information contained in VPML models is translated into the semantic network type knowledge representations described in [13] [8]. E.g., in sentences like *"attach the green wheel to the body" or "make that round thing bigger"* the semantic information associated with the parts enables the matching of attributions like *"green"* or *"round"* with the properties of the virtual objects or to find all objects of type WHEEL in the scene. Additional gestures, e.g. pointing, also contribute to the resolution of references during the interpretation of user interactions.

4 Extended Example: The Citymobile Seat

The Citymobile Seat can be adjusted by seven parameters: Five concern the scaling of subparts and two the rotational adjustment of subparts. Some of the parameters - like the length and the width of the seat - relate to variations while designing the seat before the manufacturing, others simulate changeable parts, which can be modified when the seat is in use. A port is placed on the lower end of the seat. At this port, the seat can be connected to the chassis of the Citymobile, as shown in Figure 1.

Fig. 5 - Fig. 9 show the effect when the five scaling parameters of the seat are modified. Fig. 10 shows a rotational parameter which simulates the possibility to move away the arm rests for easier access to the seat. Each modification in Figures 5–11 is based on one scaling or rotational parameter. The resulting scaling-operations of the subparts are marked with double pointed arrows. The movements which result from the hierarchical structure of the building part and the corresponding scale-modes in the VPML-file of the seat are shown by single arrows. E.g. the scaling of the whole seat in the X-direction, see Fig. 8, results in scaling the lower seat and the back-rest, while the arm-rests are only moving outward.

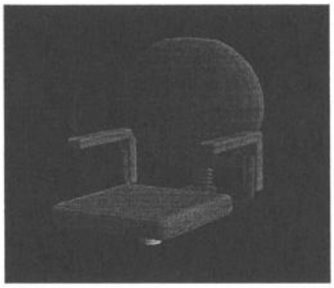


Fig. 4. The default Citymobile-Seat

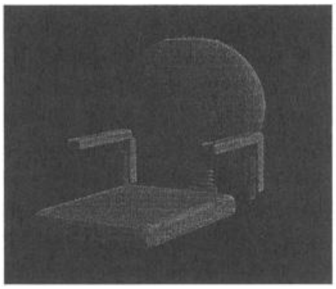


Fig. 5. The seat lengthened

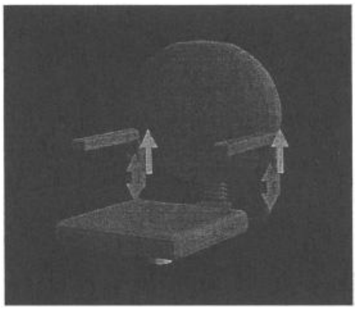


Fig. 6. Arm rests adjusted in height

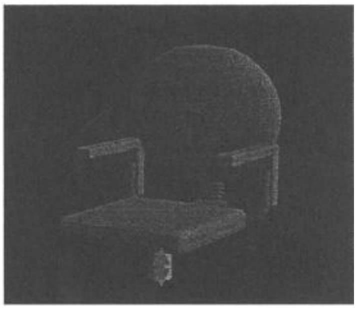


Fig. 7. The whole seat adjusted in height

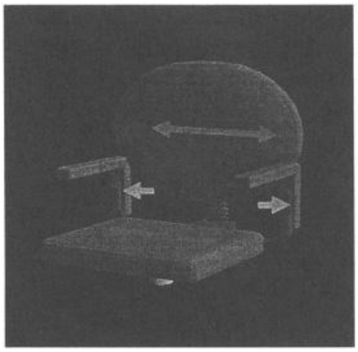


Fig. 8. The broadened seat

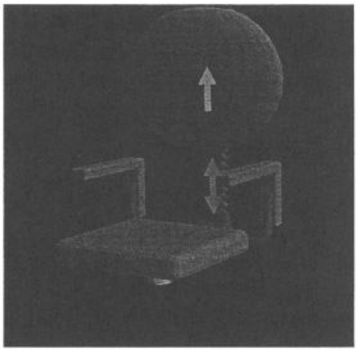


Fig. 9. Height adjustment of the backrest

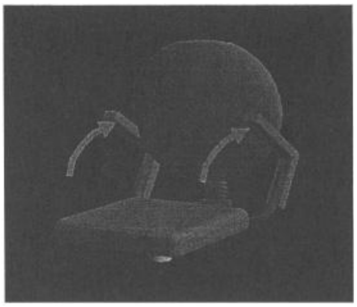


Fig. 10. The seat with arm rests rotated 45 Degrees for easier access

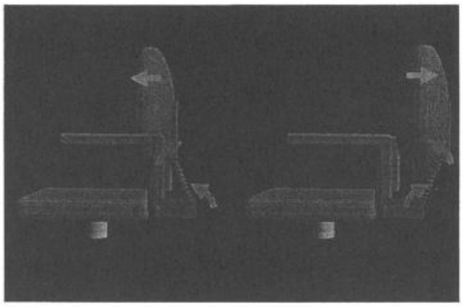


Fig. 11. Combined rotations for adjusting the backrest

In Fig. 11 the effect of two coupled rotational parameters are displayed. While the lower part of the backrest is rotated clockwise the backrest itself is rotated counter clockwise to keep it in an upright position.

The following part of a VPML-Description (see Fig. 12) shows the definition of two subparts which form the backrest of the seat with the tags for the combined rotational parameters.

```
<Part name="citymobile_seat">
      ...
                <SubPart name="backrest_holder">
                  ...
                  <Parametric>
                    <Scaling name="adjust_height"
                      min="0.8" max="2.0" axis="Y"/>
                    <Rotation name="backrest_hldr_rotate"
                      min="-20" max="20"
                      center="0 0 -0.3"/>
                  </Parametric>
                <SubPart name="backrest">
                  ...
                  <ScaleModes fixpoint="0 0.4 -0.3">
                    <ParentScaling axis="Y" mode="none"/>
                      ...
                  <ScaleModes/>
                  <Parametric>
                    <Rotation name="backrest_rotate"
                      center="0 0.4 -0.3"
                      combine="backrest_hldr_rotate"
                      transmission="-1"/>
                    </Parametric>
                  </SubPart>
                </SubPart>
      ...
</Part>
```

Fig. 12. Fragment of the VPML-Description for the backrest part of the seat

The `combine`-Attribute of the second <`Rotation`> Tag establishes the coupling between the parameters and the `transmission`-Attribute is set to `-1` in order to achieve the inverse rotation of the backrest in relation to the holder.

The fragment of the VPML-Description also shows the description of the scaling of the backrest holder (as shown in Fig. 9). This scaling parameter scales the backrest holder in Y-direction, while the backrest itself doesn't scale with the holder, since the scale mode for this axis is set to "none". The fixpoint between the two subparts is set to the upper end of the backrest holder, which causes the movement of the backrest to remain connected to this point, during a scaling operation.

5 Conclusions

We have presented VPML, a markup language for describing interactively modifiable variant parts. VPML allows the description of hierarchically structured, CSG-like part models, their scaling behavior, and their mating properties. We use VPML models in an operational CAVE-based VR system for variant design that enables the real-time scaling and assembly of variant parts based on natural language and gestural user interactions.

VPML models contain information that is exploited for several purposes in the VR system including image-based CSG for real-time visualization, dynamic generation of polygonal models, assembly simulation, and natural language processing. The different subsystems require different (yet overlapping) subsets of the information contained in VPML models. Standard XML tools like XSL transformations and SAX parsing facilitate the translation of VPML models into specialized representations required by the various subsystems.

Current work focuses on simulating gears and kinematic chains using combined parametric rotations and translations. Together with the possibility to constraint the movements of connected parts, according to their type of connection, it is possible to build a mechanism consisting of multiple variable parts and gears. In this mechanism the movements of the components are propagated to the connected parts by the movement of the gears, which are defined in VPML.

In combination with collision-detection methods it is further possible to test the movements of the mechanism for sufficient clearance for each component of interactively designed assemblies. For example it will be possible after the assembly of the steering system of the Citymobile, to test the steering by turning the steering wheel in the virtual scene and analyze e.g. how this affects the front wheels of the vehicle.

Acknowledgement

This work is partially supported by the Deutsche Forschungsgemeinschaft (DFG).

References

[1] J. A. Adam. Virtual reality is for real. *IEEE Spectrum,* 30(10):22-29, 1993.

[2] P. Biermann, B. Jung, M. Latoschik & I. Wachsmuth: Virtuelle Werkstatt: A Platform for Multimodal Assembly in VR. In *Proceedings Fourth Virtual Reality International Conference (VRIC 2002),* Laval, France, 19-21 June 2002, 53-62.

[3] J. Goldfeather, J. Hultquist, H. Fuchs. Fast Constructive Solid Geometry in the Pixel-Powers Graphics System, *Computer Graphics (SIGGRAPH '86 Proceedings),* ACM, Vol. 20, No.4, pp. 107-116, 1986.

[4] J. P. Gonzalez-Zugasti, K. N. Otto & J. D. Baker. Assessing value in platformed product family design. *Research in Engineering Design,* Vol. 13. No. l,pp 30-41. 2001.

[5] R. Gupta, D. Whitney, and D. Zeltzer. Prototyping and design for assembly analysis using multimodal virtual environments. *Computer-Aided Design,* 29(8):585-597, 1997.

[6] B. Jung, M. Latoschik, I. Wachsmuth: Knowledge-Based Assembly Simulation for Virtual Prototype Modeling. *IECON'98 - Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society, Vol. 4,* IEEE, 1998,2152-2157.

[7] B. Jung: Task-Level Assembly Modeling in Virtual Environments. *Computational Science and Its Applications - ICCSA 2003, International Conference, Montreal, Canada, May 18-21, 2003, Proceedings, Part III.* LNCS 2669. Springer. 2003, 721-730.

[8] M. E. Latoschik, M. Schilling. Incorporating VR Databases into AI Knowledge Representations: A Framework for Intelligent Graphics Applications. *Proceedings of the Sixth IASTED International Conference on Computer Graphics and Imaging.* Honolulu, Hawaii, August 2003.

[9] J. S. Lombardo, E. Mihalak, and S. R. Osborne. Collaborative virtual prototyping. *Johns Hopkins APL Technical Digest,* Vol 17(3): 295-304. 1996.

[10] D. Pavlic, N. Pavkovic & M. Storga (2002). Variant design based on product platform. *Proceedings International Design Conference – Design 2002,* pp. 397-402.

[11] M. J. Pratt. Virtual prototypes and product models in mechanical engineering. In J. Rix, S. Haas, and J. Teixeira, editors, *Virtual Prototyping - Virtual environments and the product design process,* pages 113-128. Chapman and Hall, 1995.

[12] Kurt Akeley. *OpenGL reference manual: the official reference document for OpenGL.* Addison Wesley, 1995

[13] I. Wachsmuth, B. Jung: Dynamic Conceptualization in a Mechanical-Object Assembly Environment. *Artificial Intelligence Review 10(3-4):345-368,* 1996.

Application of Repeated GA to Deformable Template Matching in Cattle Images

Horacio M. González Velasco, Carlos J. García Orellana,
Miguel Macías Macías, Ramón Gallardo Caballero, and
M. Isabel Acevedo Sotoca

Departamento de Electrónica e Ingeniería Electromecánica
Universidad de Extremadura
Av. de la Universidad, s/n. 10071 Cáceres, Spain
horacio@nernet.unex.es

Abstract. In this work, a system that enables accurate location of contours within outdoors–taken digital images is presented. This system is based on the optimization of an objective function involving a selective edge detector and a distance map extractor, applied sequentially to the original image. In order to guarantee a certain probability of success, the repeated execution of GA is applied. The performance of the system is tested against a database of 138 images of cows in lateral position, obtaining a rate of successes up to 92 % in the best of the cases.

1 Introduction

Morphological assessment of bovine livestock is one of the most important parameters to be stored in the herd-books of pure breeds. This assessment process consist in scoring a series of concrete characteristics [1, 2] which have to do mainly with their morphology and the similarities with the standard description of the breed. As the morphological assessment is normally carried out by visual inspection [1], highly-qualified and experienced staff is required. Therefore, several methods has been recently proposed to build an automatic assessment system, all of them starting from some digital images of the animal [2, 3]. In our first approach, described in [3], the morphological assessment is based on the profiles of the animals in three different positions: lateral, frontal and rear. Hence, the first task to be done in that system is the parametrization and extraction of the animal boundaries.

Many different techniques has been described to extract boundaries in digital images [4, 5], from which stand out those based on template or model matching using energy minimization methods (deformable models [6]) or genetic algorithms (GAs) [7, 8, 5]. In fact, we successfully applied GAs to approximately search the boundary of the animals in a previous work [9], using a method similar to the approach designed by Hill et al. [10, 8]. However, with this method it is not possible to guarantee the success of the search in all the cases, what is very important for this technique to be used in practical applications.

In this work, as well as improving our system to extract contours with GAs, the method recently proposed by Yuen et al. [11] is applied, in order to guarantee a given probability of success by repeatedly applying a GA over the concrete problem (*repeated GA*). After obtaining the most suitable configuration for the GA in our problem, the number of executions needed to achieve a concrete probability of success is calculated. That calculus is based on the successes and failures obtained applying the single GA over a concrete training set, representative of the problem. The results obtained with this method notably improve those from previous works, nearly reaching the expected success rate. Though the repeated GA technique was applied in the three mentioned positions, we will focus only in lateral images. Further results can be consulted in `http://nernet.unex.es/cows`.

With this work outline, in section 2 the technique used to search the contours within the images is described in detail (shape modelling, objective function, etc.). Later on in section 3 the repeated GA methodology is presented, and the number of executions needed to guarantee a concrete probability of success is calculated. The results obtained applying the repeated GA to our database of images can be found in section 4, whereas conclusions and possible improvements of our work will be presented, finally, in section 5.

2 System Description

The contour extraction problem can be converted into an optimization problem considering two steps: the parametrization of the shape we want to search and the definition of a cost function that quantitatively determines whether or not a contour is adjusted to the object. The general outline of the system used is shown in figure 1. As illustrated, for the contour modelling and parametrization the *point distribution models* (PDMs) technique [6] has been applied, which let us restrict all the parameters and precisely define the search space. On the other hand, the objective function proposed is designed to place the contour over areas of the image where edges corresponding to the object are detected.

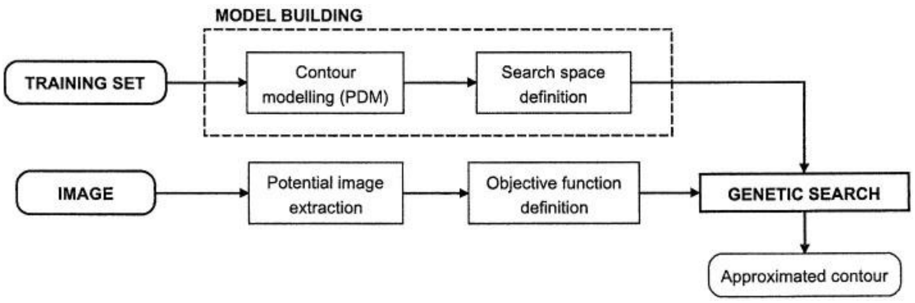


Fig. 1. General outline of the contour location system using genetic algorithms

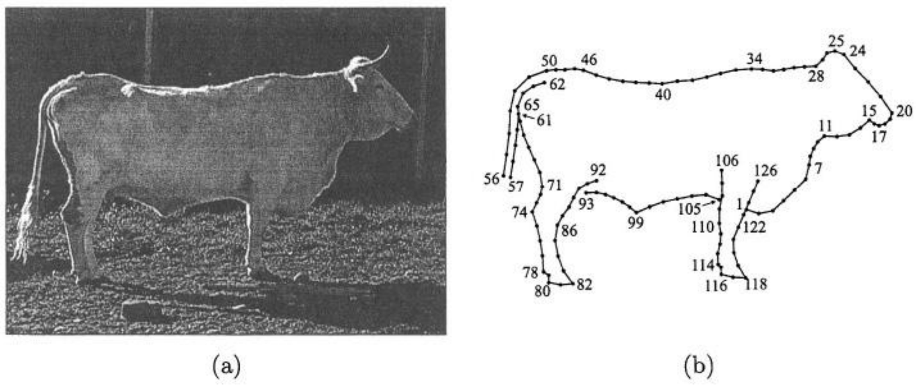


Fig. 2. Figure (a) illustrates the images dealt with. Figure (b) shows the model of the cow contour in lateral position

2.1 Shape Modelling and Search Space

As shown in figure 1, the first step consists in appropriately representing the desired shape we want to search along with its possible variations. In some previous works [7, 9] a general method known as PDM has been used, which consist of deformable models representing the contours by means of ordered groups of points (figure 2) located at specific positions of the object, that are constructed statistically, based on a set of examples. As a result of the process, the *average shape* of our object is obtained, and each contour can be represented mathematically by a vector $\boldsymbol{x}$ such as

$$\boldsymbol{x} = \boldsymbol{x}_m + \mathbf{P} \cdot \boldsymbol{b} \tag{1}$$

where $\boldsymbol{x}_m$ is the average shape, $\mathbf{P}$ is the matrix of eigenvectors of the covariance matrix and $\boldsymbol{b}$ a vector containing the weights for each eigenvector, and which properly defines the contour in our description. Fortunately, considering only few eigenvectors corresponding to the largest eigenvalues of the covariance matrix (*modes of variation,* using the terminology in [6]), practically all the variations taking place in the training set could be described.

This representation (model) of the contour is, however, in the normalized space. To project instances of this model to the space of our image a transformation that preserves the shape is required (translation $\boldsymbol{t} = (t_x, t_y)$, rotation θ and scaling s). Thus, every permitted contour in the image can be represented by a reduced set $\{s, \theta, \boldsymbol{t}, \boldsymbol{b}\}$ of parameters.

In order to have the search space precisely defined, the limits for this parameters are required. As stated in [6], to maintain a shape similar to those in the training set, b_i parameters must be restricted to values $-\sqrt{\lambda_i} \leq b_i \leq \sqrt{\lambda_i}$, where λ_i are the eigenvalues of the covariance matrix. The transformation parameters can be limited considering that we approximately know the position of the object within the image: normally the animal is in horizontal position (angle restriction), not arbitrarily small (restriction in scale) and more or less centered (restriction in $\boldsymbol{t}$).

In our specific case (lateral position), the cow contour is described using a set of 126 points (figure 2), 35 of which are significative. The model was constructed using a training set of 45 photographs, previously selected from our database of 138 images, trying to cover the variations in position as much as possible. For those 45 photographs the contours of the animals were extracted manually, placing the points of figure 2 by hand. Once the calculations were made, 20 modes of variation were enough to represent virtually every possible variation.

2.2 Objective Function and Potential Image Extraction

Once the shape is parameterized, the main step of this method is the definition of the objective function. In many consulted works [10, 7] this function is built so that it reaches a minimum when strong edges of similar magnitude are located near the points of a certain instance of the model in the image. In our case, we use a similar approach, but changing the method: our instances of the model are compared not with the image itself, but with a black and white image that we call *potential,* which presents light pixels in those positions toward which we wish our points to tend, and dark in the rest.

This potential image could be obtained using a conventional edge detector [9]. However, it is important, to avoid many local minima in the objective function, that the edge maps does not contain many more edges than those corresponding to the searched object, which is not always possible because of background objects. In our method, a system for edges selection, based on a multilayer perceptron neural network is applied. This network is trained using a set of images for which the edges of the searched object are known (in our case the 45 cattle images used to construct the model). The neural network acts over a previously obtained maximum edge map, classifying the edge pixels into two classes: edge pixels due to the searched object and edge pixels originated by objects in the background. As a result, only those edges that more likely correspond with the object we are looking for, according to the learned criterion, are selected. In figure 3 an example of this method is presented. Along with the original image (a), the maximum edge map and the results with and without the neural-networks-based edge selection system are shown. As we can see, in figure 3,c there are more edge pixel corresponding to the animal and less originated in the background than in figure 3,d. A more detailed description of this method can be found in [12]

With this potential image, we propose the following objective function to be maximized by the GA:

$$f_L(x) = \left(\sum_{j=0}^{n-2} K_j\right)^{-1} \cdot \sum_{j=0}^{n-2} \left(\sum_{(x,y)\in r_j} D(x,y) \right) \qquad (2)$$

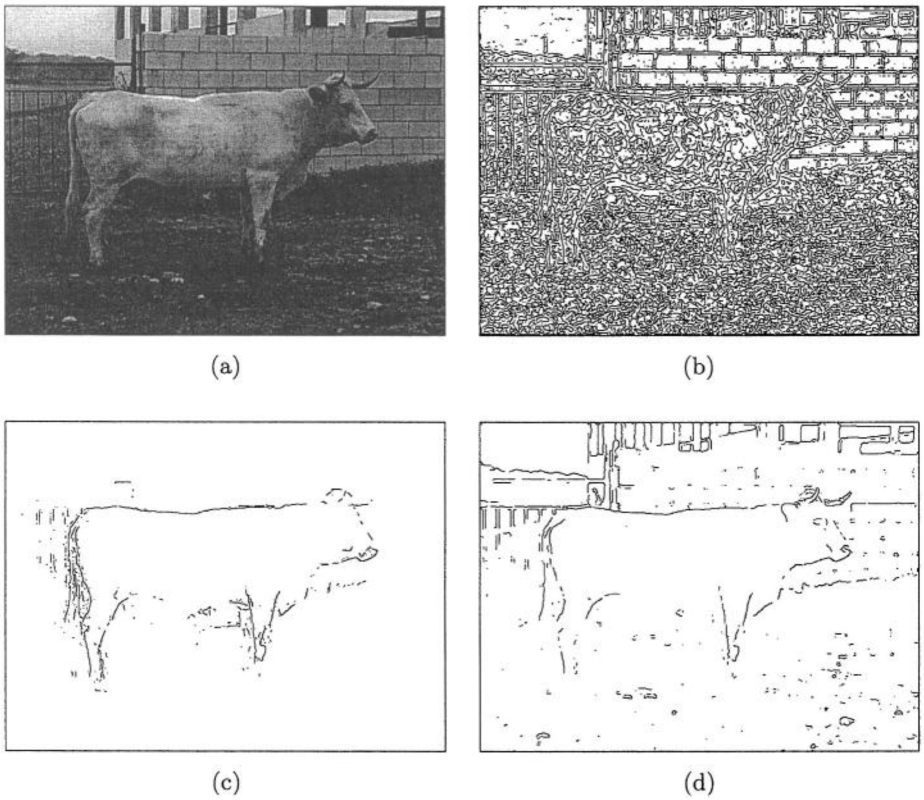


Fig. 3. In this figure, an example of our potential extraction method is presented: the original image (a), maximum edge map (b), and results with (c) and without (d) the neural-networks-based edge selection system are shown

where $\boldsymbol{x} = (x_0, y_0; x_1, y_1; \ldots; x_{n-1}, y_{n-1})$ is the set of points (coordinates of the image, integer values) that form one instance of our model, and can be easily obtained from the parameters using equation 1 and a transformation; $\boldsymbol{r_j}$ is the set of points (pixels of the image) that form the straight line which joins (x_j, y_j) and (x_{j+1}, y_{j+1}); K_j is the number of points that such a set contains; and $D(x, y)$ is a function of the distance from the point (x, y) to the nearest edge in the potential image.

The function $D(x, y)$ is defined to have a maximum value I_{max} for distance 0 and must be decreasing, so that a value less than or equal to $I_{max}/100$ is obtained for a given reach distance D_A. In our implementation, three different functions were considered (lineal, exponential and gaussian):

$$D_l(x, y) = I_{max}\left(1 - \frac{d(x, y)}{D_A}\right) \tag{3}$$

$$D_e(x,y) = I_{max} \cdot e^{-d(x,y) \cdot \ln 100 / D_A} \tag{4}$$

$$D_g(x,y) = I_{max} \cdot e^{-d(x,y)^2 \cdot \ln 100 / D_A^2} \tag{5}$$

where $d(x,y)$ is the distance from the point (x,y) to the nearest edge in the potential image. The function which produces the best results, along with the best value for D_A was selected using a GA-based method described in the following subsection. Finally, in order to avoid the calculation of distances within the objective function (that is executed a great number of times), a gray-level distance map is generated, over which the instances of our contour are projected.

2.3 Genetic Algorithm Configuration

After defining the search space and the objective function, two aspects must be determined in order to have an operating scheme of GA. On one hand, the general outline of the GA must be decided (coding, replacement and selection methods, etc). On the other hand, values for the parameters that determine the algorithm (population size, mutation rate, etc) must be given.

As there is no definitive conclusion about the best GA scheme to be used in this kind of problems, two configurations were considered, based on the ideas in [13]. The first is similar to the *basic GA* in [14], and we called it standard, while the other uses real coding, and is called *alternative*. They present the following characteristics:

– Standard configuration: binary coding (16 bits), roulette selection method (using raw fitness), two-point crossover and mutation operators, and steady-state replacement method.
– Alternative configuration: real coding, binary tournament selection method (using linear normalization to calculate fitness), two-point crossover and mutation operators, and steady-state replacement method.

To determine the most suitable scheme for our problem, along with the best values for the parameters involved (modes of variation t, distance function and reach distance D_A, population P, probabilities of crossover and mutation P_c and P_m, and replacement rate S), a secondary GA was applied, whose chromosome included those seven parameters, and whose objective function was related to the efficiency shown by the coded configuration over a concrete set of images, for which the correct position of the contour was known.

Concretely, the objective function had to run the primary GA, with the parameters encoded in the chromosome passed by the secondary GA, over a concrete number (five) of different images for which the correct position of the searched contour was known. Then, for each execution, the average distance between the points of the contour obtained and their correct position could be calculated. The average of the five values obtained in the five executions was used as the objective function of the secondary GA, and was minimized. It must be noted that we had a set of 45 images for which the correct position of the contour was known: those used in the construction of the shape model.

As a result, the best performance was obtained with the following GA configuration (alternative):

- **Modes of variation:** $t = 5$, so we have 9 parameters.
- **Distance dependence function:** gaussian.
- **Reach distance:** $D_A = 20$ pixels.
- **Coding:** Real numbers.
- **Population:** $P = 2000$ individuals.
- **Selection method:** Binary tournaments (with linear normalization).
- **Operators:** Mutation ($P_m = 0.4$) and two-points crossover ($P_c = 0.45$).
- **Replacement:** Steady-state replacement, $S = 60\%$.

It is remarkable that the best results are obtained with very few modes of variation (with five modes of variation around 85 % of the variations observed in the training set are explained), though up to 20 were considered.

3 Repeated Execution of the Genetic Algorithm

As mentioned in the introduction, the system described above presents the drawback of not assuring a certain rate of successes in the search. A similar problem is described in [11], where a method is proposed to guarantee a given probability of success by a repeated execution of the GA *(repeated GA)*. As stated in that paper, if the probability of success for one execution of the GA, P_{SGA}, is known, and N repeated runs of GA are made, then the probability of obtaining at least one success is $1 - (1 - P_{SGA})^N$. Hence, to reach a given probability P_{RGA} with repeated GA, N must be chosen such that:

$$N \geq \frac{\ln(1 - P_{RGA})}{\ln(1 - P_{SGA})} \tag{6}$$

Besides, to obtain an inferior limit P_{est} for P_{SGA}, a method based on a hypothesis test and an experiment is proposed. If c_1 is the number of successes obtained executing the GA over c_2 different and representative images, P_{est} can be calculated solving

$$\alpha = \sum_{x=c_1+1}^{c_2} b(x; c_2, P_{est}) \quad \text{with} \quad b(x; n, p) = \binom{n}{x} \cdot p^x \cdot (1-p)^{n-x} \tag{7}$$

where α is the level of significance required in the hypothesis test, and $b(x; n, p)$ is the binomial distribution with parameters n and p.

In order to apply this technique to our problem, the set of 45 images used for the shape modelling was considered, because the precise location for the points of the contour in those images was known. Then, one execution of GA was taken as a success if the mean distance between the points of the resulting contour and the points of the reference contour is smaller than 10 pixels, which is a reasonable distance that can be refined afterwards with local optimization

Table 1. Number (and percentage) of successes obtained applying both SGA and RGA to our database of 138 images. In the second row, the limbs have not been considered

	Single GA	*Repeated GA*
Dist. 7 points	85 (61.6 %)	96 (69.6 %)
Dist. 5 points	119 (86.2 %)	127 (92.0 %)

methods. After running the GA, 34 out of the 45 executions were successful. If we want to obtain $P_{RGA} = 0.95$, considering $\alpha = 0.001$ (value proposed in [11]), then N must be at least 4 executions. From the best individuals obtained in each of these N executions, the one which yields the best value for the objective function is selected as the final result.

4 Experiments and Results

Our database of 138 images (640 × 480 pixels size) corresponding to cows in lateral position was used to test the proposed system. In particular we were interested in testing the suitability of applying the repeated GA technique to our problem, the performance that could be achieved and, also, the problems that could arise.

With this aim, both the single GA (only one execution) and the repeated GA (four executions) were applied. In order to evaluate quantitatively a resulting contour, the mean distance (in pixels) between its points and their appropriate position must be calculated. However, it was very hard work to generate the contours for all the images (locating 126 points by hand), so the distance was estimated by using only seven very significative points of the contour. The points used were {20,25,40,50,78,99,114}, and are represented in figure 2. Those points were precisely located for all the images in the database.

Using that distance as a measure of the adjustment, the results obtained are presented in figure 4 and in the first row of table 1. In the mentioned graph the percentage of successes versus the distance considered to define success is represented. The successes increase rapidly with distance, though around 8 there is a change in the slope that makes the results for distance 10 to be lower than expected. In figure 6 some examples of the deformable template matching are presented. As we can see, cases a, b and c correspond to successful adjustments, with distances lower than 5 pixels, while cases d, e and f correspond to unsuccessful ones (distances over 15 pixels).

The results in the first row of table 1 were a bit disappointing, because a rate of success around 95 % should be expected, and less than 70 % was obtained. After analyzing the contours obtained, we realized that in many cases, though a great part of the contour was well–adjusted, there was an error between limbs in the foreground and in the background (figure 6, cases d and e). The position

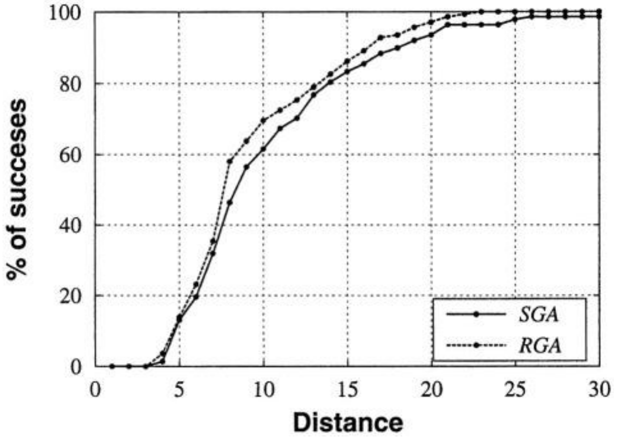


Fig. 4. In this figure, the percentage of successes versus the distance used to define success is represented, considering limbs points to calculate the distance

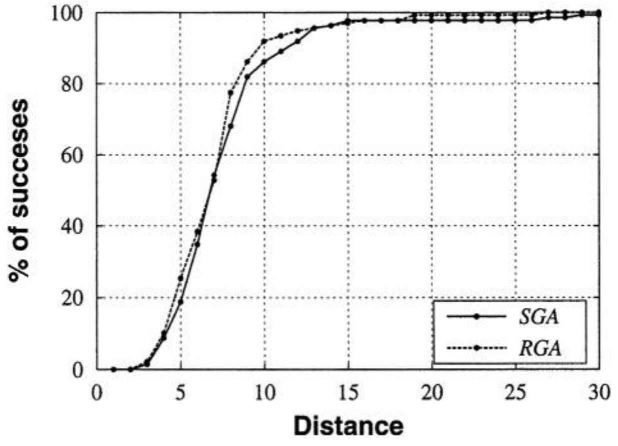
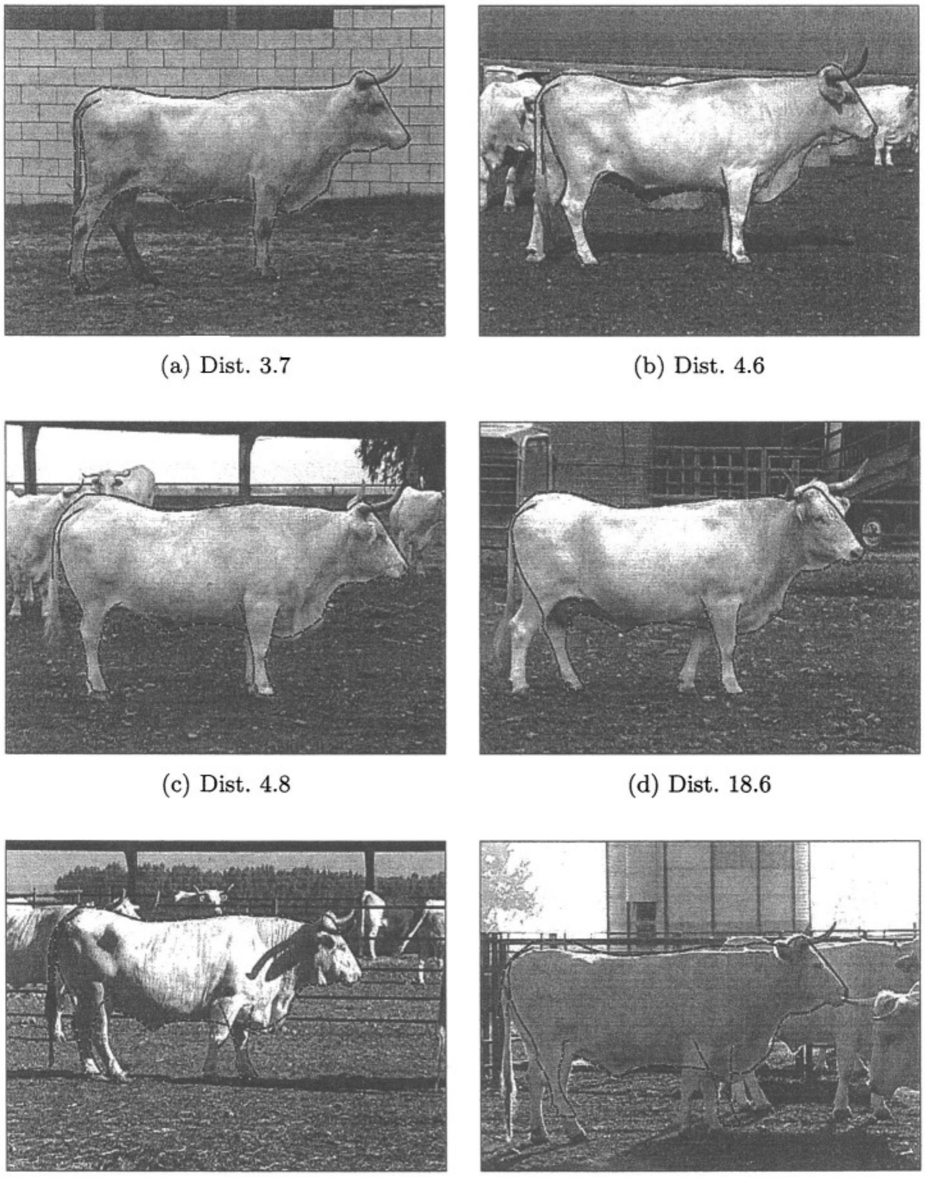


Fig. 5. As in figure 4, the percentage of successes versus the distance used to define success is represented, but now limbs points were not considered to calculate the distance

of the limbs generates an important problem, because in many images there are two (or more) locations for the template which yields a great value for the objective function, being impossible for the function to discriminate.

In order to determine the importance of this problem, we eliminated the influence of limbs position in the distance (i.e., we eliminated points 78 and 114 in that calculus). The new results obtained are shown in figure 5 and in



(a) Dist. 3.7

(b) Dist. 4.6

(c) Dist. 4.8

(d) Dist. 18.6

(e) Dist. 16.6

(f) 19.2

Fig. 6. In this figure some examples of the repeated GA application to the contour search are presented, along with the final mean distance (in pixels) to the correct position, considering the seven points described in the text

the second row of table 1. As illustrated, these results are significantly better, almost reaching 95 % for the repeated GA. In our examples, distance reduces to 9.4 and 8.9 in cases d and e respectively, being considered now as success.

These results show the applicability of the repeated GA method to our problem, and also the need to modify the objective function in order to avoid the problem generated by the limbs in background.

Also, the results obtained can be compared with those in [9], where a lateral contour without limbs was extracted. In that case, the best rate of success reported was 72.9, which is significantly smaller than the result obtained in this work. Besides, the method used in [9] to consider or not a success was based in visual inspection, what is less reliable than the method applied here.

5 Conclusions and Future Research

In this present work a system that enables accurate location of cow contours within digital images has been presented. The system is based in the repeated execution of a GA optimization, using an objective function involving a selective edge detection in the original image.

Though the results, at first sight, are not as good as we expected, it has been shown that the problem relies in the objective function: there is a systematic local maximum that appears due to the limbs in the background, which are not considered in the contour model. If we omit the limbs, the results presented are excellent. Therefore, we think that the most important part of the system that needs improvement is the objective function. We are now working in that direction, following two lines. On one hand, we try to obtain better potential images including more parameters as inputs of the neural network. On the other hand, we intend to include another term in the function to take into account the *a priori* knowledge of the problem. Besides, a multi–modal genetic search followed by a discriminating system is being considered.

On the other hand, the results in table 1 show that the improvement obtained with the repeated executions of the GA is not very large. This can be imputed to the good performance of the single GA in our problem. Anyway, thinking in the final application, what we are looking for with the repeated GA is to assure a certain probability of success, whatever the computational cost.

Acknowledgements

This work has been supported in part by the Junta de Extremadura through project 2PR01A007. We are indebted also to the staff of the *Centro de Selección y Reproducción Animal* (CENSYRA) for their technical assistance with the cattle and the photographs. Finally, we would like to express our gratitude to the reviewers, whose comments have been fundamental to improve the quality of the paper.

References

[1] Gottschalk, A.: Evaluación exterior del bovino. Hemisferio Sur S. A. (1993)

[2] Goyache, F., Del Coz, J. J., Quevedo, J.R., López, S.e.a.: Using artificial intelligence to design and implement a morphological assessment system in beef cattle. Animal Science **73** (2001) 49–60

[3] González, H.M., García, C.J., López, F. J., Macías, M., Gallardo, R.: Segmentation of bovine livestock images using GA and ASM in a two–step approach. Int. Journal of Pattern Recognition and Artificial Intelligence **17** (2003) 601–616

[4] Jain, A. K.: Fundamentals of Digital Image Processing. Prentice Hall (1989)

[5] Parker, J. R.: Algorithms for image processing and comp. vision. J. Wiley (1996)

[6] Cootes, T.F., Taylor, C. J., Cooper, D.H., Graham, J.: Active shape models – their training and application. Comp. Vision and Image Understanding **61** (1995) 38–59

[7] Hill, A., Cootes, T.F., Taylor, C.J., Lindley, K.: Med. image interpretation: a generic approach using def. templates. J. of Med. Informatics **19** (1994) 47–59

[8] Cootes, T. F., Hill, A., Taylor, C. J., Haslam, J.: The use of active shape models for locating structures in medical images. Image and Vision Computing **12** (1994) 355–366

[9] González, H. M., García, C. J., Macías, M., Acevedo, M. I.: GA techniques applied to contour search in images of bovine livestock. LNCS **2084** (2001) 482–489

[10] Hill, A., Taylor, C. J.: Model-based image interpretation using genetic algorithms. Image and Vision Computing **10** (1992) 295–300

[11] Yuen, S. Y., Fong, C. K., Lam, H. S.: Guaranteeing the probability of success using repeated runs of genetic algorithm. Im. and Vision Computing **19** (2001) 551–560

[12] García-Orellana, C. J., González-Velasco, H.M., Macías-Macías, M., Gallardo-Caballero, R., López-Aligué, F. J., Acevedo-Sotoca, I.: Edge selection using a neural classifier for colour image segmentation. In: Proc. Engineering Application of Neural Networks Conference EANN 2003. (2003) 161–168

[13] Davis, L.: Handbook of genetic algorithms. Van Nostrand Reinhold (1991)

[14] Goldberg, D. E.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley (1989)

On-the-Fly Training

Javier Melenchón, Lourdes Meler, and Ignasi Iriondo

Enginyeria i Arquitectura La Salle, Universitat Ramon Llull
Communications and Signal Theory Department
Pg. Bonanova 8, 08022, Barcelona, Spain
{jmelen,lmeler,iriondo}@salleurl.edu
http://www.salleurl.edu/eng/elsDCTS/tsenyal/index.htm

Abstract. A new algorithm for the incremental learning and non-intrusive tracking of the appearance of a previously non-seen face is presented. The computation is done in a causal fashion: the information for a given frame to be processed is combined only with the one of previous frames. To achieve this aim, a novel way for simultaneous and incremental computation of the Singular Value Decomposition (SVD) and the mean of the data is explained in this work. Previous developed methods about computing the SVD iteratively are taken into account and a novel way to extract the mean from a factorised matrix using SVD is obtained. Moreover, the results are achieved with linear computational cost and sublinear memory requirements with respect to the size of the data. Some experimental results are also reported.

1 Introduction

The last years have witnessed extraordinary advances in computer and communications technology, leading to an increasing availability of information and processing capabilities of multimedia data [1], [2]. This fact is resulting in a higher and wider demand for easier access to information [3]. On one hand, this information is mainly stored in digital format, so its acces is limited to the user's ability to communicate with computers. On the other hand, it has been remarked the great expressive power of the natural language used in human-human communication, as well as its intrinsic multimodal features [4]. Consequently, the acces to digital information could be carried out using this natural language: reducing the necessity of knowing a specific way to interact with the computer and taking advantage of its expressive features. Moreover, multimodal interfaces with an audio visual system like a talking head could be used in order to speak to the user in natural language. As a result, talking heads used in multimodal interfaces seem to be a proper solution for making acces to information easier and more pleasing for human users.

As explained in [4], multimodal input analysis is necessary when working with multimodal interfaces and relies on interaction devices e.g. facial trackers. Some non-intrusive visual trackers can be used in this sheme because they retain information regarding to position, scale, orientation and appearance of the tracked element, e.g. [5], [6], [7] and [8]. Nevertheless, the whole sequence is needed by these algorithms to be processed off-line (they have a non-causal behaviour); as a result, a real time implementation of these methods is impossible, even without considering their computational cost. This temporal restriction is caused by the computation of a Singular Value Decomposition (SVD) over the whole observed data. Moreover, memory resources are greatly affected by this fact, limiting the duration of the observed sequence. Incremental SVD computation techniques as [9], [10], [11] and [12] may be useful in this case, but they do not take into consideration the mean of the data, which is crucial in the classification of the different gestures. Fortunately, this is taken into account in [13], although it does not propose a method to extract the mean information from a given SVD and it can only update the SVD from two other known SVD.

In this paper, a new method for updating SVD and mean information as well as extracting the mean of the data contained in a given SVD without increasing the cost order of either time or memory is presented in Sect. 2. The application of this new method is carried out in Sect. 3 by a causal algorithm for the tracking and learning of the facial appearance of a person. Experimental results are given in Sect. 4 and concluding remarks are explained in Sect. 5.

2 Incremental SVD with Mean Update

2.1 Fundamentals

The singular value decomposition of matrix $\mathbf{M}_{p\times q} = [\mathbf{m}_1 \cdots \mathbf{m}_q]$ is given by:

$$\mathbf{M}_{p\times q} = \mathbf{U}_{p\times p}\mathbf{\Sigma}_{p\times q}\mathbf{V}^T_{q\times q} \ , \tag{1}$$

where $\mathbf{U} = [\mathbf{u}_1 \ \cdots \ \mathbf{u}_p]$ and $\mathbf{V} = [\mathbf{v}_1 \ \cdots \ \mathbf{v}_q]$ are orthonormal matrices; $\mathbf{u}_i$ are the eigenvectors of $\mathbf{M}\mathbf{M}^T$ and span the column space of $\mathbf{M}$; $\mathbf{v}_i$ are the eigenvectors of $\mathbf{M}^T\mathbf{M}$ and span the row space of $\mathbf{M}$; and $\mathbf{\Sigma}$ is a diagonal matrix with the singular values of either $\mathbf{M}\mathbf{M}^T$ and $\mathbf{M}^T\mathbf{M}$ in ascending order. Notice that if $\mathbf{M}$ is a rank r matrix, where $r \leq p$ and $r \leq q$, $\mathbf{\Sigma}$ has only r non-null singular values and (1) can be rewritten as the *thin SVD*: $\mathbf{M}_{p\times q} = \mathbf{U}_{p\times r}\mathbf{\Sigma}_{r\times r}\mathbf{V}^T_{q\times r}$. By the other hand, let $\mathbf{C}_{r\times q} = \mathbf{U}^T_{p\times r}\mathbf{M}_{p\times q}$ be the projections of the columns of $\mathbf{M}$ over the eigenspace spanned by $\mathbf{U}$. Using the *thin SVD* expression the projections matrix $\mathbf{C} = [\mathbf{c}_1 \ \cdots \ \mathbf{c}_q]$ can be written also as $\mathbf{C}_{r\times q} = \mathbf{\Sigma}_{r\times r}\mathbf{V}^T_{q\times r}$.

In other fields, like classification problems pointed by [13], a more suitable representation of $\mathbf{M}$ can be achieved including mean information $\overline{\mathbf{m}} = \frac{1}{q}\sum_{i=1}^{q}\mathbf{m}_i$ in (1), which has to be computed and substracted previously from $\mathbf{M}$ in order to be able to generate the SVD of $\mathbf{M} - \overline{\mathbf{m}} \cdot \mathbf{1}$:

$$\mathbf{M}_{p\times q} = \mathbf{U}_{p\times r}\mathbf{\Sigma}_{r\times r}\mathbf{V}^T_{q\times r} + \overline{\mathbf{m}}_{p\times 1}\mathbf{1}_{1\times q} \ . \tag{2}$$

2.2 Updating SVD

Assuming an existing SVD (1), if new columns $\mathbf{I}_{p\times c} = [\mathbf{I}_1 \ \cdots \ \mathbf{I}_c]$ are added in order to obtain a new matrix $\mathbf{M}'_{p\times(q+c)} = \left[\mathbf{M}_{p\times q} \ \mathbf{I}_{p\times c}\right]$, the SVD of $\mathbf{M}'$ can be updated from (1) using methods like [10] and [11], achieving:

$$\mathbf{M}'_{p\times(p+c)} = \mathbf{U}'_{p\times r'}\boldsymbol{\Sigma}'_{r'\times r'}\mathbf{V}'^{T}_{(q+c)\times r'} \ . \tag{3}$$

Otherwise, if the representation of $\mathbf{M}'$ is chosen to be as (2) and $\overline{\mathbf{m}}'$ is set to $\frac{1}{q+c}\left(\sum_{k=1}^{q}\mathbf{m}_k + \sum_{l=1}^{c}\mathbf{I}_l\right)$ the SVD becomes:

$$\mathbf{M}'_{p\times(q+c)} = \mathbf{U}'_{p\times r'}\boldsymbol{\Sigma}'_{r'\times r'}\mathbf{V}'^{T}_{(q+c)\times r'} + \overline{\mathbf{m}}'_{p\times 1}\mathbf{1}_{1\times(q+c)} \ . \tag{4}$$

Starting from (2) and matrix **I,** (4) can be obtained using the method proposed by [13] if the SVD of **I** is previously computed and q and c are known beforehand. A new method for updating the SVD and the mean using only the new observations and previous factorization is presented in Sect. 2.3.

2.3 Updating SVD and Mean

Begining with an existing factorization of $\mathbf{M}_i$ as in (5), it is desired to obtain the SVD and mean of $\mathbf{M}_f$ shown in (6):

$$\mathbf{M}_i = \mathbf{U}_i\boldsymbol{\Sigma}_i\mathbf{V}_i^T + \overline{\mathbf{m}}_i\mathbf{1} \ . \tag{5}$$

$$\mathbf{M}_f = \left[\mathbf{M}_i\ \mathbf{I}\right] = \mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_f^T + \overline{\mathbf{m}}_f\mathbf{1} \ . \tag{6}$$

Defining $\hat{\mathbf{M}}_i$ (7) and centering new columns **I** around $\overline{\mathbf{m}}_i$ (8), it can be written:

$$\hat{\mathbf{M}}_i = \mathbf{M}_i - \overline{\mathbf{m}}_i\mathbf{1} = \mathbf{U}_i\boldsymbol{\Sigma}_i\mathbf{V}_i^T \ . \tag{7}$$

$$\left[\mathbf{M}_i\ I\right] - \overline{\mathbf{m}}_i\mathbf{1} = \mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_f^T + \overline{\mathbf{m}}_f\mathbf{1} - \overline{\mathbf{m}}_i\mathbf{1} \ . \tag{8}$$

$$\left[\mathbf{M}_i - \overline{\mathbf{m}}_i\mathbf{1}\ \ \mathbf{I} - \overline{\mathbf{m}}_i\mathbf{1}\right] = \mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_f^T + \left(\overline{\mathbf{m}}_f - \overline{\mathbf{m}}_i\right)\mathbf{1} \ . \tag{9}$$

$$\left[\hat{\mathbf{M}}_i\ \hat{\mathbf{I}}\right] = \mathbf{U}_t\boldsymbol{\Sigma}_t\mathbf{V}_t^T \ . \tag{10}$$

Note that (10) is the updated SVD from (7) when some new observations $\hat{\mathbf{I}}$ are added. This update can be done as [11] suggests:

$$\left[\hat{\mathbf{M}}_i\ \hat{\mathbf{I}}\right] = \left[\mathbf{U}_i\ \mathbf{Q}_i\right]\begin{bmatrix}\boldsymbol{\Sigma}_i & \mathbf{U}_i^T\hat{\mathbf{I}}\\ \mathbf{0} & \mathbf{Q}_i^T\hat{\mathbf{I}}\end{bmatrix}\begin{bmatrix}\mathbf{V}_i^T & \mathbf{0}\\ \mathbf{0} & \mathbf{1}\end{bmatrix} = \left[\mathbf{U}_i\ \mathbf{Q}_i\right]\mathbf{U}_d\boldsymbol{\Sigma}_d\mathbf{V}_d^T\begin{bmatrix}\mathbf{V}_i^T & \mathbf{0}\\ \mathbf{0} & \mathbf{1}\end{bmatrix} = \mathbf{U}_t\boldsymbol{\Sigma}_t\mathbf{V}_t^T \tag{11}$$

where QR-decomposition is done to $\hat{\mathbf{I}} - \mathbf{U}_i\mathbf{U}_i^T\hat{\mathbf{I}} = \mathbf{Q}_i\mathbf{R}_i$ to obtain an orthogonal basis $\mathbf{Q}_i$ for the reconstruction error. Next, the *mean update algorithm* can be executed starting from the knowledge of $\mathbf{V}_t^T = \hat{\mathbf{V}}_t^T + \overline{\mathbf{v}}_t\mathbf{1}$, where $\overline{\mathbf{v}}_t = \frac{1}{q+c}\sum_{k=1}^{q+c}\mathbf{v}_k$:

$$\left[\hat{\mathbf{M}}_i\ \hat{\mathbf{I}}\right] = \mathbf{U}_t\boldsymbol{\Sigma}_t\hat{\mathbf{V}}_t^T + \mathbf{U}_t\boldsymbol{\Sigma}_t\overline{\mathbf{v}}_t\mathbf{1} = \mathbf{U}_t\boldsymbol{\Sigma}_t\hat{\mathbf{V}}_t^T + \overline{\mathbf{m}}_t\mathbf{1} \ . \tag{12}$$

$$\left[\hat{\mathbf{M}}_i\ \hat{\mathbf{I}}\right] = \mathbf{U}_t\boldsymbol{\Sigma}_t\mathbf{R}_v^T\mathbf{Q}_v^T + \overline{\mathbf{m}}_t\mathbf{1} = \mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_u^T\mathbf{Q}_v^T + \overline{\mathbf{m}}_t\mathbf{1} = \mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_f^T + \overline{\mathbf{m}}_t\mathbf{1} \ . \tag{13}$$

$$\left[\hat{\mathbf{M}}_i\ \hat{\mathbf{I}}\right] + \overline{\mathbf{m}}_i\mathbf{1} = \mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_f^T + \overline{\mathbf{m}}_t\mathbf{1} + \overline{\mathbf{m}}_i\mathbf{1} \ . \tag{14}$$

$$\left[\mathbf{M}_i\ \mathbf{I}\right] = \mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_f^T + \overline{\mathbf{m}}_f\mathbf{1} \ . \tag{15}$$

It is assumed $\mathbf{Q}_v\mathbf{R}_v$ as the QR-decomposition of $\hat{\mathbf{V}}_t$, $\mathbf{U}_f\boldsymbol{\Sigma}_f\mathbf{V}_u^T$ as the SVD of $\mathbf{U}_t\boldsymbol{\Sigma}_t\mathbf{R}_v^T$ and $\overline{\mathbf{m}}_f = \overline{\mathbf{m}}_t + \overline{\mathbf{m}}_i$. Note that (15) and (6) are equivalent.

2.4 Mean Extraction from a Given SVD

The previous method can also be used to extract the mean information from an existing SVD, e.g. trying to express $\mathbf{S} = \mathbf{U}_t\mathbf{\Sigma}_t\mathbf{V}_t^T$ as $\mathbf{S} = \mathbf{U}_f\mathbf{\Sigma}_f\mathbf{V}_f^T + \bar{\mathbf{s}} \cdot \mathbf{1}$ setting $\left[\hat{\mathbf{M}}_i \; \hat{\mathbf{I}}\right] = \mathbf{S}$ and $\overline{\mathbf{m}}_t = \mathbf{0}$ in (12) to (15).

2.5 Time and Memory Complexity

The mean update presented in section 2.3 does not increase the order of resources required in methods of incremental SVD developed in [9], [11], [10], [12] and [13]. The computational cost becomes $O\left(qr^2 + pr^2\right)$ and the memory complexity is $O\left(pr + qr\right)$, as shown in Table 1.

3 On-the-Fly Face Training

In this paper, *On-the-fly Face Training* is defined as the process of learning the photo-realistic facial appearance model of a person observed in a sequence in a rigorous causal fashion. This fact means that it is not necessary to take into account subsequent images when adding the information of the current one, which is considered only once. Note that the facial appearance is learnt in the same order as the captured images, allowing a real-time learning capability in near future, as computational resources are constantly being increased.

3.1 Data Representation

An N image sequence $\mathbf{S} = [\mathbf{I}_1 \cdots \mathbf{I}_N]$ and a set of four masks $\mathbf{\Pi}= \left\{\pi^1, \ldots, \pi^4\right\}$, attached to four facial elements (like mouth, eyes or foerehead), are given. For each image $\mathbf{I}_t$, its specific mouth, eyes and forehead appearance are extracted with $\mathbf{\Pi}$, obtaining four observation vectors $\mathbf{o}_t^r$ (see Fig. 1). Therefore, four observation matrices $\mathbf{O}^r$ can be obtained from the application of the set of masks $\mathbf{\Pi}$ over the sequence $\mathbf{S}$. Dimensionality reduction of $\mathbf{O}^r$ can be achived using

Table 1. Resource order requirements of the proposed SVD mean update algorithm

Operation	Comp. cost	Mem. requirements
$\mathbf{V}_{q\times r}^T - \left(\frac{1}{q}\sum_{k=1}^{q}(\mathbf{v}_k)_{r\times 1}\right)\mathbf{1}_{1\times q} \rightarrow \hat{\mathbf{V}}_{q\times r}^T$	$O\left(qr\right)$	$O\left(qr + r\right)$
$\hat{\mathbf{V}}_{q\times r} \rightarrow (\mathbf{Q}_v)_{q\times r}(\mathbf{R}_v)_{r\times r}$	$O\left(qr^2\right)$	$O\left(qr + r^2\right)$
$(\mathbf{U}_i)_{p\times r}(\mathbf{\Sigma}_i)_{r\times r}\left(\mathbf{R}_v^T\right)_{r\times r} \rightarrow \mathbf{T}_{p\times r}$	$O\left(pr^2 + r^3\right)$	$O\left(pr + r^2\right)$
$\mathbf{T}_{p\times r} \rightarrow (\mathbf{U}_f)_{p\times r}(\mathbf{\Sigma}_f)_{r\times r}\left(\mathbf{V}_u^T\right)_{r\times r}$	$O\left(pr^2\right)$	$O\left(pr + r^2\right)$
$(\mathbf{V}_f)_{q\times r} \rightarrow (\mathbf{Q}_v)_{q\times r}(\mathbf{V}_u)_{r\times r}$	$O\left(qr^2\right)$	$O\left(qr + r^2\right)$
Totals, assuming $p \gg r$ and $g \gg r$	$O\left(qr^2 + pr^2\right)$	$O\left(pr + qr\right)$

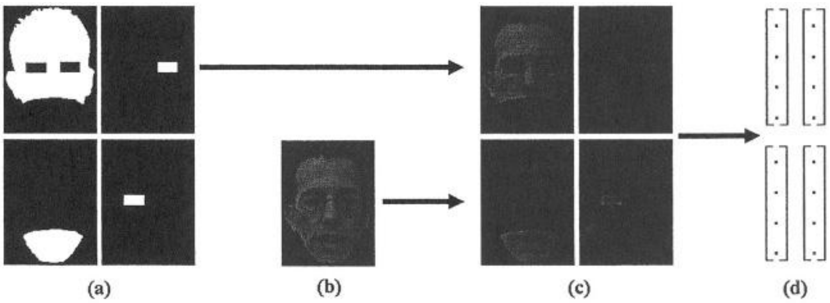


Fig. 1. (*a*) Masks π^r. (*b*) Image $\mathbf{I}_t$. (*c*) Regions $\mathbf{R}_t^r$, obtained from the application of each mask π^r over image $\mathbf{I}_t$. (*d*) Vectors $\mathbf{o}_t^r$ related to the defined regions

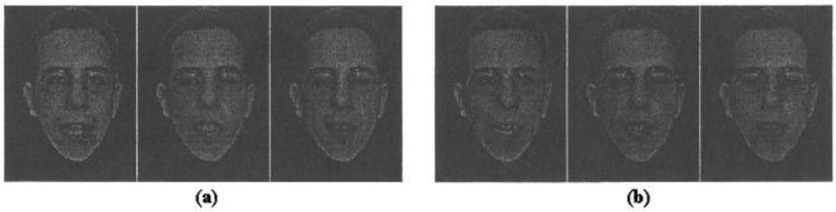


Fig. 2. (*a*) Three observed frames of a subject's face, (*b*) The same synthesized frames after learning the appearance model of this person

SVD [14]: $\mathbf{O}^r = [\mathbf{o}_1^r \cdots \mathbf{o}_N^r] = \mathbf{U}^r \mathbf{\Sigma}^r (\mathbf{V}^r)^T + \overline{\mathbf{o}}^r \mathbf{1}_{1\times N}$, where $\overline{\mathbf{o}}^r = \frac{1}{N}\sum_{k=1}^{N} \mathbf{o}_k^r$. Note that facial element appearances can be parameterized as $\mathbf{C}^r = \mathbf{\Sigma}^r (\mathbf{V}^r)^T$ (see Sect. 2.1). In the example proposed in this paper, faces composed of 41205 pixels could be codified with 35 coefficients, representing a reduction of more than 99.9% without any loss of perceptual quality (see Fig. 2).

3.2 Training Process

One major drawback of the parametrization presented in Sect. 3.1 consists in the image alignment of the sequence [5]. Unless all face images through the whole sequence have the same position, ghoslty results may appear and suboptimal dimensionality reduction will be achieved. The tracking scheme presented in this paper combines simultaneously both processes of learning and alingment. First of all, the four masks π^r are manually extracted from the first image $\mathbf{I}_1$ of sequence $\mathbf{S}$ and the first observation vectors $\mathbf{o}_1^1, \ldots, \mathbf{o}_1^4$ are obtained. Next, the corresponding alignmentcoefficients $\mathbf{a}_1$ are set to $\mathbf{0}$; they represent the affine transformation used to fit the masks onto the face on each frame [5]. Using the tracking algorithm presented in [7] over the second image $\mathbf{I}_2$, observations $\mathbf{o}_2^1, \ldots, \mathbf{o}_2^4$ and alignment coefficients $\mathbf{a}_2$ are stored. At this point, each facial element r can be factorized as $[\mathbf{o}_1^r\,\mathbf{o}_2^r] = \mathbf{O}_2^r = \mathbf{U}_2^r \mathbf{\Sigma}_2^r (\mathbf{V}_2^r)^T + \overline{\mathbf{o}}_2^r = \mathbf{U}_2^r (\mathbf{C}_2^r)^T + \overline{\mathbf{o}}_2^r$, where the current mean observation is generated by $\overline{\mathbf{o}}_2^r = \frac{\mathbf{o}_1^r + \mathbf{o}_2^r}{2}$, the eigenvectors

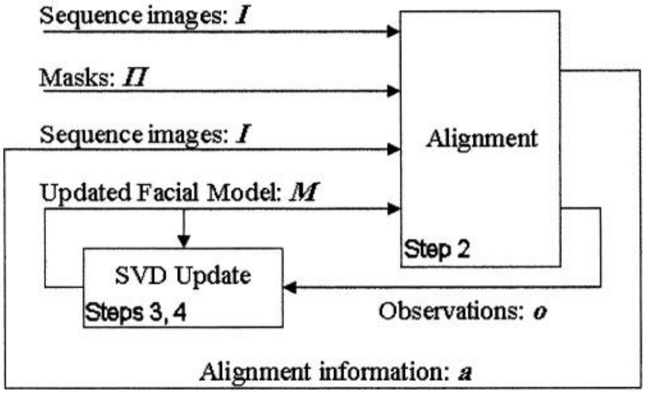


Fig. 3. Block diagram of the *On-the-fly Training Algorithm*

of $O_2^r (O_2^r)^T$ are found in U_2^r and the texture parametrization of the r-th facial element in images I_1 and I_2 is obtained in C_2^r. Once this initialization is done, the *On-the-fly Training Algorithm* (Figure 3) can be executed. Besides, only those columns of U_{t+1}^r and V_{t+1}^r whose values of Σ_{t+1}^r exceed a threshold τ are considered, keeping only those eigenvectors with enough information. The value of τ decreases from 0,5 to $0,5 \cdot 10^{-3}$ in the first images (1 seconds at 25 im/s) in order to allow better face localization when almost no information is known about its appearance. Notice that alignment parameters **a** can be used to extract gestural information in a multimodal input system [15].

On-the-fly Training Algorithm

```
In: $U_2, \Sigma_2, V_2, \bar{o}_2$, alignment coefficients $a_2$ and set of four masks $\Pi$
  1. Set $k = 2$
  2. Using $U_k, \Sigma_k, V_k, \bar{o}_k$, $a_k$ and $\Pi$, a new input image $I_{k+1}$ is aligned,
     generating $L$ observation vectors $o_{k+1}^r$ and updating the alignment
     information to produce $a_{k+1}$.
  3. Obtain $U_{k+1}, \Sigma_{k+1}, V_{k+1}$ and $\bar{o}_{k+1}$ from $U_k, \Sigma_k, V_k, \bar{o}_k$, and $o_{k+1}$ (4)-(8).
  4. Trim $U_{k+1}$ and $V_{k+1}$ according to $\Sigma_{k+1}$.
  5. Set $k = k + 1$ and go to Step 2 until there is no more new images.
Out: $U_f, \Sigma_f, V_f, \bar{o}_f$ and alignment coefficients $a_f$ for each image.
```

4 Experimental Results

The *On-the-fly Training Algorithm* has been tested over a short sequence and a long one, both recorded at a frame rate of 25 im/s. The short sequence consists of 316 images and it has been used to compare the results obtained from our *On-the-fly Training Algorithm* and its previous non-causal version [7]. Achieving the same quality in the results (see Fig. 4), the presented algorithm has reduced the

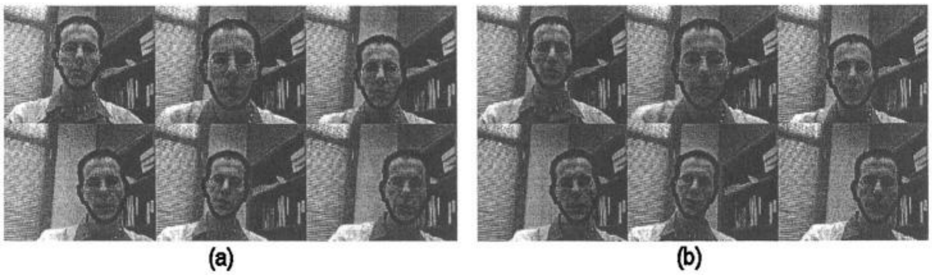


Fig. 4. Output tracking results of the learning process for: (*a*) the *On-the-fly Training Algorithm* and (*b*) the non-causal algorithm

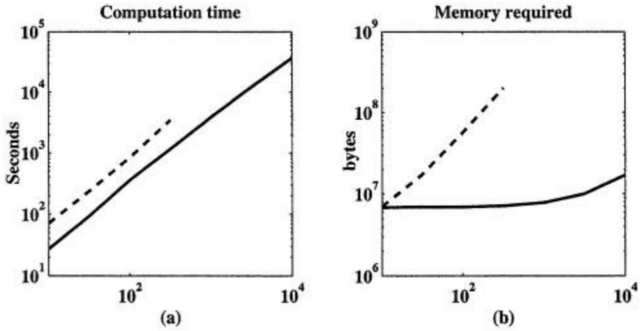


Fig. 5. Solid line represents the *On-the-fly Training Algorithm* performance while the dashed one belongs to the non-causal algorithm presented in the work of [7]. (*a*) Computation time in seconds. (*b*) Memory used in bytes

execution time about 66% with respect to [7] and has required about 7 Mbytes in front of the 200 Mbytes consumed by [7] (see the comparison in Fig. 5). Later, if we focus on the long sequence (10000 frames), its processing requirements were impossible to met with the non-causal algorithm [7] because its huge memory cost of 6000 Mbytes, although massive storage systems (e.g. hard drives) were used; the *On-the-fly Training Algorithm* reduced the memory requirements to 17 Mbytes with a processing time of a little more than 10 hours (using a 2GHz processor) (see Fig. 5).

5 Concluding Remarks

In this paper, a new method for extracting the mean of an existing SVD is presented, without increasing either the cost order of memory or time. Thus, incremental computation of SVD preserving a zero data mean is allowed. Fields that can benefit from this method can be, e.g.: classification problems, where the mean information is used to center the data; incremental computation of covariation matices, which need to be centered around its mean; causal construction of eigenspaces, where the principal components of the data are included, as well as the mean information. With respect to the latter, the *On-the-fly Algorithm* is presented in this work. Given an image sequence and a set of masks, this algorithm is capable of generating a separate eigenspace for each facial element (learning all their appearance variations due to changes in expression and visual utterances) and effectively tracking and aligning them. Furthermore, longer sequences than previous methods [5], [7] can be processed with the same visual accuracy when no ilumination changes appear. Finally, we plan to add more robustness to this algorithm using methods like [5] and more work will be done in order to achieve real time perfomance, so specific appearance models can be obtained as a person is being recorded.

References

[1] E. André: The generation of multimedia presentations. A Handbook of Natural Language Processing: techniques and applications for the processing of language as text, R. Dale, H. Misl and H. somers, Eds., Marcel Dekker Inc., 2000, 305-327

[2] S. Robbe-Reiter, N. Carbonell and P. Dauchy: Expression constraints in multimodal human-computer interaction. Intelligent User Interfaces, 2000, 225-228

[3] C. Stephanidis: Towards universal acces in the disappearing computer environment. UPGRADE, vol. IV, n. 1, 2003, pp. 53-59

[4] E. André: Natural language in multimedia/multimodal systems. Handbook of Computational Linguistics, R. Miktov, Ed., Oxford Univ. Press, 2003, 650-669

[5] F. de la Torre and M. Black: Robust parameterized component analysis: Theory and applications to 2d facial modeling. ECCV, 2002, 654-669

[6] T. Ezzat, G. Geiger and T. Poggio: Trainable videorealistic speech animation. ACM SIGGRAPH, San Antonio, Texas, July 2002, 225-228

[7] J. Melenchón, F. de la Torre, I. Iriondo, F. Alías, E. Martínez and Ll. Vicent: Text to visual synthesis with appearance models, ICIP, 2003, vol I, 237-240

[8] D. Cosker, D. Marshall, P. Rosin and Y. Hicks: Video realistic talking heads using hierarchical non-linear speech-appearance models. Mirage, France, 2003

[9] M. Gu and S. C. Eisenstat: A Stable and fast algorithm for updating the singular value decomposition. Tech. Rep. YALEU/DCS/RR-966, New Haven, 1993

[10] S. Chandrasekaran, B. Manjunath, Y. Wang, J. Winkeler and H. Zhang: An eigenspace update algorithm for image analysis. GMIP, vol. 59, no. 5, 1997, 321-332

[11] M. Brand: Incremental singular value decomposition of uncertain data with missing values. ECCV, 2002, I: 707 ff.

[12] D. Scočaj and A. Leonardis: Incremental approach to robust learning of subspaces. ÖAGM, 2002, 71-78

[13] P. M. Hall, D. R. Marshall and R. Martin: Adding and substracting eigenspaces with eigenvalue decomposition and singular value decomposition. ICV, vol. 20, no. 13-14, december 2002, 1009-1016

[14] M. Kirby: Geometric Data Analysis: An Empirical Approach to Dimensionality Reduction and the Study of Patterns. John Wiley & Sons Inc., 2001

[15] F. Keates and P. Robinson: Gestures and multimodal input. Behaviour and Information Technology, Taylor and Francis Ltd., 1999, 18(1), 35-42

Deformable Object Matching Based on Multi-scale Local Histograms

N. Pérez de la Blanca[1], J.M. Fuertes[2], and M. Lucena[2]

[1]Department of Computer Science and Artificial Intelligence
ETSII. University of Granada, 18071 Granada, Spain
nicolas@ugr.es
[2]Departmento de Informática. Escuela Politécnica Superior. Universidad de Jaén
Avenida de Madrid 35, 23071 Jaén .Spain
{jmf,mlucena}@ujaen.es

Abstract. This paper presents a technique to enable deformable objects to be matched throughout video sequences based on the information provided by the multi-scale local histograms of the images. We shall show that this technique is robust enough for viewpoint changes, lighting changes, large motions of the matched object and small changes in rotation and scale. Unlike other well-known color-based techniques, this technique only uses the gray level values of the image. The proposed algorithm is mainly based on the definition of a particular multi-scale template model and a similarity measure for histogram matching.

1 Introduction

In this paper, we approach the problem of matching deformable objects through video sequences based on the information provided by the gray level histogram of the local neighborhoods of the images. Our approach is traditional in the sense that we shall define a template of the object of interest, and we will attempt to find the image region that best matches the template. What is new about our approach is the template definition and the similarity measure. Deformable object matching/tracking remains a very challenging problem mainly due to the absence of good templates and similarity measures which are robust enough to handle all the geometrical and lighting deformations that can be present in a matching process.

Very recently, object recognition by parts has been suggested as a very efficient approach to recognize deformable object [4][5][1]. Different approaches are used in the recognition process from the basic parts, but the matching of salient parts is a common task to all approaches. Region and contour information are the main sources of information from which the location of a part of an object in an image can be estimated (e.g. [17][11][9][10]). Our approach is region-based since gray level features better model the type of application that we are interested in. Let us consider facial region matching. The main features we use are the local histograms at different spatial scales of the image. Furthermore, it is well known that histograms are robust features for translation, rotation and view point changes [14] [15] .

The use of histograms as features of interest can be traced back to Swain & Ballard [15] who demonstrated that color histograms could be used as a robust and efficient mechanism for indexing images in databases. Histograms have been used widely in object and texture recognition and image and video retrieval in visual databases [13] [3] [12]. The main drawback of using the histogram directly as the main feature is the loss of the gray level spatial information [12][15]. Recent approaches based on the space-scale theory have incorporated the image's spatial information. In [13] multidimensional histograms, which are obtained by applying Gaussian derivative filters to the image, are used. This approach incorporates the image's spatial information with global histograms. In [3], spatial information is also taken into account, but using a set of intensity histograms at multiple resolutions. None of the above approaches explicitly addresses the local spatial information present in the image. The ideas presented in [7], [6] suggest the interest in removing the local spatial information in deformable regions matching process. In [8] it is shown that very relevant information to detect salient regions in the image can be extracted from local histograms at different scales.

In this paper, by contrast with the above approaches we impose a better compromise between spatial information and robustness to deformations. In our case, the matching template for each image region is built as a spatial array, and to each of its positions, the gray level histograms (calculated from a growing sequence of spatial neighborhoods centered on this position) are associated. Although this spatial information is extremely redundant, as we show in the experiment in the case of high noise, this redundancy is extremely useful when estimating the correct location of the template. On each image, the template is iterated on all the possible locations within it. The matching on each image location is the vector of the similarity matching on each spatial scale. The optimum (minimum or maximum, according to the similarity criterion definition) of this vector defines the saliency value in each image location. The set of these values defines a saliency map associated to the image, which is the input to the final decision criteria defining the optimum location.

This paper is organized in the following way: Section 2 introduces the template definition and the similarity measure; Section 3 presents the algorithm; Section 4 shows the experimental results; and finally, Section 5 details the discussion and conclusions.

2 Template and Similarity Measure

Let $\mathcal{R}$ be a region of the image *I*. Let $\mathbf{D}_s(\mathbf{a})=\{\mathbf{x}\in \mathcal{R} \mid \|\mathbf{x}-\mathbf{a}\|<s, \mathbf{a}\in \mathcal{R}, s\in \mathrm{R}^+\}$ be the set of points inside the region $\mathcal{R}$ to a distance s of the points **a**. Let $\mathbf{N}_s=\{ \mathbf{D}_s(\mathbf{a}) \mid \mathbf{D}_s(\mathbf{a}) \subset \mathcal{R}\}$ be the set of all local discs $\mathbf{D}_s$ fully contained inside the region $\mathcal{R}$. The set $\{\mathbf{N}_s, s\in \mathcal{S}\}$ represents the information present in the image for the range of scales defined by $\mathcal{S}$. The main drawback of classical template matching methods is the rigidity imposed by the spatial disposition of the gray level values of the pixels, which prevent the template from adapting to the gray level changes on the surface of the object due to lighting or geometrical changes. The template we introduce to characterize a region removes this constraint by associating to each pixel a set of histograms instead of only one gray level. Obviously, small changes in the gray level values around a pixel can be absorbed inside its histogram.

The template associated to a region $\mathcal{R}$ is then defined by the set

$$\mathcal{T}(\mathcal{R}) = \{\mathcal{V}(\mathrm{N}_s), s \in \mathcal{S}\} \tag{1}$$

where $\mathcal{V}$ ($\mathbf{N}_s$) represents the set of histograms built up from the set of spatial neighborhoods $\mathbf{N}_s$.

The next step is to define a similarity measure associated to this template. We have considered the following vector-valued similarity distance between two templates associated to two regions of the same size

$$\mathcal{D}(\mathcal{T}(\mathcal{R}_1), \mathcal{T}(\mathcal{R}_2)) = \min_{s \in S}\left\{\frac{1}{M_s}\sum_{x \in \mathrm{N}_s} w(x)\left\|v_1(x,s) - v_2(x,s)\right\|\right\} \tag{2}$$

Where $v_i(x,s)$ defines the gray level histogram calculated at the pixel location x and scale value s and M_s is the number pixel in $\mathbf{N}_s$. The values $w(x)$ weight the error norm inversely proportional to the distance of the pixel x to the target region center. In our case we have use the radially symmetric function $w(x)=(1-x)$, if $0 \leq x \leq 1$ and 0 if $x>1$ to define the weights. Different norms calculating a distance between two dense histograms, in matching process, have already been studied [13]. In our case, all the local histograms $v(x,s)$ are very sparse since the range of gray levels present in the neighborhood of each pixel is usually very small in comparison with the full range of the image. We have experimented with the Minkowski's norm (M) for p=1,2 and the capacitory discrimination (C), an entropy-based discrimination measured equivalent

$$M_p(v_1,v_2) = \left\|v_1 - v_2\right\|_M = \left(\sum_{n \in bin}\left|v_{1n}(x,s) - v_{2n}(x,s)\right|^p\right)^{\frac{1}{p}}, \mathrm{p} = 1,2$$

$$C(v_1,v_2) = \left\|v_1 - v_2\right\|_C = 2H\left(\frac{v_1+v_2}{2}\right) - H(v_1) - H(v_2) \tag{3}$$

$$H(\mathbf{p}) = -\sum_{n \in bin} p_n \log p_n$$

to a simetrized divergence information (see [16]).

One important consequence of the histogram sparseness is the need to quantize the image gray level range before the similarity distances are calculated. It is important to note that in contrast with the results shown in [15], the bin number after the quantization process appears as a relevant parameter. We have tried to estimate this number using the statistical criteria developed for optimum bin number estimation [2], but unfortunately these estimators do not represent, in general, an adequate bin number for the matching process. In our case the bin number used in the experiments was fixed by hand. A consequence of the quantization process is the invariance to illumination differences less than the bin width. In all of our experiments, we use a uniform quantization criterion fixing the length of the interval of the gray levels of the image assigned to each bin. The same process is applied to the gray levels of the template region.

In order to estimate $\mathcal{L}(\mathcal{T}(\mathcal{R}_T),\mathcal{T}(I))$, the set of possible occurrences of the template in an image, we apply the function defined in (2) on each scale and on all the possible image locations in which the template region can be embedded within the image. These values define the saliency vector-map associated to the image I. The set $\mathcal{L}$ is defined by the union of all spatial local minima present on each scale of the saliency map.

$$\mathcal{L}(\mathcal{T}(\mathcal{R}_T),\mathcal{T}(I)) = \bigcup_{s\in S}\left\{x\in I \,\middle|\, \mathcal{D}_{\mathcal{R}_T}(\mathcal{T}(I_x)) < \mathcal{D}_{\mathcal{R}_T}(\mathcal{T}(I_y))\ \forall y\in \mathbf{D}_s(x)\right\} \tag{4}$$

where $\mathcal{D}_{\mathcal{R}_T}(\mathcal{T}(I_x))$ means $\mathcal{D}(\mathcal{T}(\mathcal{R}_T),\mathcal{T}(I_x))$, and the subscript x indicates the image point where the template is centered. More sophisticated matching techniques can be applied on a subset of these points to decide the best of all of them.

3 The Algorithm

The previous steps can be summarized as follows:

```
1.- Fix the scale range.
2.- Build up the template $\mathcal{T}(\mathcal{R}_T)$ of the region template for the prefixed range of scales.
3.- For each image
    3.1 Build up the template of the i-th image $\mathcal{T}(I_i)$
    3.2 Calculate the saliency map between $\mathcal{T}(\mathcal{R}_T)$ and $\mathcal{T}(I_i)$
    3.3 Calculate $\mathcal{L}$ , the union of all local minima on all scales for a prefixed
        neighborhood size.
4.- Apply a final matching criterion on the set of point $\mathcal{L}$.
```

In order to speed up the efficiency of the algorithm we start applying the algorithm to a sub-sampled version of the region template and images, where the scale values were divided accordingly. In this case, the estimated points and its neighbourhoods for a fixed size define the set of point $\mathcal{L}$. In order to get the maximum accuracy in our matching process the step 4 is carried out on the original images. It is also important for the efficiency in time to implement the histogram calculation using an adaptive process along all the image locations. The most costly step in this algorithm is the saliency map calculation on each image location. In this respect and taking into account the information redundancy present in the template, the error measure given in (2) can only be calculated on a subset of the pixel. In order to remove the produced error by a constant difference in illumination, the template and the target region are fixed to the same average value before calculating the error differences.

4 Experimental Results

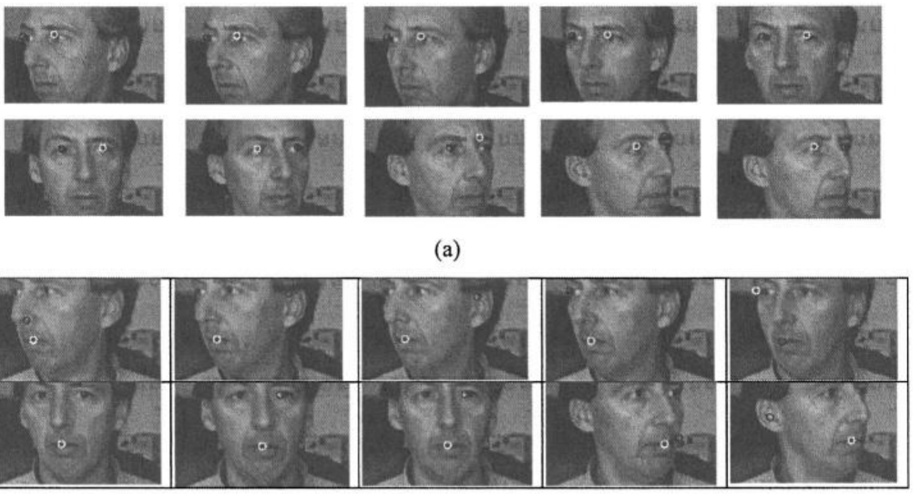
Several experiments have been performed in order to assess the effectiveness of the proposed algorithm. Firstly, we focus our experiments to show how robust our algorithm is to drastic changes in the object pose. Secondly, we also show how the algorithm is capable of a reasonable level of shape generalization, since with only one sample it is possible to successfully track different instances of the same kind of object. Thirdly, we show how robust our algorithm is when there are a very large change in pose and very hard noise condition. In all the experiments, the final decision criterion has been to take the location with the highest saliency measure. In all the experiments, the scale range used was s=2,3,4,5,6, which means they are circular neighborhoods with diameters ranging from 3 to 11 pixels. In all the experiments, the template region is a rectangular sub-image. The background pixels, when present, are removed using a binary mask. The bin number was fixed on each experiment by hand. The three distances between histograms, $\mathbf{M_1}$, $\mathbf{M_2}$ and C, were used in our experiments. Although no very significant differences were detected among them, the Euclidean distance $\mathbf{M_2}$ obtained in all the experiemnts the more accurate matches. All results shown in this section are referred to the Euclidean distance.

We have used video sequences of human heads in motion for the first two experiments, and sequences obtained from the national TV signal for the third experiment. The head in motion sequences were captured in 640x480 format by the same digital camera, but in different lighting conditions. The aim is to match the eyes and the mouth throughout the entire sequences. In our experiments, the template region was an instance of the matched object chosen from an image of the sequence. However, we also show the results of using the generic template region on different image sequence. For reasons of efficiency in our experiments, we reduce the image size to the head zone giving 176x224 size images.

Figure 2 shows the images of a person turning his head from right to left and vice versa. In this case, the region templates for the eyes and mouth, respectively, have been taken from the region of the right eye and the mouth of one of the central images with the head in a front-to-parallel position. Figure 1 shows the two region templates

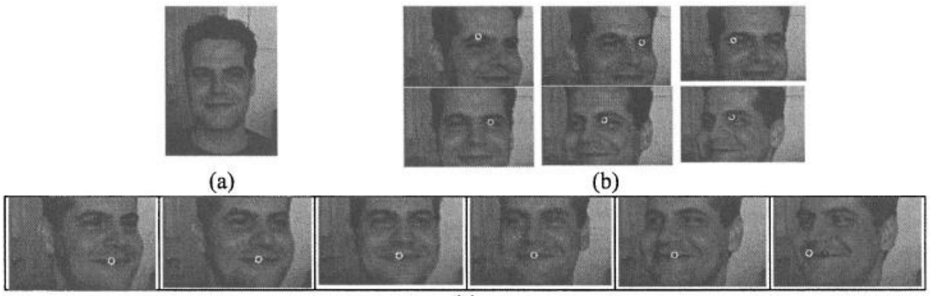
Fig. 1. a) Template region used for tracking the eyes; 24x16 pixels, b) Template region used for tracking the mouth; 36x16 pixels. Both template regions belong to the sequence shown in figure 2

Figure 3. shows the results obtained using generic template regions obtained form figure 3 (a) on an image sequence with different expression and very strong changes in the point of view of the face. This experiment shows the robustness of the algorithm for matching very different instances of the same object.


(a)

(b)

Fig. 2. a) Pieces of relevant images from the eye tracking sequence;. b) Pieces of relevant images from the mouth tracking sequence. The white circle indicates the highest saliency point. The black circle indicates the second highest saliency point.. The shown images are 50x170 pixels


(a) (b)

(c)

Fig. 3. Robustness of the tracking using generic templates; a) Template image; b) The eye tracking using the eye template extracted from the image (a); c) The mouth tracking using the mouth template extracted from the image (a). The white circle indicates the highest saliency point. The black circle indicates the second highest saliency point. The shown images are 50x170 pixels

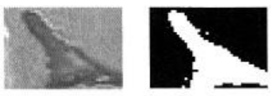
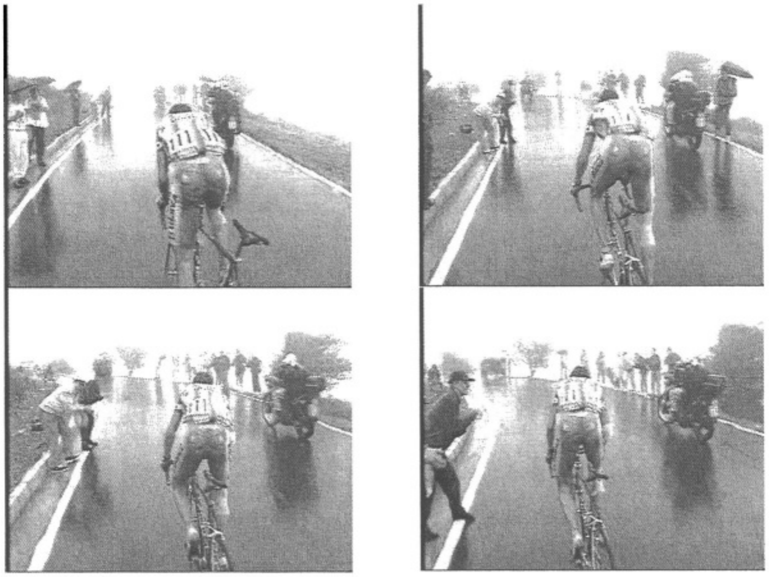

(a) (b)

Fig. 4. a) Template region used in the Figure 5. sequence tracking.; 24x16 pixels; b) Binary mask used to remove background pixels


(a)

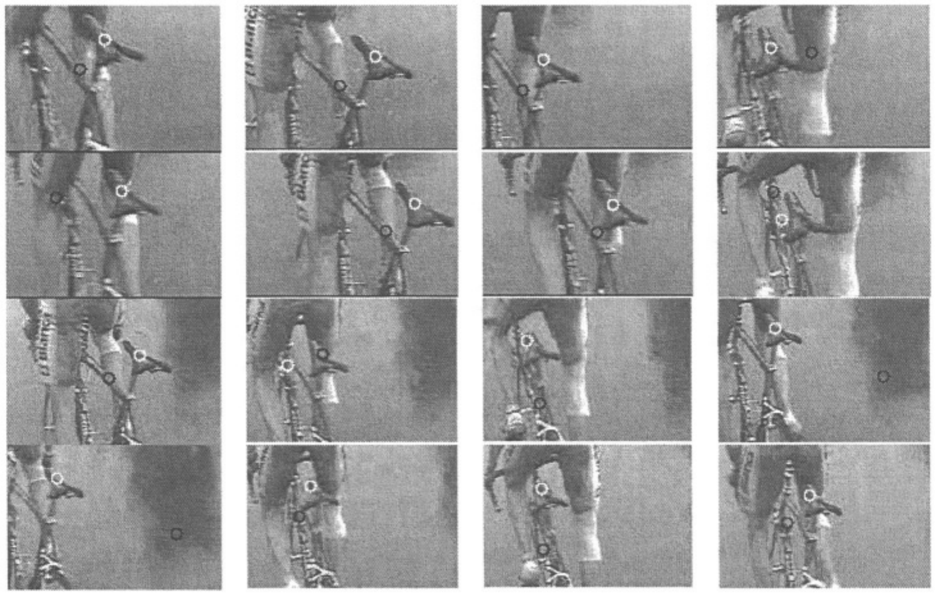

(b)

Fig. 5. a) Four images of the full sequence are shown; b) A subset of relevant details of the tracking sequence is shown. The white circle indicates the highest saliency point. The black circle indicates the second highest saliency point

In figure 5 we show a sequence recorded in a bicycle race. The aim is to track the bicycle saddle throughout the sequence. Figure 4 shows the template region obtained from one image in the sequence. In this sequence, the fast bicycle motion joined to the moving camera produces large changes of the saddle viewpoint. In this case, the level of noise is also very high for several reasons: firstly, the digitalization process reduces the images from PAL format to QCIF format; secondly, large lighting changes can be observed throughout the sequence; and thirdly, because of the effect of the rain and fog present in the scenery (see figure 5 (a)).

In all the experiments, we have tried with different sampling steps (0-4) on the image axis in order to calculate the expression (2). In almost all the images a sampling step of 4 pixels in both axes was sufficient to obtain the highest saliency value in the best location. However, in some cases with clutter background or large changes in geometry or lighting, all the pixels had to be considered.

5 Discussion and Conclusions

The first experiment (Figure 2) shows how our algorithm is stable and robust enough for viewpoint changes. The right eye, defining the region template, is always matched as the best location throughout the entire sequence. We also show how the loss of local spatial information has the advantage of matching different instances of the same object but with different shapes. The left eye is the second best location in all the images. Furthermore, Figure 3 also shows how our template is flexible enough to match very different instances of an object. This means that the template definition is capable of codifying the relevant information about the object removing the local spatial details. In the last experiment (Figure 5), robustness to a non-Gaussian high level of noise and drastic changes in the point of view is shown. It is important to remark that in this difficult sequence of 150 images only in very few images the best location is the second in saliency value.

In all the experiments we have only considered translation motions of the template since our interest is to show that the proposed algorithm is capable of successfully matching a large set of different instance of the original template. Of course the adding of motions as rotation or scale should improved very much the technique. One of the main drawbacks of our algorithm is the loss of the image-plane rotation invariance that is present when the full image histogram is considered. The approaches given in [13][3] do not present these problems since they consider full image histograms. However, in preliminary experiments carried out considering global histograms instead of local histograms, poorer results were obtained. In any case, a comparative study of all these techniques would be an interesting future line of research.

In order to compare these results with the traditional correlation matching algorithms we run the same experiments using this algorithm but we obtained completely unsatisfactory results in terms of the object matching position.

In conclusion, the proposed algorithm represents an efficient generalization of the classical matching by correlation algorithm for the case of deformable objects. This algorithm enables us to match different instances of the same object obtained from a very wide viewpoint range. The loss of the local order imposed by the local histogram uses have revealed a high level of robustness in template matching with strong shape deformations even in high noise conditions. It has also proved to be robust enough for lighting changes by using histograms with a suitable bin number. Although in theory the algorithm is not robust for image-plane rotation and scale, experiments have shown that there is also invariance to small rotations and scale.

Acknowledgement

This work has been financing by the Grant TIC-2001-3316 from the Spanish Ministry of Science and Technology

References

[1] S. Agarwal and D. Roth, Learning a sparse representation for object detection, ECCV'02,113-130,2002

[2] L.Birgé and Y.Rozenholc. How many bins should be put in a regular histogram. Technical Report 721. Laboratoire de Probabilitiés et Modèles Aléatoires, CNRS-UMR 7599, Université Paris VI & Université Paris VII. 2002.

[3] E.Hadjidemetriou, M.D. Grossberg and S.K. Nayar: Spatial information in multiresolution histograms, In Intern. Conf. CVPR'01, 2001.

[4] B.Heisele, P.Ho, J.Wu and T.Poggio, Face recognition: component-based versus global approaches, Computer Vision and Image Understanding 91,6-21,2003

[5] R.Fergus, P. Perona and A. Zisserman: Object class recognition by unsupervised scale-invariant learning. In IEEE CVPR'03,264-271,2003.

[6] L.D.Griffin, Scale-imprecission space, Image and Vision Computing 15, 369-398, 1999.

[7] J.J.Koenderink and A.J. Van Doorn: The Structure of locally orderless images, Intern. Journal of Computer Vision 318273),159-168,1999.

[8] T. Kadir and M. Brady: Scale, saliency and image description. Intern. Journal of Computer Vision, 45 (2):83-105, 2001.

[9] D.G.Lowe, Object recognition from local scale-invariant features. In ICCV'99,1150-1157.

[10] J.Matas, O.Chum, M.Urban and T.Pajdla: Robust wide baseline stereo from maximally stable extremal regions. In BMCV'02 Conference, 384-393,2002

[11] K.Mikolajczyk and C. Schmid: An affine invariant interest point detector. In ECCV'02, 128-142,2002

[12] W. Niblack. The QBIC project: Querying images by content using color, texture and shape. In Proc. Of SPIE Conf. on Storage and Retrieval for image and video database, vol-1908, 173-187, 1993.

[13] B. Schiele and J. L. Crowley. Object recognition using multidimensional receptive field histograms. In ECCV'96, Vol I, pages 610--619, 1996.

[14] B. Schiele and J. L. Crowley: Robustness of object recognition to view point changes using multidimensional receptive fields histograms. ECIS-VAP, 1996.

[15] M.J. Swain and D.H. Ballard. Color Indexing Intern. Journal of Computer Vision, 7(1):11-32.1991.

[16] F.Topsoe. Some inequalities for information divergence and related measures of discrimination. IEEE Trans. Information Theory vol. IT-46, pp. 1602 -1609, July 2000

[17] T.Tuytelaars and L. Van Gool: Wide baseline stereo based on local affinely invariant regions, In British Machine Vision Conference, Bristol, U.K.,412-422. 2000

Detecting Human Heads and Face Orientations Under Dynamic Environment

Akihiro Sugimoto[1], Mitsuhiro Kimura[2], and Takashi Matsuyama[2]

[1] National Institute of Informatics
Tokyo 101-8430, Japan
[2] Graduate School of Informatics, Kyoto University, Japan
sugimoto@nii.ac.jp

Abstract. We propose a two-step method for detecting human heads and estimating face orientations under the dynamic environment. In the first step, the method employs an ellipse as the contour model of human-head appearances to deal with wide variety of appearances. Our method then evaluates the ellipse to detect possible human heads. In the second step, on the other hand, our method focuses on features, such as eyes, the mouth or cheeks, inside the ellipse to model facial components. The method evaluates not only such components themselves but also their geometric configuration to eliminate false positives in the first step and, at the same time, to estimate face orientations. In the both steps, our method employs robust image-features against lighting conditions in evaluating the models. Consequently, our method can correctly and stably detect human heads and estimate face orientations even under the dynamic environment. Our intensive experiments show the effectiveness of the proposed method.

1 Introduction

Automatically detecting and tracking people and their movements is important in many applications such as in– and out-door surveillance, distance learning, or interfaces for human-computer interaction [2, 4, 6, 7, 8, 9]. In particular, the human face is a key object of interest for visual discrimination and identification. A tremendous amount of researches has been made for detecting human heads/faces and for recognizing face orientations/expressions (see [3, 20] for surveys). Most existing methods in the literatures, however, focus on only one of these two. Namely, methods to detect human heads/faces (see [1, 12, 16, 17, 21], for example) do not estimate orientations of the detected heads/faces, and methods to recognize face orientations/expressions (see [10, 13, 14, 15, 18], for example) assume that human faces in an image or an image sequence have been already segmented.

To build a fully automated system that recognizes human faces from images, it is essential to develop robust and efficient algorithms to detect human heads and, at the same time, to identify face orientations. Given a single image or a sequence of images, the goal of automatic human-face recognition is to detect human heads/faces and estimate their orientations regardless of not only their positions, scales, orientations, poses, but also individuals, background changes and lighting conditions.

This paper proposes a two-step method for detecting human heads and, at the same time, for estimating face orientations by a monocular camera under the dynamic environment. In the both steps, we employ models of the human-head contour and face orientations to enhance robustness and stableness in detection. We also introduce model evaluation with only image-features robust against lighting conditions, i.e., the gradient of intensity and texture.

In the first step, our method employs an ellipse as the contour model of human-head appearances to deal with wide variety of appearances. The ellipse is constructed from one ellipsoid based on the camera position with its angle of depression in the environment. Our method then evaluates the ellipse over a given image to detect possible human heads. In evaluation of an ellipse, two other ellipses are generated inside and outside of the ellipse, and the gradient of intensity along the perimeter of the three ellipses is used for accurate detection of human-head appearances.

In the second step, on the other hand, our method focuses on facial components such as eyes, the mouth or cheeks to construct inner models for face-orientation estimation. Based on the camera position with its angle of depression, our method projects the facial components on the ellipsoid onto the ellipse to generate inner models of human-head appearances. Our method then evaluates not only such components themselves but their geometric configuration to eliminate false positives in the first step and, at the same time, to estimate face orientations. Here the Gabor-Wavelets filter, which is verified its robustness and stableness against changes in scale, orientation and illumination, is used in detecting features representing the facial components.

Consequently, our method can correctly and stably detect human heads and estimate face orientations even under the dynamic environment such as illumination changes or face-orientation changes. Our intensive experiments using a face-image database and real-situation images show the effectiveness of the proposed method.

2 Contour Model for Human-Head Appearances

The model-based approach is inevitable to enhance stableness against dynamic changes in environment. This is because features detected from images without any models often generate false positives in recognition.

2.1 Human Head and Its Appearances

Human beings have almost the same contour in shape of the head and an ellipse approximates the appearance of the contour. These observations remain invariant against changes in face orientation. We, therefore, model the contour of human-head appearances by the ellipse [1, 12, 16].

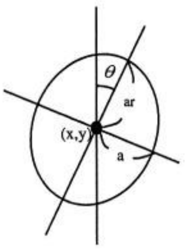


Fig. 1. Geometric model of human head appearances

An ellipse has five parameters in the image (Fig. 1): the 2D coordinates of the ellipse center, the length of the semiminor axis, the oblateness, and the slant of the ellipse.

These parameters of the ellipse change depending on the angle of depression of a camera even though human heads are in the same pose. In particular, the change in oblateness is outstanding. To investigate this change in oblateness, we introduce an ellipsoid to the human-head model in 3D. We assume that the ellipsoid is represented in the world coordinates by

$$x^2 + y^2 + \frac{z^2}{r^2} = 1, \tag{2.1}$$

where $r \geq 1$. We then derive an ellipse as the contour model of human-head appearances depending on the angle of depression of the camera (Fig. 2).

When we set up a camera with any angle of depression, the ellipsoid (2.1) is observed as an ellipse. The length of the semiminor axis of the ellipse is always one. The length of the semimajor axis, on the other hand, is between one and r depending on the angle of depression of the camera.

Now we determine the oblateness, r' $(1 \leq r' \leq r)$, of the ellipse observed by a camera with φ angle of depression providing that the distance of the camera position from the ellipsoid is large enough (Fig. 3). We consider the ellipse obtained through the projection of (2.1) onto the *xz*–plane and its tangential line ℓ. We see that the ellipse, the projection of (2.1) onto the *xz*–plane, is represented by

$$x^2 + \frac{z^2}{r^2} = 1. \tag{2.2}$$

Let its tangential line with slant φ from the $x-$axis be

$$z = \sin\varphi x + b, \tag{2.3}$$

where b is the z-intercept. Combining (2.2) and (2.3), we can compute b. We then have the coordinates of their contact point, from which it follows that

$$r' = \sqrt{\frac{r^4 + \tan^2\varphi}{r^2 + \tan^2\varphi}}.$$

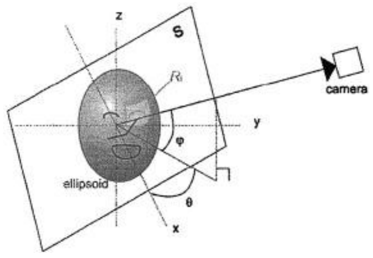


Fig. 2. Human-head model

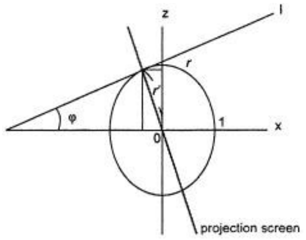


Fig. 3. Projection of the ellipsoid onto the xz–plane

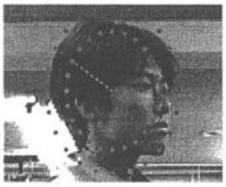


Fig. 4. The (red) ellipse representing a human-head appearance

This relates φ, i.e., the angle of depression of the camera, with r', i.e., the oblateness of the ellipse representing the contour of human-head appearances. We dynamically compute the oblateness of the ellipse from the camera position based on this relationship.

2.2 Evaluation of Contour Model

When we are given an ellipse in the image, how to evaluate the goodness of the ellipse to recognize as a human-head appearance is a tough issue. Employing image features invariant under the dynamic changes in environment is indispensable. We employ, in this paper, the gradient of intensity in evaluating the ellipse to obtain not a human-head appearance itself but an applicant of human-head appearances. This is because the gradient of intensity is robust against illumination changes.

When we fit an ellipse to the contour of a human-head appearance, we have the following observations (Fig. 4):

- Great gradient magnitude of intensity at the ellipse perimeter.
- Continuous changes in intensity along the ellipse perimeter except for the boundary between hair and skin.
- Continuous changes in intensity from just inside the ellipse.

We thus evaluate a given ellipse in three different ways. One is evaluation on the gradient magnitude of intensity at the perimeter of the ellipse. Another is

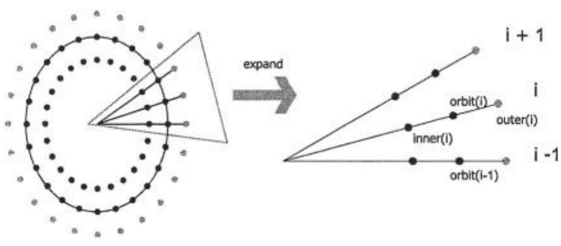


Fig. 5. Evaluation of the ellipse (red: the ellipse to be evaluated)

evaluation on intensity changes along the perimeter of the ellipse and the other is evaluation on intensity changes from the adjacent part inside the ellipse. Introducing these three aspects in evaluation of an ellipse results in more accurately and more robustly obtaining applicants of human-head appearances.

For evaluating an ellipse, we construct two other ellipses (Fig. 5). One is a smaller size ellipse with the identical center and the other is a larger size ellipse with the identical center. In Fig. 5, the red ellipse is to be evaluated and the blue ellipse is the smaller size one and the green is the larger size one. We denote by $orbit(i)$ the intensity of the intersection point of the (red) ellipse to be evaluated and ray i whose end point is the ellipse center. We remark that we have N rays with the same angle-interval and they are sorted by the angle from the horizontal axis in the image. $outer(i)$ and $inner(i)$ are defined in the same way for the cases of the larger size ellipse (green ellipse) and the smaller size ellipse (blue ellipse), respectively.

We now have the following function evaluating the (red) ellipse.

$$f(p) = k\frac{1}{N}\sum_{i=1}^{N}\{G(i) - O(i) - I(i)\}, \tag{2.4}$$

where p is the parameter vector representing the red ellipse and

$$G(i) = |outer(i) - orbit(i)|, \tag{2.5}$$
$$O(i) = |orbit(i) - orbit(i-1)|, \tag{2.6}$$
$$I(i) = |orbit(i) - inner(i)|. \tag{2.7}$$

Note that k is the constant making the value dimensionless [1]. (2.5), (2.6), and (2.7) evaluate the gradient magnitude of intensity at the ellipse perimeter, intensity changes along the ellipse perimeter and intensity changes from just inside the ellipse, respectively. Ellipses having a small value of (2.4) are then regarded as applicants of human-head appearances.

In the next section, we evaluate the applicants of human-head appearances based on features inherent in the human face to recognize as a human-head appearance and, at the same time, to identify the face orientation.

3 Inner Models for Face Orientations

3.1 Facial Components

Eyebrows, eyes, the mouth, the nose and cheeks are the features inherent in the human face. Here we focus on eyes, the mouth and cheeks, and characterize textures around such facial components. We remark that textures are robust against illumination changes.

In oriental countries, we observe around eyes (1) a dark area due to eyebrows, (2) a bright area due to eyelids, and (3) a dark area due to the pupil. These are observations along the vertical direction of the human face and these characterize the texture of an eye area. We also observe that the eye area is symmetrical with respect to the pupil. As for an area around the mouth, on the other hand, we observe (1) a bright area due to the upper lip, (2) a dark area due to the mouth, and (3) a bright area due to the lower lip. In addition, the mouth area is also symmetrical with respect to the vertical center of the face. These observations characterize the texture of a mouth area. We see no complex textures in a cheek area. These observations are almost invariant and stable under changes in illumination, in face-orientation and in scale.

The geometric configuration of the facial components, i.e., the relative position between eyes, the mouth and cheeks is also invariant. Combining the characteristic of textures of the facial components with their geometric configuration enables us to recognize human heads/faces.

For each applicant of human-head appearances, we detect the facial components with their geometric configuration to verify whether it is a human-head appearance. We remark that we can easily identify the scale in detecting facial components since we have already obtained applicants of human-head appearances in terms of ellipses.

3.2 Detecting Facial Components Using Gabor-Wavelets

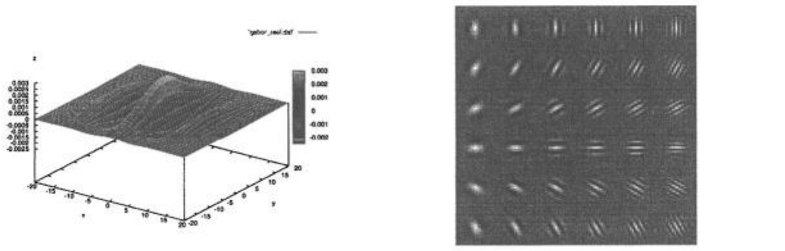
In detecting facial feature points described in the previous section, the Gabor-Wavelets filter is most promising in robustness and stableness against illumination changes [5, 11, 13, 14, 19]. We thus use Gabor-Wavelets to extract the facial feature points, eyes, the mouth and cheeks, as a set of multi-scale and multi-orientation coefficients.

Applying the Gabor-Wavelets filter to a point (x_0, y_0) of a given image $f(x, y)$ can be written as a convolution

$$\psi(x_0, y_0, \sigma, \omega, \phi) = \iint \mathrm{d}x\mathrm{d}y f(x, y) G(x - x_0, y - y_0, \sigma, \omega, \phi)$$

with Gabor kernel $G(x, y, \sigma, \omega, \phi)$s where G is formulated [11] by

$$G(x, y, \sigma, \omega, \phi) = \kappa \mathrm{e}^{\frac{-1}{4\pi\sigma^2}(\tilde{x}^2 + \tilde{y}^2)} \mathrm{e}^{\mathrm{j}\omega\tilde{x}}.$$



(a) Real-part of a kernel (b) Kernels with different orientations

Fig. 6. Gabor kernels

Here

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \quad \kappa = \frac{1}{4\pi^2\sigma^2}, \quad \mathrm{j} = \sqrt{-1}.$$

σ, ω, ϕ are the parameters representing the scale, frequency and orientation, respectively. Note that $(\tilde{x}, \tilde{y})$ is obtained by rotating image point (x, y) by ϕ.

Figure 6 shows an example of a set of Gabor-Wavelets, (a) is the real part of the Gabor kernel with $\sigma = 3.0, \omega = 0.5°, \phi = 0°$, and (b) is the kernels with the same scale, different orientations and frequencies.

We can selectively apply the Gabor-Wavelets filter to particular locations. In addition, we can easily specify scales, frequencies, and orientations in the application of the Gabor-Wavelets filter. In other words, we can apply the Gabor-Wavelets filter to specific regions in the image, i.e., pre-obtained applicants of human-head appearances, with selective parameters in scale, frequency, and orientation to extract a feature vector. This is because we have already detected such applicants of human-head appearances in terms of ellipses (we have already roughly estimated a size of a human-head appearance). This reduces the computational cost in recognizing human-head appearances in the practical sense.

We characterized in Section 3.1 textures around eyes and the mouth along the vertical direction of the human face. To detect these textures we only have to select the parameters in the Gabor-Wavelets filter so that the filter detects the textures along the semimajor axis of the ellipse. Points with maximal values in the response ellipse-region can be eyes and those with minimal values can be a mouth. The area with no singularity, on the other hand, can be cheeks.

3.3 Inner Models of Head Appearances with Facial Components

We construct here inner models of human-head appearances based on the ellipsoid (2.1). As shown in Fig. 2, area R_i on the ellipsoid denoting a facial component such as an eye or a mouth is projected onto the plane when it is viewed from the camera with direction (θ, φ), where θ is the rotation angle toward the camera from the front of the face and φ is the angle of depression of the camera.

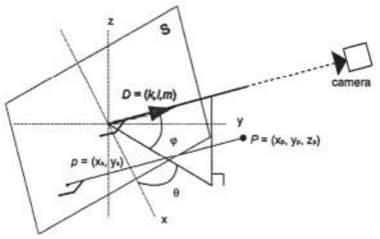


Fig. 7. Geometry in deriving an inner model

The projected area then enables us to identify the location of the facial component in the human-head appearance. Hence, we can construct the inner models of human-head appearances. We remark that we can measure φ in advance when we set up a camera in the environment.

We consider plane S that goes though the origin and whose normal vector is identical with the viewing line of the camera (Fig. 7). Let $\boldsymbol{D} = (k, l, m)^{\top}$ be the unit normal vector of plane S. S is then expressed by $kx + ly + mz = 0$. It is easy to see that k, l and m are expressed in terms of θ and φ:

$$k = \cos\varphi\cos\theta, \quad l = \cos\varphi\sin\theta, \quad m = \sin\varphi.$$

Letting $\boldsymbol{p}$ be the foot of the perpendicular from a 3D point $\boldsymbol{P}$ onto S, we can easily relate $\boldsymbol{P}$ and $\boldsymbol{p}$ by

$$\boldsymbol{p} = \begin{pmatrix} \boldsymbol{X}_S^{\top} \\ \boldsymbol{Y}_S^{\top} \end{pmatrix} \boldsymbol{P}.$$

Here $\boldsymbol{X}_S^{\top}, \boldsymbol{Y}_S^{\top}$ are the orthogonal unit vectors in 3D representing the coordinates in S:

$$\boldsymbol{X}_S = \frac{1}{\sqrt{1-m^2}} \begin{pmatrix} l \\ -k \\ 0 \end{pmatrix}, \quad \boldsymbol{Y}_S = \frac{m}{\sqrt{1-m^2}} \begin{pmatrix} -k \\ -l \\ \frac{1-m^2}{m} \end{pmatrix}.$$

In this way, when depression angle φ and rotation angle θ are specified, we can project a facial area of the ellipsoid onto the image plane to obtain an inner model of the human-head appearance that represents the facial components with their geometric configuration.

Figure 8 shows the inner models of human-head appearances with $\varphi = 0$ (upper: $\theta = 0°, 30°, 60°, 90°$, lower: $\theta = 180°, 270°, 300°, 330°$). R_1 and R_2 denote the eye areas. R_3 denotes the mouth area, and R_4 and R_5 denote the cheek areas.

To the response ellipse-region of the Gabor-Wavelets filter, we apply the inner model matching to detect human-head appearances and face orientations. To be more concrete, if we find eyes, a mouth and cheeks in a response ellipse, we then identify that the ellipse is a human-head appearance and that the orientation of the matched inner-model is the face orientation of the appearance. Otherwise, we identify that the ellipse is not a human-head appearance and eliminate the ellipse.

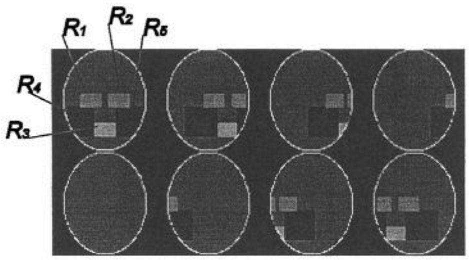


Fig. 8. Inner models of human-head appearances with the facial components

4 Algorithm

Based on the above discussion, we describe here the algorithm for detecting human-head appearances with face orientations. To reduce the computational cost in generating applicants of human-head appearances, we introduce the coarse-to-fine sampling of the parameters representing ellipses. Namely, we first coarsely sample points in the parameter space for the ellipse and then minutely sample the area around the points that are selected based on plausibility of the human-head appearance.

Step 1: Capture an image.
Step 2: Search applicants of human heads in the image.
  2.1: Randomly sample points in the parameter space representing the ellipses that are generated from (2.1); let $\{p_i\}$ be the sampled set.
  2.2: Evaluate each entry of $\{p_i\}$ by (2.4); let $\{p_{i^*}\}$ be the set of samples whose scores of f in (2.4) are less than a given threshold.
  2.3: More minutely sample points in the area around each entry of $\{p_{i^*}\}$ and let $\{p_j^*\}$ be the sampled set. (Note that $\{p_j^*\}$ is applicants of human-head appearances.)
Step 3: To each entry of $\{p_j^*\}$, generate inner models of human-head appearances.
Step 4: Apply the Gabor-Wavelets filter to each entry of $\{p_j^*\}$ to detect facial feature points.
Step 5: To each p_j^*, apply the matching with the corresponding inner models.
Step 6: If p_j^* matches one of its corresponding inner models with a high score, then recognize p_j^* as a human-head appearance and the face orientation as that of the matched inner-model. If p_j^* does not match any of its corresponding inner models with a high score, then eliminate p_j^*.

5 Experimental Evaluation

A. Evaluation on Face Orientations Using a Face-Image Database

We first evaluated our algorithm using a face-image database. The database contains face images of 300 persons with the ages ranging uniformly from 15

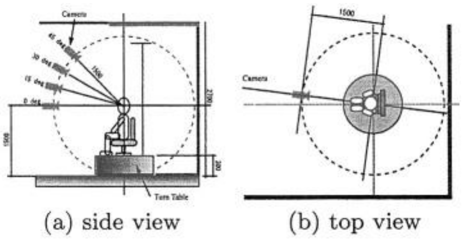


Fig. 9. Parameters in obtaining a face-image database

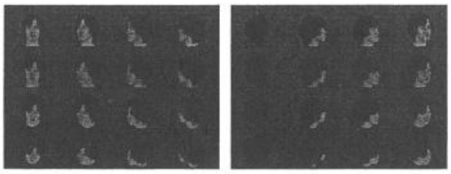


Fig. 10. Sample images of the face-image database (with 0° of depression)

Table 1. Recognition accuracy for different face orientations (%)

angles of depression	face orientations							
	0°	30°	60°	90°	180°	270°	300°	330°
0°	84.7	86.3	83.7	31.0	97.0	34.7	80.0	79.3
15°	64.7	86.3	75.3	27.7	97.7	21.0	71.7	71.7
30°	23.7	75.3	70.3	14.0	99.0	10.0	51.0	51.7
45°	17.7	61.7	51.0	16.0	94.7	8.3	27.0	27.0

years old to 65 years old including men and women. Each person is taken his/her face images from different directions as shown in Fig. 9. To each face image in the database, attached is the ground truth of the direction from which the image is taken.

We used 9600(= 32 × 300) face images in the database where 32 directions are used in taking images of each person: the angles of depression of the camera were $\varphi = 0°, 15°, 30°, 45°$ and the rotation angles with respect to the horizontal axis, i.e., face orientations, were 0°, 30°, 60°, 90°, 180°, 270°, 300°, 330°. Fig. 10 shows samples of the face images of one person in the database.

We applied our algorithm to the 9600 images to detect face orientations. Table 1 shows the recognition rates of the estimated face-orientations.

Table 1 shows that face orientations are in general recognized with high scores. We see low accuracy in orientations with 90° and 270°. This is because one eye and one cheek do not appear in the face with such orientations and thus the inner model matching becomes unstable. We also see that accuracy becomes higher as the angle of depression of the camera becomes smaller. The small angle of depression of the camera means that the face is captured from the horizontal direction of the face and that the facial components clearly appear in the image. It is understood that clearly appearing facial components improves the estimation accuracy of face orientations. A large angle of depression, on the other hand, causes great changes not only in human-head appearance but also in face appearance. Handling such great changes with our models has limitation. This is because we generate a contour model and inner models of human-head

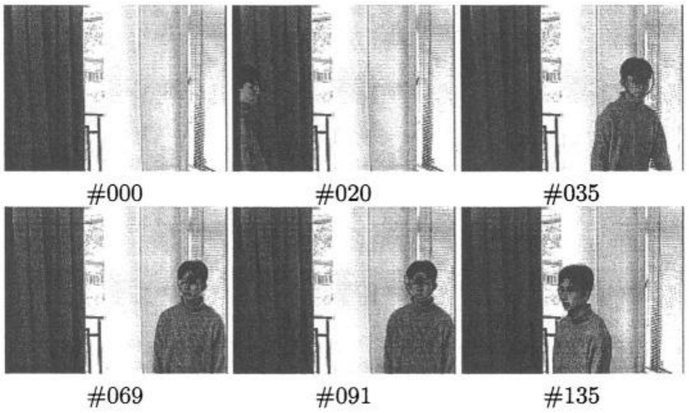


Fig. 11. Examples of detected human heads

appearances from only one ellipsoid. On the other hand, we see that face images from the orientation with 180°, back images of human heads, are recognized stably and accurately independent of the change in angle of depression. This is due to stableness of the Gabor-Wavelets filter in face-feature detection.

B. Evaluation in the Real Situation

Secondly, we evaluated the performance of our method in the real situation and found the robustness and the efficiency of our method.

We set up a camera with 1.8m height and with about 0° angle of depression. We generated the situation in which under changing lighting conditions one person walks around in front of the camera with distance between about 2m and 4m for 20 seconds. 200 images were captured during the time. To the captured images, we applied our method to detect human-head appearances and face orientations.

Figure 11 shows examples of the captured images with frame numbers. The ellipses detected as a human-head appearance are superimposed on the frames. Colors of the ellipses denote face orientations.

We verified that human-head appearances are detected almost correctly and accurately in all the images in spite of changes in illumination. To see the performance of our method, we evaluated the accuracy of the detected human-head appearances. We first fitted an ellipse onto the head appearance in each image by hand to obtain the true ellipse as the reference. We introduced two kinds of evaluation to the ellipse that was recognized as the human-head appearance: one is the accuracy of the center and the other is the accuracy of the semiminor length. We computed the distance (the position error) between the center of the detected ellipse and that of the true ellipse. We also computed the ratio (the size error) of the semiminor length of the detected ellipse to that of the true ellipse.

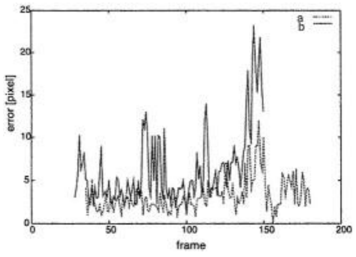


Fig. 12. Position errors in human-head detection (a: our method, b: simple evaluation)

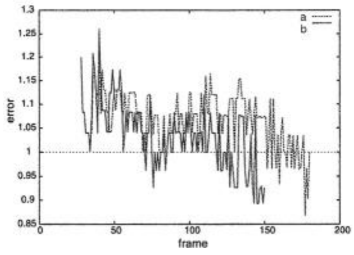


Fig. 13. Size errors in human-head detection (a: our method, b: simple evaluation)

Table 2. Errors in detecting the human head

error		our method	simple evaluation
position	mean [pixels]	3.250	6.401
	standard deviation [pixels]	1.950	4.386
size	mean	0.0762	0.0617
	standard deviation	0.04489	0.04877

These results are shown in Figs.12 and 13. We remark that the same evaluation was applied to the method (called the *simple-evaluation method* (cf. [1, 16])) where the ellipse is evaluated only by (2.5), i.e., the gradient magnitude of intensity at the ellipse perimeter. For the position error and the difference of the size error from 1.0, the average and standard deviation over the image sequence were calculated, which is shown in Table 2.

Figures 12,13 and Table 2 show the effectiveness of our method. Superiority of our method to the simple-evaluation method indicates that introducing the smaller- and larger-size ellipses to ellipse evaluation improves the accuracy in detecting the positions of human-head appearances.

6 Conclusion

We proposed a two-step method for detecting human heads and estimating face orientations by a monocular camera under the dynamic environment. In the both steps, we employ models of the human-head contour and face orientations to enhance robustness and stableness in detection. We also introduced model evaluation with only image-features robust against lighting conditions.

The first step employs an ellipse as the contour model of human-head appearances to deal with wide variety of appearances. The ellipse was constructed from one ellipsoid based on a camera position with its angle of depression in the environment. We then evaluated the ellipse over a given image to detect possible human-head appearances where we generated two other ellipses inside and outside of the ellipse to improve accuracy in detection of human-head appearances.

The second step, on the other hand, focuses on facial components such as eyes, the mouth or cheeks to construct inner models for face-orientation estimation. We evaluated not only such components themselves but also their geometric configuration to eliminate false positives in the first step and, at the same time, to estimate face orientations. Here we used the Gabor-Wavelets filter in detecting features representing the facial components because its robustness and stableness against changes in scale, orientation and illumination are verified.

Consequently, our method can correctly and stably detect human heads and estimate face orientations even under changes in face orientation and in illumination. Our intensive experiments showed the effectiveness of the proposed method. Incorporating wider variety of face orientations into our method is left for future work.

Acknowledgements

The facial data in this paper are used permission of Softopia Japan, Research and Development Division, HOIP Laboratory. It is strictly prohibited to copy, use, or distribute the facial data without permission. This work is in part supported by Grant-in-Aid for Scientific Research of the Ministry of Education, Culture, Sports, Science and Technology of Japan under the contract of 13224051.

References

[1] S. Birchfield: Elliptical Head Tracking Using Intensity Gradients and Color Histograms, *Proc. of CVPR,* pp. 232–237, 1998.

[2] T. J. Cham and J. M. Rehg: A Multiple Hypothesis Approach to Figure Tracking, *Proc. of CVPR,* Vol. 2, pp. 239–245, 1999.

[3] R. Chellappa, C. L. Wilson and S. Sirohey: Human and Machine Recognition of Faces, A Survey, *Proc. of IEEE,* 83 (1995), pp. 705–740.

[4] Y. Cui, S. Samarasekera, Q. Huang and M. Greiffenhagen: Indoor Monitoring via the Collaboration between a Peripheral Sensor and a Foveal Sensor, *Proc. of the IEEE Workshop on Visual Surveillance,* pp. 2–9, 1998.

[5] G. Donato, M. S. Bartlett, J. C. Hager, P. Ekman and T. J. Sejnowski: Classifying Facial Actions, *IEEE Trans. on PAMI,* 21 (1999), 10, pp. 974–989.

[6] L. Davis, S. Fejes, D. Harwood, Y. Yacoob, I. Hariatoglu and M. J. Black: Visual Surveillance of Human Activity, *Proc. of the 3rd ACCV,* Vol.2, pp. 267–274, 1998.

[7] D.M. Gavrila: The Visual Analysis of Human Movement: A Survey, *Computer Vision and Image Understanding,* 73 (1999), 1, pp. 82–98.

[8] I. Haritaoglu, D. Harwood and L. S. Davis: W^4S: A Real-Time System for Detecting and Tracking People in $2\frac{1}{2}D$, *Proc. of the 5th ECCV,* Vol. 1, pp. 877–892, 1998.

[9] I. Haritaoglu, D. Harwood and L. S. Davis: An Appearance-based Body Model for Multiple People Tracking, *Proc. of the 15th ICPR,* Vol. 4, pp. 184–187, 2000.

[10] T.-K. Kim, H. Kim, W. Hwang, S.-C. Kee and J. Kittler: Independent Component Analysis in a Facial Local Residue Space, *Proc. of CVPR,* 2003.

[11] T. S. Lee: Image Representation Using 2D Gabor Wavelets, *IEEE Trans. on PAMI,* 18 (1996), 10, pp. 959–971.

[12] A. Sugimoto, K. Yachi and T. Matsuyama: Tracking Human Heads Based on Interaction between Hypotheses with Certainty, *Proc. of the 13th Scandinavian Conf. on Image Analysis,* (J. Bigun and T. Gustavsson eds: *Image Analysis,* Lecture Notes in Computer Science, Vol. 2749, Springer), pp. 617–624, 2003.

[13] Y. Tian, T. Kanade and J. F. Cohn: *Recognizing Facial Actions by Combining Geometric Features and Regional Appearance Patterns,* CMU-RI-TR-01-0, Robotics Institute, CMU, 2001.

[14] Y. Tian, T. Kanade and J. F. Cohn: Evaluation of Gabor-Wavelets Based Facial Action Unit Recognition in Image Sequences of Increasing Complexity, *Proc. of the 5th FG,* pp. 229–234, 2002.

[15] Y. Wu and K. Toyama: Wide-Range, Person- and Illumination-Insensitive Head Orientation Estimation, *Proc. of the 4th FG,* pp. 183–188, 2000.

[16] K. Yachi, T. Wada and T. Matsuyama: Human Head Tracking using Adaptive Appearance Models with a Fixed-Viewpoint Pan-Tilt-Zoom Camera *Proc. of the 4th FG,* pp. 150–155, 2000.

[17] Z. Zeng and S. Ma: Head Tracking by Active Particle Filtering, *Proc. of the 5th FG,* pp. 89–94, 2002.

[18] L. Zhang and D. Samaras: Face Recognition under Variable Lighting using Harmonic Image Exemplars, *Proc. of CVPR,* 2003.

[19] Z. Zhang, M. Lyons, M. Schuster and S. Akamatsu: Comparison between Geometry-based and Gabor-Wavelets-based Facial Expression Recognition using Multi-layer Perception, Proc. of the 3rd FG, pp. 454–459, 1998.

[20] W. Y. Zhao, R. Chellappa, A. Rosenfeld and P. J. Phillips: *Face Recognition: A Literature Survey,* CAR-TR-984, UMD, 2000.

[21] S. Zhou, V. Krueger and R. Chellappa: Face Recognition from Video: A Condensation Approach, *Proc. of the 5th FG,* pp. 221–226, 2002.

Analysis of Human Walking Based on *aSpaces*

J. Gonzàlez[1], J. Varona[2], F.X. Roca[1], and J.J. Villanueva[1]

[1] Computer Vision Center & Dept. d'Informàtica,
Edifici O, Universitat Autònoma de Barcelona (UAB),
08193 Bellaterra, Spain
`{poal,xavir,villanueva}@cvc.uab.es`
`http://www.cvc.uab.es`

[2] Dept. Matemàtiques i Informàtica & Unitat de Gràfics i Visió,
Universitat de les Illes Balears (UIB),
07071 Palma de Mallorca, Spain
`vdmijvg4@uib.es`
`http://dmi.uib.es`

Abstract. In this paper, we address the analysis of human actions by comparing different performances of the same action, i.e. walking. To achieve this goal, we define a proper human body model which maximizes the differences between human postures and, moreover, reflects the anatomical structure of the human beings. Subsequently, a human action space, called *aSpace,* is built in order to represent a performance, i.e., a predefined sequence of postures, as a parametric manifold. The final human action representation is called *p–action*, which is based on the most characteristic human body postures found during several walking performances. These postures are found automatically by means of a predefined distance function, and they are called *key-frames*. By using key-frames, we *synchronize* any performance with respect to the action model. Furthermore, by considering an arc length parameterization, independence from the speed at which performances are played is attained. Consequently, the *style* of human walking can be successfully analysed by establishing differences between a male and a female walkers.

1 Introduction

Computational models of action style are relevant to several important application areas [4]. On the one hand, it helps to enhance the qualitative description provided by a human action recognition module. Thus, for example, it is important to generate style descriptions which best characterize an specific agent for identification purposes. Also, the style of a performance can help to establish ergonomic evaluation and athletic training procedures. Another application domain is to enhance the human action library by training different action models for different action styles, using the data acquired from a motion capture system. Thus, it should be possible to re-synthesize human performances exhibiting different postures.

In the literature, the most studied human action is *walking*. Human walking is a complex, structured, and constrained action, which involves to maintain

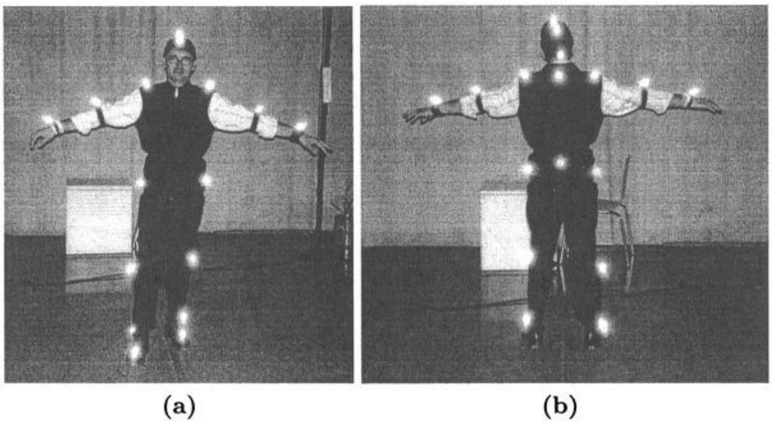


Fig. 1. Procedure for data acquisition. Figs. **(a)** and **(b)** shows the agent with the 19 markers on the joints and other characteristic points of its body

the balance of the human body while transporting the figure from one place to another. The most exploited characteristic is the cyclic nature of walking, because it provides uniformity to the observed performance. In this section, we propose to use our human action model in the study of the *style* inherent in human walking performances, such as the gender, the walking pace, or the effects of carrying load, for example.

In this paper, we show how to use the *aSpace* representation developed in [5, 6] to establish a characterization of the walking style in terms of the gender of the walker. The resulting characterization will consist of a description of the variation of specific limb angles during several performances played by agents of different gender. The aim is to compare human performances to derive conceptual differences between male and female walkers.

2 Defining the Training Samples

Motion capture is the process of recording live movement and translating it into usable mathematical terms by tracking a number of key points or regions/segments in space over time and combining them to obtain a 3-D representation of the performance [9].

In our experiments, an optical system was used to provide real training data to our algorithms. The system is based on six synchronized video cameras to record images, which incorporates all the elements and equipment necessary for the automatic control of cameras and lights during the capture process. It also includes an advanced software pack for the reconstruction of movements and the effective treatment of occlusions.

Consequently, the subject first placed a set of 19 reflective markers on the joints and other characteristic points of the body, see Fig. 1.**(a)** and **(b)**. These

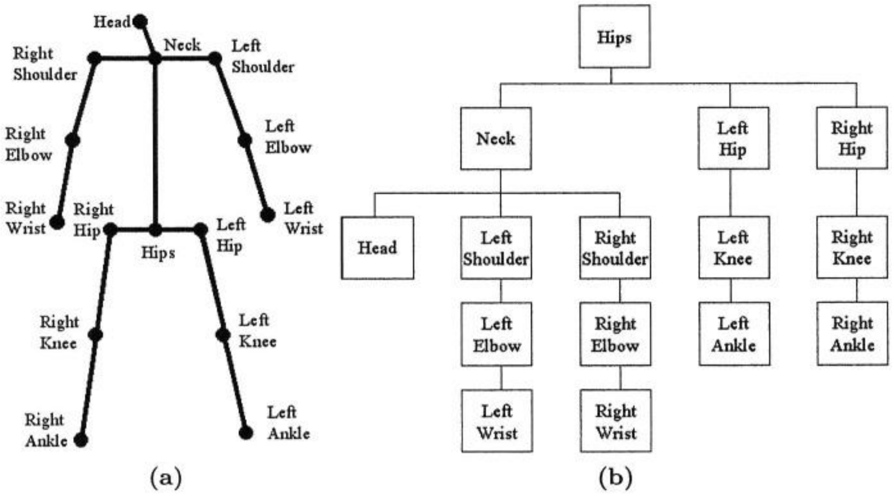


Fig. 2. (**a**) Generic human body model represented using a stick figure similar to [3], here composed of twelve limbs and fifteen joints. (**b**) Hierarchy of the joints of the human body model. Once the 3-D position of these joints are obtained from a motion capture system, the human body representation is built conforming to this hierarchy

markers are small round pieces of plastic covered in reflective material. Subsequently, the agent is placed in a controlled environment (i.e., controlled illumination and reflective noise), where the capture will be carried out. As a result, the accurate 3-D positions of the markers are obtained for each recorded frame, 30 frames per second.

An action will be represented as a sequence of postures, so a proper body model is required. In our experiments, not all the 19 markers are considered to model human actions. In fact, we only process those markers which correspond to the joints of a predefined human body model. The body model considered is composed of twelve rigid body parts (hip, torso, shoulder, neck, two thighs, two legs, two arms and two forearms) and fifteen joints, see Fig. 2.(**a**). These joints are structured in a hierarchical manner, where the root is located at the hip, see Fig. 2.(**b**).

We represent the human body by describing the elevation and orientation of each limb using three different angles which are more natural to be used for limb movement description [1]. We consider the 3-D polar space coordinate system which describes the orientation of a limb in terms of its latitude and longitude. As a result, the twelve independently moving limbs in the 3-D polar space have a total of twenty-four rotational DOFs which correspond to thirty-six absolute angles (w.r.t. the world coordinate system).

So we compute the 3-D polar angles of a limb (i.e., elevation ϕ_l, latitude θ_l, and longitude ψ_l) as:

$$\phi_l = \tan^{-1}\left(\frac{y_i - y_j}{\sqrt{(x_i - x_j)^2 + (z_i - z_j)^2}}\right),$$
$$\theta_l = \tan^{-1}\left(\frac{x_i - x_j}{\sqrt{(y_i - y_j)^2 + (z_i - z_j)^2}}\right),$$
$$\psi_l = \tan^{-1}\left(\frac{z_i - z_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}\right), \tag{1}$$

where denominators are also prevented to be equal to zero. Using this description, angle values lie between the range of $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, and the angle discontinuity problem is avoided.

Note that human actions are constrained movement patterns which involve to move the limbs of the body in a particular manner. That means, there is a relationship between the movement of different limbs while performing an action. In order to incorporate this relationship into the human action representation, we consider the hierarchy of Fig. 2.(**b**) in order to describe each limb with respect to its parent. That means, the relative angles between two adjacent limbs are next computed using the absolute angles of Eq. (1). Consequently, by describing the the human body using the relative angles of the limbs, we actually model the body as a hierarchical and articulated figure.

As a result, the model of the human body consists of thirty-six relative angles:

$$\mathbf{\Delta}_s = (\phi'_1, \theta'_1, \psi'_1, \phi'_2, \theta'_2, \psi'_2, ..., \phi'_{12}, \theta'_{12}, \psi'_{12})^T. \tag{2}$$

Using this definition, we measure the *relative motion* of the human body. In order to measure the *global motion* of the agent within the scene, the variation of the height of the hip $\mathbf{u}_s$ over time is included in the model definition:

$$\mathbf{x}_s = (\mathbf{u}_s, \mathbf{\Delta}_s)^T. \tag{3}$$

Therefore, our training data set is composed of r sequences $\mathbf{A} = \{\mathbf{H}_1, \mathbf{H}_2, ..., \mathbf{H}_r\}$, each one corresponding to a cycle or *stride* of the *aWalk* action $\mathbf{A}$. Three males and three females were recorded, each one walking five times in circles. Each sequence $\mathbf{H}_j$ corresponds to f_j human body configurations:

$$\mathbf{H}_j = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{f_j}\}, \tag{4}$$

where each $\mathbf{x}_i$ of dimensionality $n \times 1$ stands for the 37 values of the human body model described previously. Consequently, our human performance analysis is restricted to be applied to the variation of these twelve limbs.

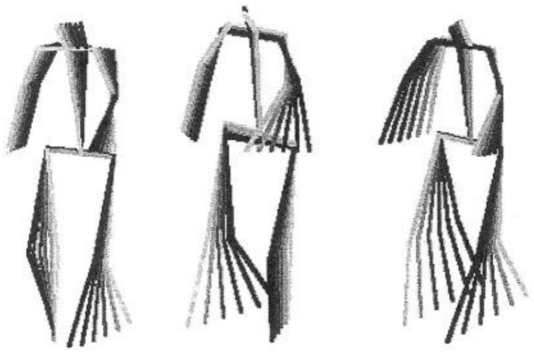


Fig. 3. The three most important modes of variation of the *aWalk aSpace*

3 The *aWalk aSpace*

Once the learning samples are available, we compute the *aSpace* representation of the *aWalk* action, as detailed in [5]. So, we define the complete set of acquired human postures as:

$$\mathbf{A} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_f\}, \tag{5}$$

where f refers to the overall number of training postures for this action:

$$f = \sum_{j=1}^{r} f_j. \tag{6}$$

The mean human posture $\bar{\mathbf{x}}$ and the covariance matrix Σ of $\mathbf{A}$ are calculated. Subsequently, the eigenvalues λ_i and eigenvectors $\mathbf{e}_i$ of Σ are found by solving the eigenvector decomposition equation.

We preserve major linear correlations by considering the eigenvectors corresponding to the largest eigenvalues. Fig. 3 shows the three eigenvectors associated to the three largest eigenvalues, which correspond to the most relevant modes of change of the human posture in the *aWalk aSpace*. As expected, these modes of variation are mainly related to the movement of legs and arms.

So, by selecting the first m eigenvectors, $\{\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_m\}$, we determine the most important modes of variation of human body during the *aWalk* action. The value for m is commonly determined by eigenvalue thresholding. Consider the overall variance of the training samples, computed as the sum of the eigenvalues:

$$\lambda_T = \sum_{k=1}^{n} \lambda_k. \tag{7}$$

If we need to guarantee that the first m eigenvectors actually model, for example, 95% of the overall variance of the samples, we choose m so that:

$$\frac{\sum_{k=1}^{m} \lambda_k}{\lambda_T} \geq 0.95. \tag{8}$$

The individual contribution of each eigenvector determines that 95% of the variation of the training data is captured by the thirteen eigenvectors associated to the thirteen largest eigenvalues. So the resulting *aWalk aSpace* is defined as:

$$\mathbf{\Omega} = (\mathbf{E}, \mathbf{\Lambda}, \bar{\mathbf{x}}). \tag{9}$$

4 Parametric Action Representation: The *p–action*

Using the *aWalk aSpace,* each performance is represented as a set of points, each point corresponding to the projection of a learning human posture $\mathbf{x}_i$:

$$\mathbf{y}_i = [\mathbf{e}_1, ..., \mathbf{e}_m]^T (\mathbf{x}_i - \bar{\mathbf{x}}). \tag{10}$$

Thus, we obtain a set of discrete points $\mathbf{y}_i$ in the action space that represents the action class $\mathbf{\Omega}$. By projecting the set of human postures of an *aWalk* performance $\mathbf{H}_j$, we obtain a cloud of points wich corresponds to the projections of the postures exhibited during such a performance.

We consider the projections of each performance as the control values for an interpolating curve $\mathbf{g}_j(p)$, which is computed using a standard cubic-spline interpolation algorithm [10]. The parameter p refers to the temporal variation of the posture, which is normalized for each performance, that is, $p \in [0, 1]$. Thus, by varying p, we actually move along the manifold.

This process is repeated for each performance of the learning set, thus obtaining r manifolds:

$$\mathbf{g}_j(p), \qquad p \in [0,1], j = 1, ..., r. \tag{11}$$

Afterwards, the mean manifold $\mathbf{g}(p)$ is obtained by interpolating between these means for each index p. This performance representation is not influenced by its duration, expressed in seconds or number of frames. Unfortunately, this resulting parametric manifold is influenced by the fact that any subject performs an action in the way he or she is used to. That is to say, the extreme variability of human posture configurations recorded during different performances of the *aWalk* action affects the mean calculation for each index p. As a result, the manifold may comprise abrupt changes of direction.

A similar problem can be found in the computer animation domain, where the goal is to generate virtual figures exhibiting smooth and realistic movement. Commonly, animators define and draw a set of specific frames, called *key frames* or *extremes,* which assist the task of drawing the intermediate frames of the animated sequence.

Likewise, our goal consists in the extraction of the most characteristic body posture configurations which will correspond to the set of key-frames for that action. From a probabilistic point of view, we define characteristic postures as

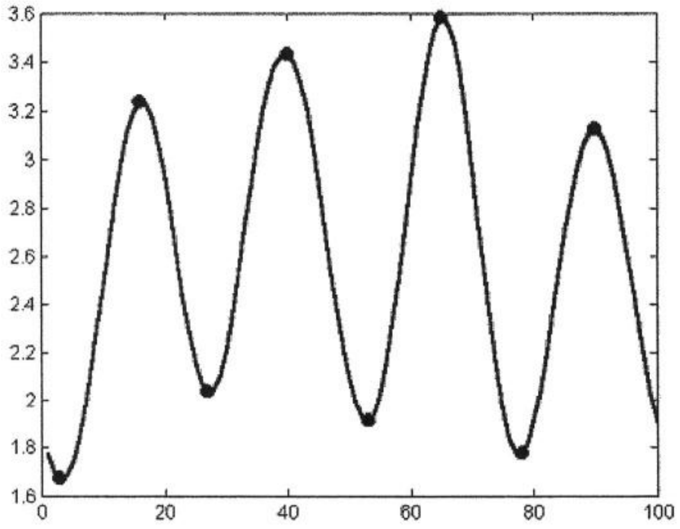


Fig. 4. Distance measure after pose ordering applied to the points of the mean manifold in the *aWalk aSpace*. Maxima (i.e., the key-frames) also correspond to important changes of direction of the manifold

the least likely body postures exhibited during the action performances. As the *aSpace* is built based on PCA, such a space can also be used to compute the action class conditional density $P(\mathbf{x}_j|\Omega^A)$.

We assume that the *Mahalanobis distance* is a sufficient statistic for characterizing the likelihood:

$$d(\mathbf{x}_j) = (\mathbf{x}_j - \bar{\mathbf{x}})^T \Sigma (\mathbf{x}_j - \bar{\mathbf{x}}). \tag{12}$$

So, once the mean manifold $\mathbf{g}(p)$ is established, we compute the likelihood values for the sequence of pose-ordered projections that lie in such a manifold [2, 8]. That is, we apply Eq. (12) for each component of the manifold $\mathbf{g}(p)$. Local maxima of this function correspond to locally maximal distances or, in other words, to the least likely samples, see Fig. 4.

Since each maximum of the distance function corresponds to a key-frame $\mathbf{k}_i$, the number of key-frames k is determined by the number of maxima. Thus, we obtain the set of time-ordered key-frames for the *a Walk* action:

$$\mathbf{K} = \{\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_k\}, \qquad \mathbf{k}_i \in \mathbf{g}(p). \tag{13}$$

Once the key-frame set $\mathbf{K}$ is found, the final human action model is represented as a parametric manifold $\mathbf{f}(p)$, called *p–action*, which is built by interpolation between the peaks of the distance function defined in Eq. (12). We refer the reader to [5] for additional details.

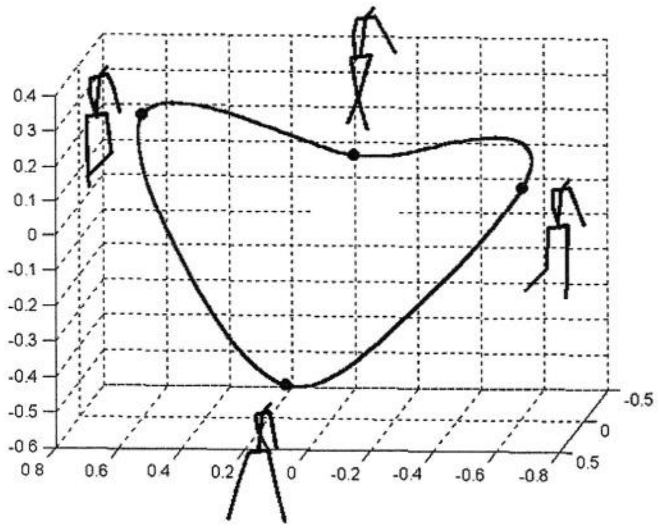


Fig. 5. Prototypical performance manifold, or *p–action*, in the *aWalk aSpace.* Depicted human postures correspond to the key-frame set

Fig. 5 shows the *aWalk p–action*, and the depicted postures correspond to the key-frame set. To conclude, the *aWalk* action model is defined as:

$$\mathbf{\Gamma} = (\mathbf{\Omega}, \mathbf{K}, \mathbf{f}). \tag{14}$$

5 Human Performance Comparison

In order to compare performances played by male and female agents, we define two different training sets:

$$\begin{aligned} \mathbf{H}^{W_M} &= \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{f_M}\}, \\ \mathbf{H}^{W_F} &= \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{f_F}\}, \end{aligned} \tag{15}$$

that is, the set human postures exhibited during several *aWalk* performances for a male and a female agent, respectively. In our experiments, near 50 *aWalk* cycles per walker have been recorded. As a result, the training data is composed of near 1500 human posture configurations per agent.

Next, we project the human postures of $\mathbf{H}^{W_M}$ and $\mathbf{H}^{W_F}$ in the *aWalk aSpace,* as shown in Fig. 6. The cyclic nature of the *a Walk* action explains the resulting circular clouds of projections. Also, note that both performances do not *intersect,* that is, they do not exhibit the same set of human postures. This is due to the

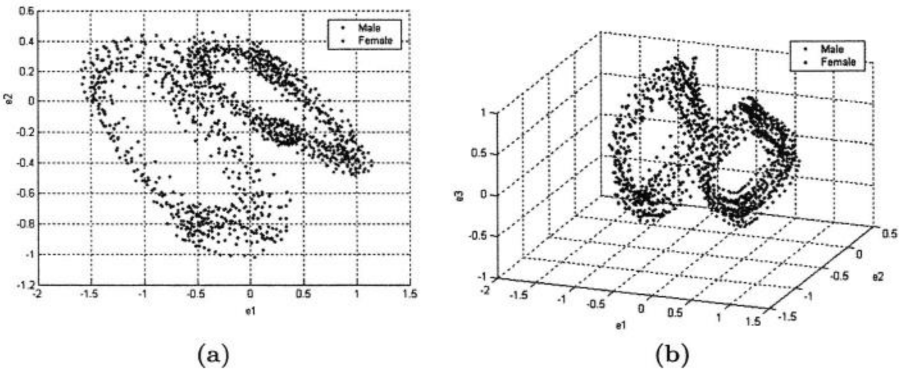


Fig. 6. Male and female postures projected in the *aWalk aSpace,* by considering two **(a)** and three **(b)** eigenvectors for the *aSpace* representation

high variability inherent in human performances. Consequently, we can *identify* a posture as belonging to a male or female walker.

However, the scope of this paper is not centered on determining a *discriminative* procedure between generic male and female walkers. Instead, we look for a comparison procedure to subsequently *evaluate* the variation of the angles of specific agents while performing the same action, in order to derive a characterization of the action *style.*

Following the procedure described in the last section, we use the projections of each walker to compute the performance representation for the male $\mathbf{\Gamma}^{W_M}$ and female $\mathbf{\Gamma}^{W_F}$ agents:

$$\begin{aligned} \mathbf{\Gamma}^{W_M} &= (\mathbf{\Omega}, \mathbf{K}^{W_M}, \mathbf{f}^{W_M}), \\ \mathbf{\Gamma}^{W_F} &= (\mathbf{\Omega}, \mathbf{K}^{W_F}, \mathbf{f}^{W_F}), \end{aligned} \tag{16}$$

where $\mathbf{f}^{W_M}$ and $\mathbf{f}^{W_F}$ refer to the male and female *p–actions*, respectively. These manifolds have been obtained by interpolation between the key-frames of their respective key-frame set, i.e., $\mathbf{K}^{W_M}$ and $\mathbf{K}^{W_F}$. Fig. 7 shows the resulting *p–action* representations in the *aWalk aSpace.*

6 Arc Length Parameterization of *p–actions*

In order to compare the human posture variation for both performances, we sample both *p–actions* to describe each manifold as a sequence of projections:

$$\begin{aligned} \mathbf{f}^{W_M}(\mathbf{p}) &= [\, \mathbf{y}_1^{W_M}, \mathbf{y}_2^{W_M}, ..., \mathbf{y}_{q_M}^{W_M} \,], \\ \mathbf{f}^{W_F}(\mathbf{p}) &= [\, \mathbf{y}_1^{W_F}, \mathbf{y}_2^{W_F}, ..., \mathbf{y}_{q_F}^{W_F} \,], \end{aligned} \tag{17}$$

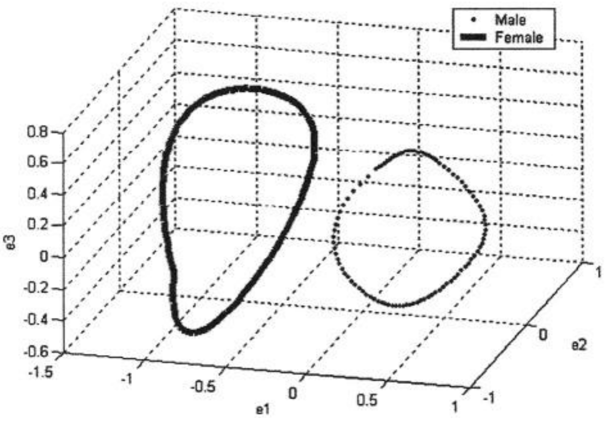


Fig. 7. Male and female performance representations in the *aWalk aSpace*

where q_M and q_F refer to the number of projections considered for performance comparison. However, the sampling rate of both *p–actions* should be established in order to attain independence from the speed at which both performances have been played. Thus, synchronization of recorded performances is compulsory to allow comparison.

Speed control is achieved by considering the distance along a curve of interpolation or, in other words, by establishing a reparameterization of the curve by arc length [7]. Thus, once the *aWalk p–action* is parameterized by arc length, it is possible to control the speed at which the manifold is traversed.

Subsequently, the key-frames will be exploited for synchronization: the idea of synchronization arises from the assumption that any performance of a given action should present the key-frames of such an action. Therefore, the key-frame set is considered as the reference postures in order to adjust or *synchronize* any new performance to our action model. Subsequently, by considering the arc length parameterization, the aim is to sample the new performance and the *p–action* so that the key-frames are equally spaced in both manifolds.

Therefore, both *p–actions* are parameterized by arc length and, subsequently, the synchronization procedure described in [6] is applied: once the key-frames establish the correspondences for $\mathbf{f}^{W_M}$ and $\mathbf{f}^{W_F}$, we modify the rate at which the male and female *p–actions* are sampled, so that their key-frames coincide in time with the key-frames of the *aWalk p–action*.

As a result of synchronization, differences between a performance and the prototypical action can be derived by analyzing the resulting angle variation curves. Such differences can be associated with natural language terms related to speed, naturalness, or suddenness, for example. These terms can be used to enhance the description of a recognized action.

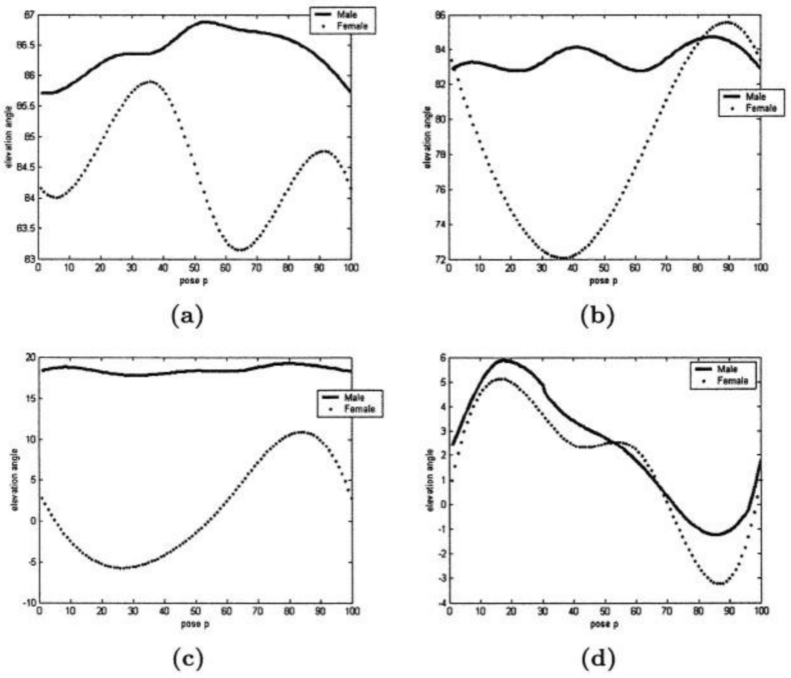


Fig. 8. The elevation variation for the shoulder **(a)**, torso **(b)**, left arm **(c)**, and right thigh **(d)** limbs are depicted for a male and a female walker

7 Experimental Results

Once the male and female *p–actions* are synchronized, the angle variation for different limbs of the human body model can be analysed. Fig. 8.**(a)**, **(b)**, **(c)**, and **(d)** show the evolution of the elevation angle for four limbs of the human body model, namely the shoulder, torso, left arm, and right thigh, respectively.

By comparing the depicted angle variation values of both walkers, several differences can be observed. The female walker moves her shoulder in a higher degree than the male shoulder. That is, the swing movement of the shoulder is more accentuated for the female. Also, the female bends the torso in a higher inclination degree. Therefore, the swing movement of the shoulder and torso for the male agent is less pronounced. The female walker also exhibits an emphasized swing movement in her left arm. On the contrary, the male agent does not show a relevant swing movement for his left arm. As expected, when the left arm swings backward, the right thigh swings forward, and vice versa. When comparing the angle variation of the right thigh for both walkers, few dissimilarities can be derived. In fact, most differences between the male and the female performances have been found in the elevation values of the limbs corresponding to the upper part of the human body.

8 Conclusions

Summarizing, a comparison framework has been presented which allows to evaluate the variation of the angles of specific human body limbs for different agents while performing the same action. This analysis of human actions helps to determine those human body model parameters which best characterize an specific action style. Consequently, a suitable characterization of the action *style* can be built by analyzing the resulting angle values.

Using the characterization about action styles, human action recognition procedures can be enhanced by deriving style attributes about recognized performances. Also, we can enhance human action synthesis procedures by incorporating restrictions about a predefined action style, which the virtual agent should obey while reproducing the requested action.

Acknowledgements

This work has been supported by project TIC2003-08865 of spanish CICYT.

References

[1] D. Ballard and C. Brown. *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[2] H. Borotschnig, L. Paletta, M. Prantl, and A. Pinz. Appearance-based active object recognition. *Image and Vision Computing,* 18:715–727, 2000.

[3] J. Cheng and M. F. Moura. Capture and represention of human walking in live video sequences. *IEEE Transactions on Multimedia,* 1(2):144–156, 1999.

[4] J. Davis and A. F. Bobick. The representation and recognition of movement using temporal templates. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'97),* pages 928–934, San Juan, Puerto Rico, 1997.

[5] J. Gonzàlez, J. Varona, F. X. Roca, and J. J. Villanueva. *aSpaces:* Action spaces for recognition and synthesis of human actions. In *Proc. Second International Workshop on Articulated Motion and Deformable Objects (AMDO 2002),* pages 189–200, Palma de Mallorca, Spain, 2002.

[6] J. Gonzàlez, J. Varona, F. X. Roca, and J. J. Villanueva. A human action comparison framework for motion understanding. In *Artificial Intelligence Research and Developments. Frontiers in Artificial Intelligence and Applications,* volume 100, pages 168–177. IOS Press, 2003.

[7] B. Guenter and R. Parent. Computing the arc length of parametric curves. *IEEE Computer Graphics and Applications,* 10(3):72–78, May 1990.

[8] H. Murase and S. K. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision,* 14:5–24, 1995.

[9] F. J. Perales, A. Igelmo, J. M. Buades, P. Negre, and G.Bernat. Human motion analysis & synthesis using computer vision and graphics techniques. Some applications. In *IX Spanish Symposium on Pattern Recognition and Image Analysis,* volume 1, pages 271–277, Benicassim, Spain, 16-18 May 2001.

[10] W. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C.* Cambridge University Press, Cambridge, 1988.

Complex Articulated Object Tracking

Andrew I. Comport, Éric Marchand, and François Chaumette

IRISA - INRIA Rennes
Campus de Beaulieu, 35042 Rennes, France
Firstname.Lastname@irisa.fr

Abstract. In this paper new results are presented for tracking complex multi-body objects. The theoretical framework is based on robotics techniques and uses an a-priori model of the object including a general mechanical link description. A new kinematic-set formulation takes into account that articulated degrees of freedom are directly observable from the camera and therefore their estimation does not need to pass via a kinematic-chain back to the root. By doing this the tracking techniques are efficient and precise leading to real-time performance and accurate measurements. The system is locally based upon an accurate modeling of a distance criteria. A general method is given for defining any type of mechanical link and experimental results show prismatic, rotational and helical type links. A statistical M-estimation technique is applied to improve robustness. A monocular camera system was used as a real-time sensor to verify the theory.

1 Introduction

Previously, non-rigid motion has been classed into three categories describing different levels of constraints on the movement of a body: articulated, elastic and fluid [1]. In this paper the first class of non-rigid motion is considered and a link is made with the remaining classes. An “articulated” object is defined as a multi-body system composed of at least two rigid **components** and at most six independent degrees of freedom between any two components. With articulated motion, a non-rigid but constrained dependence exists between the components of an **object**. Previous methods have attempted to describe articulated motion either with or without an a-priori model of the object. In this study a 3D model is used due to greater robustness and efficient computation. Knowing the object in advance helps to predict hidden movement, which is particularly interesting in the case of non-rigid motion because there is an increased amount of self-occlusion. Knowing the model also allows an analytic relation for the system dynamics to be more precisely derived.

State of the Art In general, the methods which have been proposed in the past for articulated object tracking rely on a good rigid tracking method. In computer vision the geometric primitives considered for tracking have been numerous, however, amongst them distance based features have shown to be efficient and robust [2, 3, 4, 5]. Another important issue is the 2D-3D registration problem. *Purely geometric* (eg, [6]), or *numerical and iterative* [7] approaches may be considered. *Linear approaches* use a least-squares method to estimate the pose and are considered to be more suitable for initialization procedures. *Full-scale non-linear optimization techniques* (e.g., [2, 8, 3, 5]) consists of minimizing the error between the observation and the forward-projection of the model. In this case, minimization is handled using numerical iterative algorithms such as Newton-Raphson or Levenberg-Marquardt. The main advantage of these approaches are their accuracy. The main drawback is that they may be subject to local minima and, worse, divergence. This approach is better suited to maintaining an already initialized estimation.

Within this context it is possible to envisage different ways to model the pose of an articulated object. The first method for tracking articulated objects using kinematic chains (see Figure 1) appears in well known work by Lowe [9]. He demonstrates a classical method using partial derivatives. In his paper the kinematic chain of articulations is represented as tree structure of internal rotation and translation parameters and the model points are stored in the leaves of this tree.The position and partial derivatives of each point in camera-centered coordinates is determined by the transformations along the path back to the root.

Recently, more complex features have been used for non-rigid object tracking in [10]. They make use of deformable super-quadric models combined with a kinematic chain approach. However, real-time performance is traded-off for more complex models. Furthermore, this method requires multiple viewpoints in order to minimize the system of equations. As Lowe points out, the tendencies in computer graphics have been toward local approximations via polyhedral models. Ruff and Horaud [11] give another kinematic-chain style method for the estimation of articulated motion with an un-calibrated stereo rig. They introduce the notion of projective kinematics which allows rigid and articulated motions to be represented within the transformation group of projective space. The authors link the inherent projective motions to the Lie-group structure of the displacement group. The minimization is determined in projective space and is therefore invariant to camera calibration parameters.

A second approach has been proposed by Drummond and Cippola [3] which treats articulated objects as groups of rigid components with constraints between them directly in camera coordinates (see Figure 2). It appears that the full pose of each rigid component is initially computed independently requiring the estimation of a redundant number of parameters. Lagrange multipliers are then used to constrain these parameters according to simple link definitions. This method uses Lie Algebra to project the measurement vector (distances) onto the subspace defined by the Euclidean transformation group (kinematic screw). They also implement M-estimation to improve robustness.

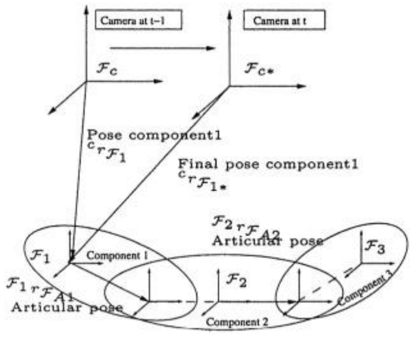


Fig. 1. Kinematic chain method: The pose of an articulated object is determined via a kinematic chain of rigid bodies extending to sub components

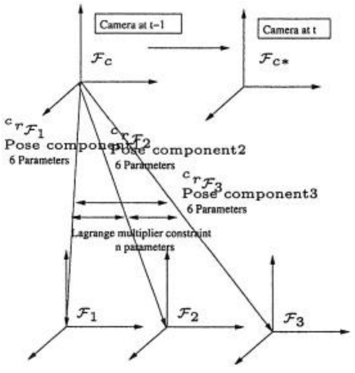


Fig. 2. Lagrange Multiplier method: The pose between the camera and each part of the object is calculated directly. Constraints between the components are enforced via Lagrange multipliers

Contribution A new model is proposed in this paper which is based on the observation that within a vision system one has *direct access* with a camera to the parameters of an articulated object. Thus, unlike traditional techniques using robotics based approaches, there is no need to sum partial derivatives along a kinematic chain back to the root. As will be shown, the joint reference frame plays an important role in modeling articulated objects. The method presented in this paper also integrates a mechanical link formulation for simple definition of articulations.

It is important to correctly model the behavior of the system to obtain maximum decoupling of joint parameters and therefore interesting minimization properties. In this paper a kinematic set approach is proposed. With articulated motion, unlike the case of rigid motion, the subsets of movement which may be attributed to either the object or camera are not unique. A novel subset approach is used whereby the minimization is carried out on decoupled subsets of parameters by defining subspace projectors from the joint definitions. This allows the error seen in the image to be partially decoupled from the velocities of the object by determining the independent sets of velocities present in object space. The principal advantages of this approach are that it:

- is more efficient in terms of computation than previous methods.
- eliminates the propagation of errors between free parameters.
- models more closely the real behavior of the system than a camera frame based approach.

In the remainder of this paper, Section 2 presents the principle of the approach. In Section 3 articulated object motion and velocity are defined. In Section 4 a non-linear control law is derived for tracking articulated objects. In Section 5, several experimental results are given for different virtual links.

2 Overview and Motivations

The objective of the proposed approach is to maintain an estimate of a set of minimal parameters describing the configuration of an articulated object in $SE(n)$. This set of parameters are defined by a vector of n parameters $\mathbf{q} \in \mathbb{R}^n$. This vector is composed of subsets which fully describe the velocity of each component.

In order to maintain an estimate of $\mathbf{q}$, the underlying idea is to minimize a non-linear system of equations so that the projected contour of the object model in the image is aligned with the actual position of the contours in the image. This can be seen as the dual problem of visual servoing whereby minimizing the parameters corresponds to moving an arm-to-eye robot so as to observe the arm at a given position in the image (note that an object is not necessarily fixed to the ground). This duality, known as Virtual Visual Servoing has been explained in depth in previous papers [5, 12].

To perform the alignment, an error Δ is defined in the image between the projected features $\mathbf{s}(\mathbf{q})$ of the model and their corresponding features in the image $\mathbf{s_d}$ (desired features). The features of each component are projected using their associated camera poses ${}^c\mathbf{r}_{\mathcal{F}_1}(\mathbf{q})$ and ${}^c\mathbf{r}_{\mathcal{F}_2}(\mathbf{q})$ where each component's camera pose is composed of a subset of object parameters $\mathbf{q}$. In this paper distance features are used. This error is therefore defined as:

$$\Delta = \left(\mathbf{s}(\mathbf{q}) - \mathbf{s_d}\right) = \left[pr(\mathbf{q}, {}^o\mathbf{S}) - \mathbf{s_d}\right], \tag{1}$$

where ${}^o\mathbf{S}$ are the 3D coordinates of the *sensor* features in the object frame of reference. $pr(\mathbf{q}, {}^o\mathbf{S})$ is the camera projection model according to the object parameters $\mathbf{q}$.

The parameters of the object are initially needed and they are computed using the algorithm of Dementhon and Davis [7]. This algorithm is used to calculate the component's poses in the camera frame and they are calculated separately.

The parameters are projected into object space and variables in common between the components are averaged so that initialization errors are minimal.

In order to render the minimization of these errors more robust they are minimized using a robust approach based on M-estimation techniques.

$$\Delta_{\mathcal{R}} = \rho\Big(\mathbf{s}(\mathbf{q}) - \mathbf{s_d}\Big), \tag{2}$$

where $\rho(u)$ is a robust function [13] that grows sub-quadratically and is monotonically nondecreasing with increasing $|u|$. In this article Tukey's function is used because it allows complete rejection of outliers.

This is integrated into an iteratively re-weighted least squares (IRLS) minimization procedure so as to render those errors at the extremities of the distribution less likely.

3 Modeling

Articulated motion is defined as Euclidean transformations which preserve *subsets* of distances and orientation of object features.

The modeling of object motion is based on rigid body differential geometry. The set of rigid-body positions and orientations belongs to a Lie group, *SE*(3) (Special Euclidean group). These vectors are known as screws. The tangent space is the vector space of all velocities and belongs to the Lie algebra, *se*(3). This is the algebra of twists which is also inherent in the study of non-rigid motion. An articulated object, for example, must have a velocity contained in *se*(3), however, joint movement can be considered by sub-algebras of *se*(3).

The basic relation can be written which relates the movement of a sensor feature $\dot{\mathbf{s}}$ to the movement of the object parameters:

$$\dot{\mathbf{s}} = \mathbf{L_s}\mathbf{A}\dot{\mathbf{q}} \tag{3}$$

where

- $\mathbf{L_s}$ is called the feature Jacobian [14] or interaction matrix [15] between the camera and the sensor features $\mathbf{s}$.
- $\mathbf{A}$ is an Articulation matrix describing the differential relation between components.
- $\mathbf{L_s}\mathbf{A}$ being the Jacobian between the sensor and the entire object.

The Articulation matrix is the central issue in this paper. It corresponds to the mapping:

$$\mathbf{v} = \mathbf{A}\dot{\mathbf{q}} \tag{4}$$

where $\mathbf{v}$ is a vector of 'stacked' 6 dimensional twists each corresponding to the full motion in *se*(3) of each component.

The subsets of parameters which make up the object parameters are illustrated by a Venn diagram in Figure 3. In order that these sets can be obtained independently it is necessary to decouple their interaction. The only case where this occurs is in the joint frame of reference. Thus the following section considers the definition of a joint.

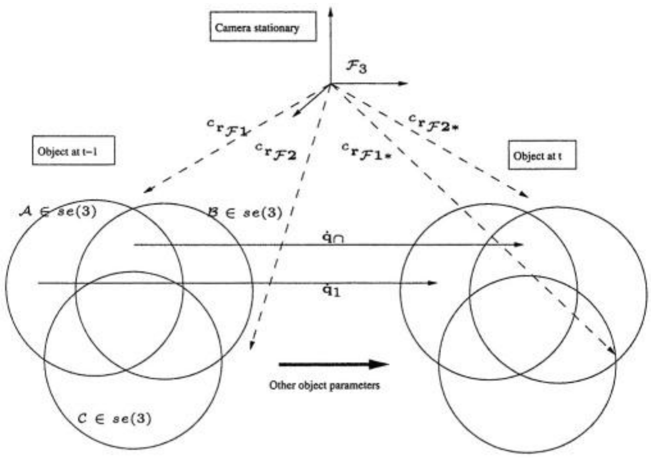


Fig. 3. Kinematic set method: The joint parameters are minimized in object space and kinematic set are used to decouple the system. Decoupling occurs at the intersection of parameter sets

3.1 Mechanical Joint Concept

A mechanical joint is fully defined by a matrix, vector pair which links two components. It is composed of a constraint matrix $\mathbf{S}^{\perp}$ which defines the type of the link and a pose vector $\mathbf{r}$ defining the position of the articulation. Thus the articulation matrix is:

$$\mathbf{A}_l(\mathbf{S}_l^{\perp}, \mathbf{r}_l), \tag{5}$$

where $\mathbf{S}_l^{\perp}$ corresponds to the configuration of joint l and $\mathbf{r}$ is a parameter vector describing the location of joint l.

The following two subsections explain the definition of these parameters.

3.2 Joint Configuration – $\mathbf{S}_l$

A joint configuration is fully defined by:

$$\mathbf{S}^{\perp}{}_l = \begin{pmatrix} s^{\perp}_{1,1} & \dots & s^{\perp}_{1,c} \\ \vdots & \ddots & \\ s^{\perp}_{6,1} & & s^{\perp}_{6,c} \end{pmatrix}, \tag{6}$$

The holonomic constraint matrix, $\mathbf{S}^{\perp}$, is defined such that each column vector defines one free degree of freedom at the corresponding link. The number of non-zero columns of $\mathbf{S}^{\perp}$ is referred to as the *class* c of the link. The rows of a column define the type of the link by defining which combination of translations and rotations are permitted as well as their proportions. In the experiments considered in Section 5 two different types of class 1 links are considered: A rotational link around the x axis:

$$\mathbf{S}^{\perp} = (0,0,0,1,0,0), \tag{7}$$

A helical link around and along the z axis:

$$\mathbf{S}^{\perp} = (0,0,a,0,0,1), \tag{8}$$

where the value of 'a' relates the translation along the z axis to a one rotation around the z axis.

The set of velocities that a first component can undertake which leaves a second component invariant is defined by $S^{\perp} \subset se(3)$. This is the orthogonal compliment of the sub-space $S \subset se(3)$ which constitutes the velocities which are in common between two components. Since a component, that is linked to another, is composed of these two subspaces it is possible to extract these subspaces by defining standard bases for the kernel and the image. The kernel is chosen to be $\mathbf{S}^{\perp}$ so that the image is given by (with abuse of notation):

$$\mathbf{S}_l = Ker\big((\mathbf{S}_l^{\perp})^T\big), \tag{9}$$

The matrix $\mathbf{S}_l$ and its orthogonal compliment $\mathbf{S}_l^{\perp}$ can be used to project the kinematic twist (velocities) onto two orthogonal subspaces (For more than 1 joint it is necessary to project onto a common vector basis).

Thus a subspace projection matrix is given as:

$$\begin{aligned} \mathbf{P}_l &= \mathbf{S}_l\mathbf{S}_l^{+}, \\ \mathbf{P}_l^{\perp} &= \mathbf{S}_l^{\perp}\mathbf{S}_l^{\perp +} = \mathbb{I}_6 - \mathbf{P}_l, \end{aligned} \tag{10}$$

where $\mathbf{S}^{+} = (\mathbf{S}^T\mathbf{S})^{-1}\mathbf{S}^T$ is the pseudo-inverse of $\mathbf{S}$.

This ensures that the resulting projected velocities are defined according to a common basis defined by the parameters of the pose vector in equation (6). This then allows the twist transformations, given in the following section, to be applied to these quantities.

3.3 Joint Location – $\mathbf{r}_l$

A joint location is fully defined by a pose vector:

$$\mathbf{r}_l = {}^{\mathcal{F}_c}\mathbf{r}_l = (t_x, t_y, t_z, \theta_x, \theta_y, \theta_z), \tag{11}$$

where $\mathcal{F}_c$ indicates the camera frame and l represents the joint frame.

A joint configuration is only valid in the joint reference frame. A kinematic twist transformation matrix is used to obtain the velocity of the joint frame w.r.t its previous position. The Lie algebra provides the transformation of vector quantities as $\mathbf{V}(\mathbf{r})$. This is a kinematic twist transformation from frame a to frame b given as:

$${}^a\mathbf{V}_b = \begin{bmatrix} {}^a\mathbf{R}_b & [{}^a\mathbf{t}_b]_{\times}^{a}\mathbf{R}_b \\ 0_3 & {}^a\mathbf{R}_b \end{bmatrix}, \tag{12}$$

where ${}^a\mathbf{R}_b$ is a rotation matrix between frames and ${}^a\mathbf{t}_b$ a translation vector between frames which are obtained from ${}^a\mathbf{r}_b$. $[\mathbf{t}]_x$ is the skew symmetric matrix related to $\mathbf{t}$.

The projector defined in (10) is applied in the joint reference frame. It is possible to choose the object frame as a common reference frame as in [16]. In this paper the camera frame is chosen as the common reference frame so that a generic subspace projection operators $\mathbf{J}_l$ and $\mathbf{J}_l^{\perp}$ can be defined as:

$$\begin{aligned} \mathbf{J}_l &= Im({}^c\mathbf{V}_l\ \mathbf{P}_l\ {}^l\mathbf{V}_c), \\ \mathbf{J}_l^{\perp} &= Ker(\mathbf{J}) = Im({}^c\mathbf{V}_l\ \mathbf{P}_l^{\perp}\ {}^l\mathbf{V}_c), \end{aligned} \tag{13}$$

where Im represents the Image operator which reduces the column space to its mutually independent basis form. The first transformation $\mathbf{V}$ maps the velocities to the joint frame l and the second re-maps back to the camera reference frame.

3.4 Articulation Matrix

Using the previous joint definition is is possible to define the Articulation matrix according to equation (4) and taking into account the joint subspaces given by equation (13).

The derivation of the Articulation matrix corresponds to:

$$\mathbf{A} = \begin{pmatrix} \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}} \\ \vdots \\ \frac{\partial \mathbf{r}_m}{\partial \mathbf{q}} \end{pmatrix}, \tag{14}$$

where m is the number of components.

For an object with two components and one jointand using the orthogonal subspace projectors given in equation (13), $\mathbf{A}$ is given by:

$$\mathbf{A} = \begin{pmatrix} \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_\cap} & \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}_1} & 0 \\ \frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_\cap} & 0 & \frac{\partial \mathbf{r}_2}{\partial \mathbf{q}_2} \end{pmatrix} = \begin{pmatrix} \mathbf{J}_1 & \mathbf{J}_1^{\perp} & 0 \\ \mathbf{J}_1 & 0 & \mathbf{J}_1^{\perp} \end{pmatrix}, \tag{15}$$

where $\mathbf{q}_\cap$, $\mathbf{q}_1$, $\mathbf{q}_2$ are vectors representing the sets of intersecting velocities and each components free parameters respectively. These sets are easily identified when referring to Figure 3. Given $dim(\mathbf{J}_1) = 6 - c$ and $dim(\mathbf{J}_1^{\perp}) = c$, the mapping $\mathbf{A}$ is indeed dimension $12 \times (6 + c)$, remembering that c is the class of the link. The derivation of objects with more than one joint follows in a similar manner and is left to the reader.

It is important to note that this method introduces decoupling of the minimization problem. This is apparent in equation (15) where extra zeros appear in the Jacobian compared to the traditional case of a kinematic chain. Indeed, in the particular case of two components and one articulation a kinematic chain has only one zero.

4 Registration

In this section a new tracking control law is derived. The aim of the control scheme is to minimize the objective function given in equation (2). Thus, the error function is given as:

$$\begin{pmatrix} \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_m \end{pmatrix} = \mathbf{D} \begin{pmatrix} \mathbf{s}_1(\mathbf{q}) - \mathbf{s}_{\mathbf{d}1} \\ \vdots \\ \mathbf{s}_m(\mathbf{q}) - \mathbf{s}_{\mathbf{d}m} \end{pmatrix}, \quad (16)$$

where $\mathbf{q}$ is a vector composed of the minimal set of velocities corresponding to the object's motion and each $\mathbf{e}_i$ corresponds to an error vector for component i. $\mathbf{D}$ is a diagonal weighting matrix corresponding to the likelihood of a particular error within the robust distribution:

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & & 0 \\ & \ddots & \\ 0 & & \mathbf{D}_m \end{pmatrix},$$

where each matrix $\mathbf{D}_i$ is a diagonal matrix corresponding to a component which has weights w_j along the diagonal. These weights correspond to the uncertainty of the measured visual feature j. The computation of the weights are described in [5].

If $\mathbf{D}$ were constant, the derivative of equation (16) would be given by:

$$\begin{aligned} \begin{pmatrix} \dot{\mathbf{e}}_1 \\ \vdots \\ \dot{\mathbf{e}}_i \end{pmatrix} &= \begin{pmatrix} \frac{\partial \mathbf{e}_1}{\partial \mathbf{s}_1} \frac{\partial \mathbf{s}_1}{\partial \mathbf{r}_1} \frac{\partial \mathbf{r}_1}{\partial \mathbf{q}} \\ \vdots \\ \frac{\partial \mathbf{e}_m}{\partial \mathbf{s}_m} \frac{\partial \mathbf{s}_m}{\partial \mathbf{r}_m} \frac{\partial \mathbf{r}_m}{\partial \mathbf{q}} \end{pmatrix} \dot{\mathbf{q}} \\ &= \mathbf{D}\mathbf{L}_\mathbf{s}\mathbf{A}\dot{\mathbf{q}} \end{aligned} \quad (17)$$

where $\dot{\mathbf{q}}$ is the minimal velocity vector, $\mathbf{A}$ is the articulation matrix describing the mapping in equation (4) and $\mathbf{L}_\mathbf{s}$ is the 'stacked' interaction matrix as in equation (3) given as:

$$\mathbf{L}_\mathbf{s} = \begin{pmatrix} \frac{\partial \mathbf{s}_1}{\partial \mathbf{r}_1} \\ \vdots \\ \frac{\partial \mathbf{s}_m}{\partial \mathbf{r}_m} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{\mathbf{s}1} & & \mathbf{0}_6 \\ & \ddots & \\ \mathbf{0}_6 & & \mathbf{L}_{\mathbf{s}m} \end{pmatrix}, \quad (18)$$

If an exponential decrease of the error $\mathbf{e}$ is specified:

$$\dot{\mathbf{e}} = -\lambda \mathbf{e}, \quad (19)$$

where λ is a positive scalar, the following control law is obtained by combining equation (19) and equation (17):

$$\dot{\mathbf{q}} = -\lambda(\widehat{\mathbf{D}}\widehat{\mathbf{L}_\mathbf{s}}\widehat{\mathbf{A}})^{+}\widehat{\mathbf{D}}(\mathbf{s}(\mathbf{q}) - \mathbf{s}_\mathbf{d}), \quad (20)$$

where $\widehat{\mathbf{L}_s}$ is a model or an approximation of the real matrix $\mathbf{L_s}$. $\widehat{\mathbf{D}}$ a chosen model for $\mathbf{D}$ and $\widehat{\mathbf{A}}$ depends on the previous pose estimation.

For the example of one joint given in equation (15), the sets of velocities to be estimated are:

$$\dot{\mathbf{q}} = \left(\dot{\mathbf{q}}_\cap, \dot{\mathbf{q}}_1, \dot{\mathbf{q}}_2\right), \tag{21}$$

Once these velocities are obtained they can be related back to the camera frame as in equation (4):

$$\begin{pmatrix} {}^c\mathbf{v}_1 \\ {}^c\mathbf{v}_2 \end{pmatrix} = \mathbf{A} \begin{pmatrix} \dot{\mathbf{q}}_\cap \\ \dot{\mathbf{q}}_1 \\ \dot{\mathbf{q}}_2 \end{pmatrix} \tag{22}$$

5 Results

In this section three experiments are presented for tracking of articulated objects in *real* sequences. Both camera and object motion as well as articulated motion have been introduced into each experiment. The complex task of implementing this algorithm was a major part of the work. Indeed this required correct modeling of features of type distance to lines, correct modeling of feature sets and correct implementation of the interaction between these feature sets represented as a graph of feature sets.

The rigid tracking method used here is based on a monocular vision system. Local tracking is performed via a 1D oriented gradient search to the normal of parametric contours at a specified sampling distance. This 1D search provides real-time performance. Local tracking provides a redundant group of distance to contour based features which are used together in order to calculate the global pose of the object. The use of redundant measures allows the elimination of noise and leads to high estimation precision. These local measures form an objective function which is minimized via a non-linear minimization procedure using virtual visual servoing(VVS) [5]. These previous results demonstrate a general method for deriving interaction matrices for any type of distance to contour and also show the robustness of this approach with respect to occlusion and background clutter.

The basic implementation of the algorithm gives the following pseudo-code:

```
1. Obtain initial pose.
2. Acquire new image and project the model onto the image.
3. Search for corresponding points normal to the projected contours.
4. Determine the error $\mathbf{e}$ in the image.
5. Calculate $(\widehat{\mathbf{D}}\widehat{\mathbf{H}}\widehat{\mathbf{A}})$.
6. Determine set velocities as in equation (20) and then component positions.
7. Repeat to 4 until the error converges.
8. Update the pose parameters and repeat to 3.
```

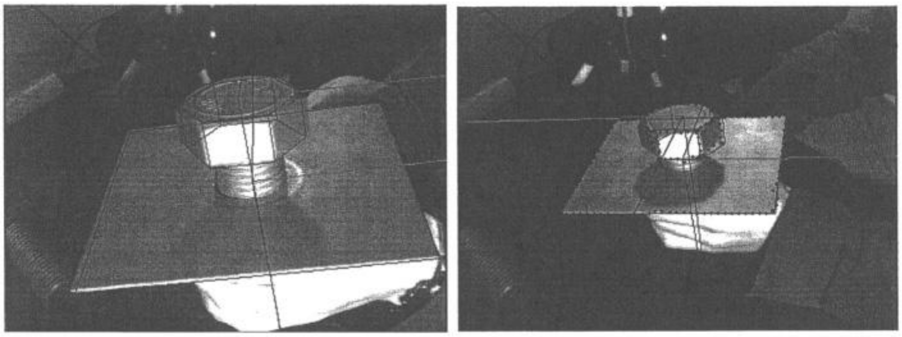


Fig. 4. Helical movement of a screw whilst the screw and the platform are simultaneously in movement.In this and all the following figures the reference frames for each component are shown as well as a projection of the CAD model onto the image. The axes of the frames are drawn in yellow, blue and red. The contour of the object is shown in blue. The points found to the normal of the contour are in red and the points rejected by the M-estimator are shown in green

5.1 Helical Link

This first experiment, reported in Figure 4 was carried out for class one link with helical movement simultaneously along and around the z axis. The constraint vector was defined as in equation (8) and the object frame was chosen to coincide with the joint frame. Note that the constraint vector was defined by taking into consideration that for $10 \times 2\pi$ rotations of the screw it translated 4.5cm along the z axis.

Tracking of this object displayed real time efficiency with the main loop computation taking on average 25ms per image. It should be noted that tracking of the screw alone as a rigid object fails completely due to the limited contour information and difficult self occulsions. When tracked simultaneously with the plate as an articulated object the tracking of the screw is also based on the measurements of the plate making the tracking possible. M-estimation was carried out separately for each component.

5.2 Robotic Arm

A recent experiment was carried out for two class one links on a robotic arm. The articulations tracked were rotational links around the z and x axes. The constraint vectors were each defined by a pose and a constraint matrix as given in equation (7). Note that the constraints are no longer defined in the same coordinate system as in the previous case. This sequence also displays real time efficiency with the tracking computation taking on average 25ms per image. It should be noted that the features used on the components of the arm are not full rank and do not hold enough information to calculate their pose individually. As in the previous experiment, the articulated tracker overcomes this situation.

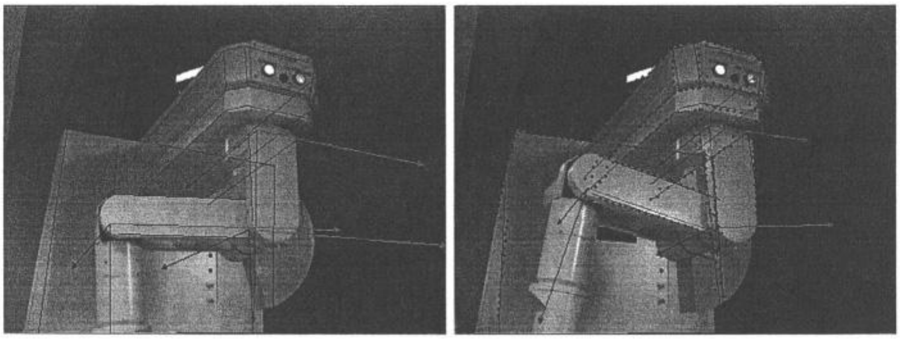


Fig. 5. Movement of a robotic arm with two degrees of freedom

6 Conclusion

The method presented here demonstrates an efficient approach to tracking complex articulated objects. A framework is given for defining any type of mechanical link between components of a object. A method for object-based tracking has been derived and implemented. Furthermore, a kinematic set formulation for tracking articulated objects has been described. It has been shown that it is possible to decouple the interaction between articulated components using this approach. Subsequent computational efficiency and visual precision have been demonstrated.

In perspective, automatic initialization methods could be considered using partially exhaustive RANSAC [17] based techniques. A better initial estimate could also be obtained using a kinematic chain formulation.

References

[1] Aggarwal, J., Cai, Q., Liao, W., Sabata, B.: Nonrigid motion analysis: Articulated and elastic motion. Computrer Vision and Image Understanding **70** (1998) 142–156

[2] Lowe, D.: Three-dimensional object recognition from single two-dimensional images. Artificial Intelligence **31** (1987) 355–394

[3] Drummond, T., Cipolla, R.: Real-time visual tracking of complex structures. IEEE Trans. on Pattern Analysis and Machine Intelligence **27** (2002) 932–946

[4] Marchand, E., Bouthemy, P., Chaumette, F., Moreau, V.: Robust real-time visual tracking using a 2d-3d model-based approach. In: IEEE Int. Conf. on Computer Vision, ICCV'99. Volume 1., Kerkira, Greece (1999) 262–268

[5] Comport, A.I., Marchand, E., Chaumette, F.: A real-time tracker for markerless augmented reality. In: ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'03, Tokyo, Japan (2003) 36–45

[6] Dhome, M., Richetin, M., Lapresté, J.T., Rives, G.: Determination of the attitude of 3-d objects from a single perspective view. IEEE Trans. on Pattern Analysis and Machine Intelligence **11** (1989) 1265–1278

[7] Dementhon, D., Davis, L.: Model-based object pose in 25 lines of codes. Int. J. of Computer Vision **15** (1995) 123–141

[8] Lu, C., Hager, G., Mjolsness, E.: Fast and globally convergent pose estimation from video images. IEEE Trans. on Pattern Analysis and Machine Intelligence **22** (2000) 610–622

[9] Lowe, D.: Fitting parameterized three-dimensional models to images. IEEE Trans. on Pattern Analysis and Machine Intelligence **13** (1991) 441–450

[10] Nunomaki, T., Yonemoto, S., Arita, D., Taniguchi, R.: Multipart non-rigid object tracking based on time model-space gradients. Articulated Motion and Deformable Objects First International Workshop (2000) 78–82

[11] Ruf, A., Horaud, R.: Rigid and articulated motion seen with an uncalibrated stereo rig. In: IEEE Int. Conf. on Computer Vision, Corfu, Greece (1999) 789–796

[12] Marchand, E., Chaumette, F.: Virtual visual servoing: a framework for real-time augmented reality. In: EUROGRAPHICS'02 Conference Proceeding. Volume 21(3) of Computer Graphics Forum., Saarebrücken, Germany (2002) 289–298

[13] Huber, P.J.: Robust Statistics. Wiler, New York (1981)

[14] Hutchinson, S., Hager, G., Corke, P.: A tutorial on visual servo control. IEEE Trans. on Robotics and Automation **12** (1996) 651–670

[15] Espiau, B., Chaumette, F., Rives, P.: A new approach to visual servoing in robotics. IEEE Trans. on Robotics and Automation **8** (1992) 313–326

[16] Comport, A.I., Marchand, E., Chaumette, F.: Object-based visual 3d tracking of articulated objects via kinematic sets. In: IEEE Workshop on Articulated and Non-Rigid Motion, Washington, DC (2004)

[17] Fischler, N., Bolles, R.: Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. Communication of the ACM **24** (1981) 381–395

2D Human Tracking by Efficient Model Fitting Using a Path Relinking Particle Filter

Juan José Pantrigo[1], Ángel Sánchez[1], Kostas Gianikellis[2], and Antonio S. Montemayor[1]

[1]Universidad Rey Juan Carlos, c/ Tulipán s/n
28933 Móstoles, Spain
{j.j.pantrigo,an.sanchez,a.sanz}@escet.urjc.es
[2]Universidad de Extremadura, Avda. Universidad s/n
10071 Cáceres, Spain
kgiannik@unex.es

Abstract. This paper presents a 2D model-based Path Relinking Particle Filter (PRPF) algorithm for human motion tracking and analysis applications. PRPF algorithm hybridizes both Particle Filter and Path Relinking frameworks. The proposed algorithm increases the performance of general Particle Filter by improving the quality of the estimate, by adapting computational load to problem constraints and by reducing the number of required evaluations of the weighting function. A 2D human body model was used to develop tracking with PRPF. This model consists of a simple hierarchical set of articulated limbs, which is described by geometrical parameters. It is easily adaptable to the tracking application requirements. We have applied the PRPF algorithm to 2D human pose estimation in different movement tracking activities such as walking and jumping. Qualitative experimental results show that the model-based PRPF is appropriate in 2D human pose estimation studies.

1 Introduction

Automatic visual analysis of human motion is an active research topic in Computer Vision and its interest has been growing in the last decade [1][2][3][4]. Analysis and synthesis of human motion has numerous applications. In Visual Surveillance, gait recognition has been used in access control systems [1]. In Advanced User Interfaces, visual analysis of human movement is applied in detecting human presence and interpreting human behaviour [1]. Human motion analysis in Medicine can be employed to characterize and diagnose certain types of disorders [4]. Finally, visual analysis of human movement is also used in Biomechanics, studying human body behavior subject to mechanical loads in three main areas: medical, sports and occupational [5].

Human body is usually represented as an articulated model. This is due to the fact that it consists of a set of limbs linked one to each other at joints which allow different movements of these limbs [6]. Most studies in human motion analysis are based on articulated models that properly describe the human body [2] [6] [7] [8]. Model-based tracking allows extracting body posture in an effortless way and handling occlusions.

2D contour representation of human body is relevant in the extraction of the human body projection in the image plane. In this description, human body segments are similar to 2D ribbons or blobs. In the work by Ju [7] a cardboard people model was proposed. Human body segments were modelled by planar patches. Leung and Yang [9] used a 2D ribbons with U-shaped edge segments. Rohr [6] proposed a 2D motion model in which a set of analytically motion curves represented the postures.

One particular pose of the subject can be expressed as a single point in a state-space. In this n-dimensional space each axis represents a degree of freedom (DOF) of a joint in the model. Thus, all possible solutions to the pose estimation problem are represented as points in this state-space. The goal of the model is to connect the state-space with the 2D image space. This is achieved by creating a set of synthetic model images and comparing them to measurements taken at each frame thus obtaining a similarity frame estimate. Low level features such as blobs (silhouette), edges (contours), colour and movement have been widely used in diverse approaches [2].

There are several methods for the comparison between synthetic data and frame measurements. A usual approach, given by a Kalman Filter, predicts just one state and estimates the difference between the synthetic data and the measurements data [2]. Another approach, given by a Particle Filter algorithm, predicts the most likely states using a multiple hypothesis framework. The Particle Filter (PF) algorithm, (also termed as Condensation algorithm) enables the modelling of a stochastic process with an arbitrary probability density function (pdf), by approximating it numerically with a set of points called particles in a process state-space [10].

The problem with using an articulated model for human body representation is the high dimensionality of the state-space and the high computational effort it supposes [11]. Also, in the Condensation approach, the number of required particles grows with the size of the state-space, as demonstrated in [12]. To address this difficulty, several optimized PF algorithms have been proposed. They use different strategies to improve their performance. Deutscher [11] developed an algorithm termed Annealed Particle Filter (APF) for tracking people. This filter works well for full-body models with 29 DOFs. Partitioned Sampling (PS) [12] is a statistical approach to tackle hierarchical search problems. PS consists by dividing the state space into two or more partitions, and sequentially applying the stated dynamic model for each partition followed by a weighted resampling stage.

We propose a model-based Path Relinking Particle Filter (PRPF) [13] applied to the visual tracking. This algorithm is inspired by the Path Relinking Metaheuristic proposed by Glover [14][15] as a way to integrate intensification and diversification strategies in the context of combinatorial optimization problems. PRPF hybridizes both Particle Filter (PF) and Path Relinking (PR) frameworks in two different stages. In the PF stage, a particle set is propagated and updated to obtain a new particle set. In PR stage, an elite subset (called *RefSet*) from the particle set is selected, and new solutions are constructed by exploring trajectories that connect each of the particles in the *RefSet*. Also, a geometrical model is used to represent the human body in a 2D space as a hierarchical set of articulated limbs, which allows us to obtain pose from images at any time. We represent the shape of the human body by a set of plane trapeziums and ellipses connected by joints. Moreover, this model is simple, the number of required parameters is small and it is easily adaptable to describe different articulated objects.

2 Particle Filter

General particle filters (PF) are sequential Monte Carlo estimators based on particle representations of probability densities, which can be applied to any state-space model [16]. The state-space model consists of two processes: (i) an observation process $\mathbf{p}(\mathbf{Z}_t|\mathbf{X}_t)$, where $\mathbf{X}$ denotes the system state vector and $\mathbf{Z}$ is the observation vector, and (ii) a transition process $\mathbf{p}(\mathbf{X}_t|\mathbf{X}_{t-1})$. Assuming that observations $\{\mathbf{Z}_0, \mathbf{Z}_1, \ldots, \mathbf{Z}_t\}$ are sequentially measured in time, the goal is to estimate the new system state $\{\chi_0, \chi_1, \ldots, \chi_t\}$ at each time. In the framework of Sequential Bayesian Modelling, posterior pdf is estimated in two stages:

(i) Prediction: the posterior pdf $\mathbf{p}(\mathbf{X}_{t-1}|\mathbf{Z}_{t-1})$ is propagated at time step *t* using the Chapman-Kolmogorov equation:

$$p(\mathbf{X}_t \mid \mathbf{Z}_{t-1}) = \int p(\mathbf{X}_t \mid \mathbf{X}_{t-1}) p(\mathbf{X}_{t-1} \mid \mathbf{Z}_{t-1}) d\mathbf{X}_{t-1} \tag{1}$$

A predefined system model is used to obtain an updated particle set.

(ii) Evaluation: posterior pdf $\mathbf{p}(\mathbf{X}_t|\mathbf{Z}_t)$ is computed using the observation vector $\mathbf{Z}_t$:

$$p(\mathbf{X}_t \mid \mathbf{Z}_t) = \frac{p(\mathbf{Z}_t \mid \mathbf{X}_t) p(\mathbf{X}_t \mid \mathbf{Z}_{t-1})}{p(\mathbf{Z}_t \mid \mathbf{Z}_{t-1})} \tag{2}$$

In Figure 1 an outline of the Particle Filter scheme is shown. The aim of the PF algorithm is the recursive estimation of the posterior pdf $\mathbf{p}(\mathbf{X}_t|\mathbf{Z}_t)$, that constitutes a complete solution to the sequential estimation problem. This pdf is represented by a set of weighted particles $\{(\mathbf{x}_t^0, \pi_t^0) \ldots (\mathbf{x}_t^N, \pi_t^N)\}$, where the weights $\pi_t^n \propto \mathbf{p}(\mathbf{Z}_t| \mathbf{X}_t = \mathbf{x}_t^n)$ are normalised. The state χ_t can be estimated by the equation:

$$\chi_t = \sum_{n=1}^{N} \pi_t^n \mathbf{x}_t^n \tag{3}$$

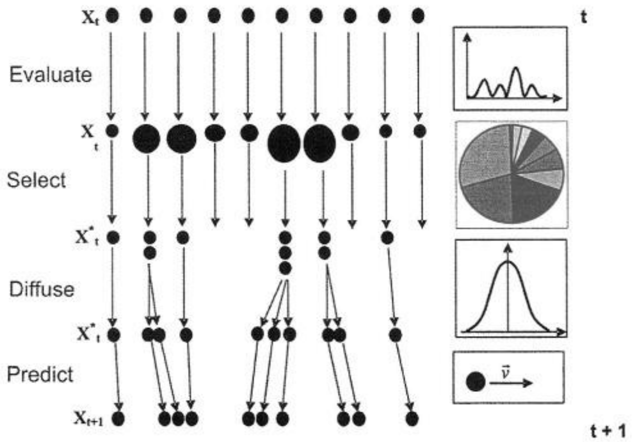


Fig. 1. Particle Filter scheme

PF starts by setting up an initial population $\mathbf{X}_0$ of N particles using a known pdf. The measurement vector $\mathbf{Z}_t$ at time step t, is obtained from the image. Particle weights $\mathbf{\Pi}_t$ are computed using the weighting function. Weights are normalized and a new particle set $\mathbf{X}^*_t$ is selected. As particles with larger weight values can be chosen several times, a diffusion stage are applied to avoid the loss of diversity in $\mathbf{X}^*_t$. Finally, particle set at time step $t+1$, $\mathbf{X}_{t+1}$, is predicted using the motion model. A pseudocode of a general PF is detailed in [13][16].

3 Path Relinking

Path Relinking (PR) [14][15] is an evolutionary metaheuristic in the context of the combinatorial optimization problems. PR constructs new high quality solutions by combining other previous solutions based on the exploration of paths that connect them. To yield better solutions than the original ones, PR starts from a given set of elite candidates, called *RefSet* (short for "Reference Set"). These solutions are selected through a search process and are ordered according to their corresponding qualitative values. New candidates are then generated, by exploring trajectories that connect solutions in the *RefSet*. The metaheuristic starts with two of these solutions x' and x'', and it generates a path $x' = x(1), x(2), \ldots, x(r) = x''$ in the neighbourhood space that leads toward the new sequence. In order to produce better quality solutions, it is convenient to add a local search optimization phase. A pseudocode of PR can be found in [13][14].

4 Path Relinking Particle Filter

Path Relinking Particle Filter (PRPF) algorithm was introduced in [13] to be applied to estimation problems in sequential processes that can be expressed using the state-space model abstraction. As pointed out in section 1, PRPF integrates both PF and PR frameworks in two different stages. The PRPF algorithm is centered on a delimited region of the state-space in which it is highly probable to find new better solutions than the initial ones. PRPF increases the performance of general PF by improving the quality of the estimate, adapting computational load to constraints and reducing the number of required evaluations of the particle weighting function. Fig. 2. shows a graphical template of the PRPF method. Dashed lines separate the two main components in the PRPF scheme: PF and PR optimization, respectively.

PRPF starts with an initial population of N particles drawn from a known pdf. Each particle represents a possible solution of the problem. Particle weights are computed using a weighting function and a measurement vector. PR stage is later applied improving the best obtained solutions of the particle filter stage. A *RefSet* is created selecting the b ($b \ll N$) best particles. New solutions are generated and evaluated, by exploring trajectories that connect all possible pairs of particles in the *RefSet*. In order to improve the solution fitness, a local search from some of the generated solutions within the PR procedure is performed. PR stage ends when the news generated solutions do not improve the quality of the *RefSet*.

Once the PR stage is over, the "worst" particles are replaced with the *RefSet* solutions. Then, a new population of particles is created by selecting the individuals from the whole particle set with probabilities according to their weights. In order to avoid the loss of diversity, a diffusion stage is applied to the particles of the new set. At the end, particles are projected into the next time step by making use of the update rule. The pseudocode of PRPF algorithm for visual tracking is detailed in [13].

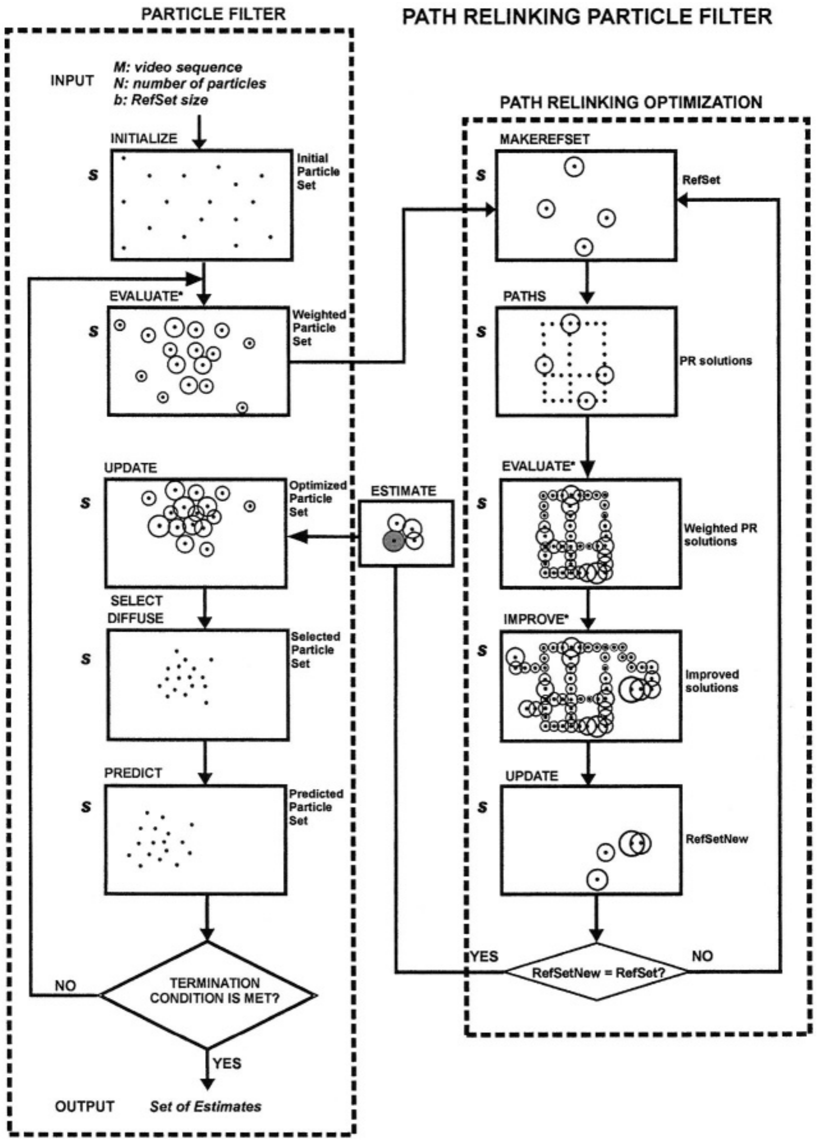


Fig. 2. Path Relinking Particle Filter scheme. Actual frame measures are required during EVALUATE and IMPROVE stages (*)

PRPF estimator quality is improved with respect to PF and the required number of evaluations for the weighting function is also reduced. Therefore, PRPF search in the state-space is not performed randomly like in a general particle filter. PRPF is time-adaptive since the number of evaluations of the weighting function changes in each time step. If the initial solutions in the *RefSet* are far away one from each other, then paths connecting solutions became long enough, and the number of explored solutions increases. It is not possible to have any estimate of the previous state of the system at the beginning of the visual tracking, therefore the particle filter is usually randomly initialized. The number of individuals in the particle filter does not change during the algorithm execution. PRPF algorithm reduces the total required number of evaluations of the weighting function when increasing the number of total time steps.

5 Models for Human Pose Estimation

Each one of the involved models in our framework is detailed in this section. A geometrical model is required to link solutions in the state-space with 2D image feature extraction. Observation and system models respectively define the observation and transition process in the state-space model abstraction.

5.1 Geometrical Model

We use an a priori 2D geometrical model to represent the observed subject. It consists of a hierarchical set of articulated limbs. This model stores geometrical (time-independent) parameters describing the body components. Fig. 3. illustrates the proposed blobs and edge models for upper-body tracking. As shown in the experiments section, this model can be easily extended to describe the full human body.

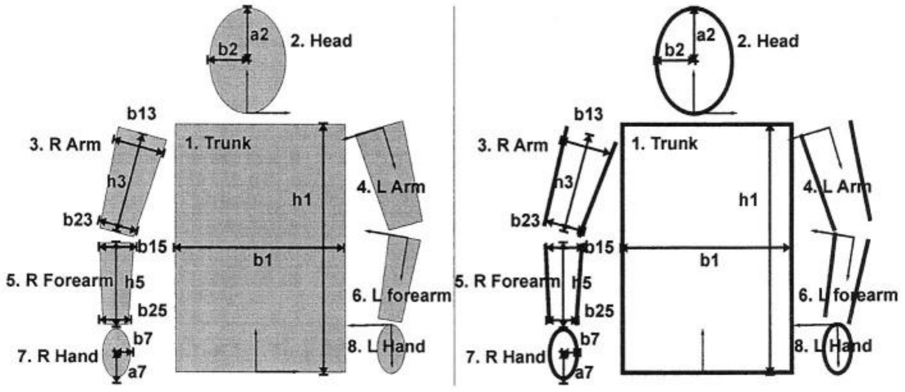


Fig. 3. Proposed blob (left) and edge (right) configuration for human upper-body model

Table 1. Limb properties in a human upper-body model

	Trunk	Head	R. arm	R. forearm	R. hand
Ident.	1	2	3	5	7
Shape	T	E	T	T	E
Level	1	2	2	3	4
Father		1	1	3	5
Size	[h1, b1, b1]	[a2, b2]	[h1, b13, b23]	[h1, b15, b25]	[a7, b7]
Pos.		[0, h1+Δ]	[-b1/2, h1-b12/2]	[0, h3]	[0, h5]

Body limbs are represented by a set of trapezium-shaped (trunk, arms, legs, and feet) and ellipse-shaped (head and hands) ribbons which are connected by joints. *Size* of trapeziums (T) is described by three parameters: one for the length and two for the axes. *Size* of ellipses (E) is described by two axes. Each limb is jointed with a *father* limb except trunk. *Position* and *orientation* of each body part is described in his father frame. The coordinate system for the body parts are aligned with the natural axes. The origin of a coordinate system is located at the point in which each limb is jointed with his father limb. *Level* of the limb is related to the distance from the body center, and it is useful to calculate position and orientation of body parts in the global reference system. Several examples of limbs descriptions in the proposed model are shown in Table 1.

Particles store time-dependent values relating to limb positions, orientations and velocities. The state $\mathbf{x}_t^i$ of a particle $(\mathbf{x}_t^i, \pi_t^i)$ in an eight-limb model is described as:

$$\left[x_1, y_1, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \dot{x}_1, \dot{y}_1, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5, \dot{\theta}_6, \dot{\theta}_7, \dot{\theta}_8\right] \tag{4}$$

where x and y are the spatial positions, θ is the limb orientation in the father's system of reference and $\dot{x}$, $\dot{y}$ and $\dot{\theta}$ represents the first derivative of it's corresponding variable.

The goal of the geometrical model is to connect solutions in the multi-dimensional state-space with the 2D image features. Thus, the method predicts the pose of the model for the next frame and creates edge and blobs synthetic images. Note that used tracking model parameters are defined with respect to the camera view point. Features, those extracted from each frame in the video sequence and those predicted by the PRPF, are compared in order to obtain a corresponding similarity measure. This similarity value is iteratively used to establish the weights of the different particles for the following frame during the tracking stage.

5.2 Observation Model and Weighting Function

The observation model specifies the image features to be extracted. To construct the weighting function it is necessary to use adequate image features. In controlled environments, edges and silhouette are relatively easy to extract from both, the image and the geometrical model. Continuous edges extracted from a human image usually provide a good measure of visible body limbs. However, they are sensitive to noise. A region-based feature such as silhouette has the advantage over edges of being less sensitive to noise [2]. On the other hand, details may be lost in the extraction of silhouettes. In order to overcome these difficulties both a silhouette and an edge based model are used.

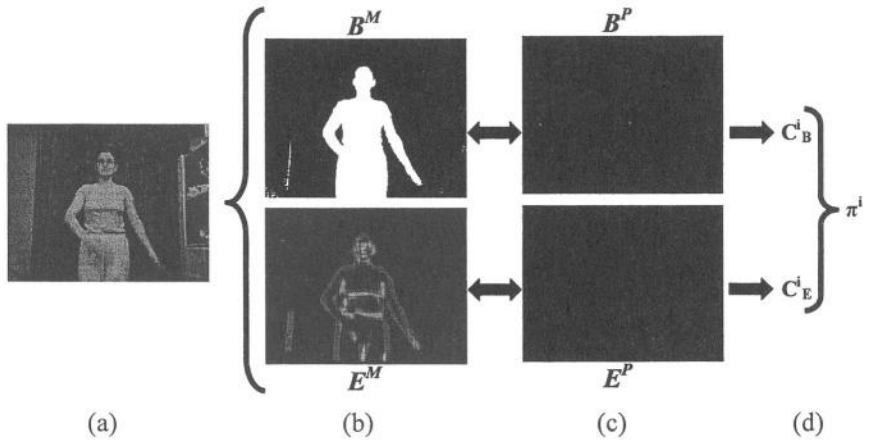


Fig. 4. Observation process: (a) initial image, (b) feature extraction, (c) particle prediction and (d) particle weight computation

Fig. 4. represents the observation process that leads to the particle weights computation. Continuous edges extracted from a human image usually provide a good measure of visible body limbs. We use a Canny edge method to extract edges in the human body image. The resulting edges are then smoothed using a convolution operation. This produces a pixel map E^M in which each pixel is set to a value related to its proximity to an edge. Another pixel map E^P is built extracting edges produced by the geometrical model of the configuration predicted by the *ith* particle, for each pixel *j* in the pixel map. Similarly, background subtraction was used to obtain human silhouette. Two pixel maps B^M and B^P are built and compared to compute the corresponding values of C^i_B. Differences between these two maps are computed by:

$$\forall i \in \{1..N_{part}\}, \forall j \in \{1..N_{pixel}\}: C^i_E = \sum_j \left| E^M_j - E^B_j \right| \tag{5}$$

$$\forall i \in \{1..N_{part}\}, \forall j \in \{1..N_{pixel}\}: C^i_B = \sum_j \left| B^M_j - B^B_j \right| \tag{6}$$

Finally, edges and blobs coefficients are combined to obtain *ith* particle weight at each frame using the following weighting function:

$$\forall i \in \{1..N_{part}\}, \pi^i = e^{-\alpha(C^i_E + C^i_B)} \tag{7}$$

where α is an experimental parameter.

5.3 System Model

System model describes how particles evolve over the time. The update rule used in this work is:

$$\begin{cases} x_{t+\Delta t} = x_t + \dot{x}_t \Delta t \\ \dot{x}_{t+\Delta t} = \dot{x}_t + G_x \Delta t \end{cases} \tag{8}$$

where x represents some spatial (linear or angular) variable, Δt is the temporal step and G_X is a random Gaussian variable with zero mean and normal deviation σ_X.

6 Experimental Results

In order to analyse the performance of the proposed model-based PRPF system, people performing different activities were recorded in different scenarios. PRPF algorithm was implemented using MATLAB 6.1. Fig. 5. shows the model adjustment for a subject performing planar movements. Upper-body model consists of eight limbs. Right elbow angle estimation using PRPF, Sampling Importance Resampling (SIR) and manual digitizing curves are shown in Fig. 6.

Fig. 7. shows a runner tracked with a ten limbs full-body model. This sequence demonstrates that the model adjustment is accurate. In Figure 8 a countermovement jump sequence is shown. A full-body model formed by only five limbs is employed. Selected "non consecutive" frames are shown in both figures. Right knee (left) and hip (right) angle estimation using PRPF and manual digitizing curves are shown in Figure 9. Finally, Table 2 shows the MSE values of some calculated angles from frontal and jump sequences.

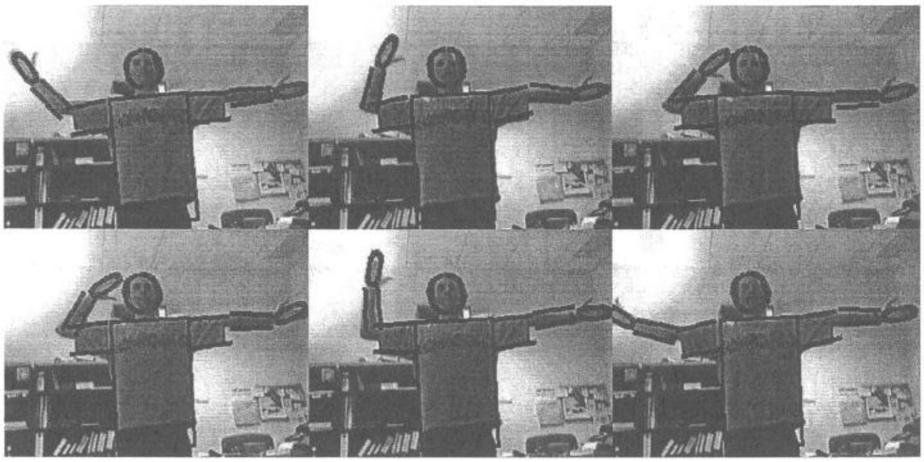


Fig. 5. Visual model adjustment for a subject performing planar movements (frames # 1, 10, 20, 30, 40 and 50)

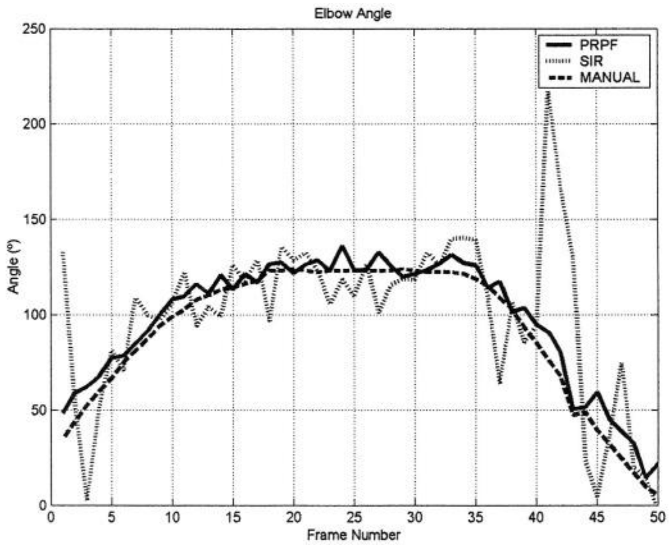


Fig. 6. Right elbow angle estimation using PRPF, SIR and manual digitizing

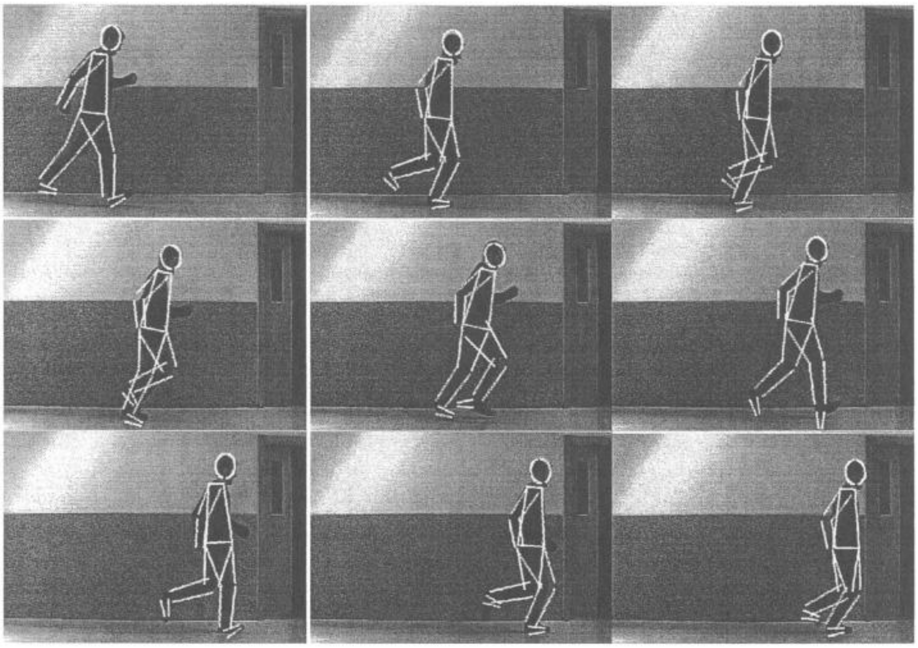


Fig. 7. Visual model adjustment for a running man

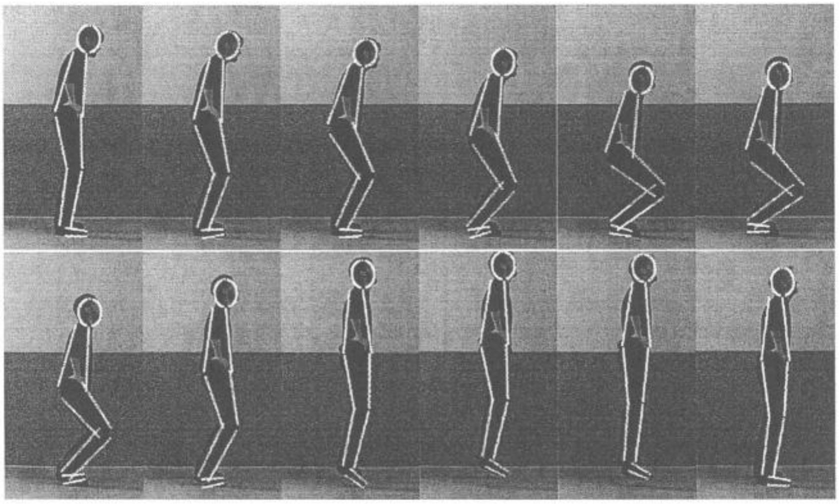


Fig. 8. Visual model adjustment for a jumping man

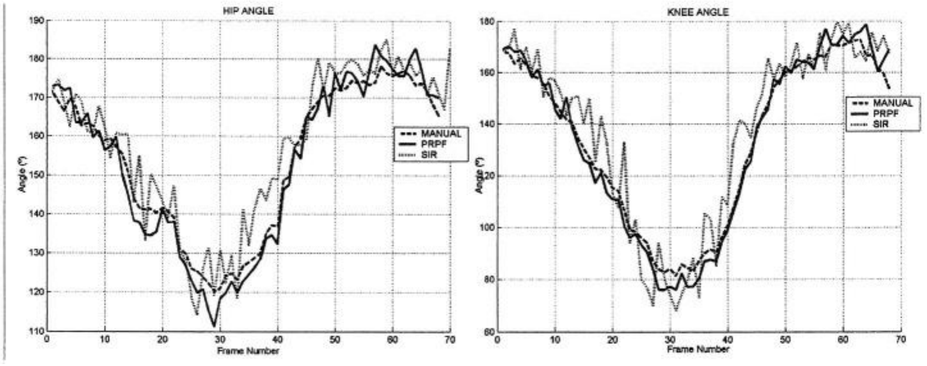


Fig. 9. Right hip (left) and knee (right) angle estimation using PRPF, SIR and manual digitizing

Table 2. *MSE / frame* values with respect to manual digitizing and N_{part} */ frame* of SIR and PRPF for two different motion sequences

SEQUENCES		SIR	PRPF
JUMP (figure 8)	N_{part} / frame	1600	1363
	MSE Knee Angle / frame	124,4	20,6
	MSE Hip Angle / frame	45,9	15,4
FRONTAL MOVEMENT (figure 5)	N_{part} / frame	4000	2838
	MSE R Elbow Angle / frame	1220,4	71,7
	MSE L Elbow Angle / frame	6912,4	194,2

7 Conclusion

The main contribution of this work is the application of the Path Relinking Particle Filter (PRPF) algorithm to model-based human motion tracking and analysis. PRPF was originally developed for general estimation problems in sequential processes that are represented by the state-space model abstraction. Experimental results have shown that this model-based PRPF framework can be very efficiently applied to the 2D human pose estimation problem, as demonstrated by Table 2. The proposed geometrical human model is flexible and has been designed to be easily adapted to the different analyzed human motion activities. In this way, quite energetic activities such as running and jumping in different environment have been easily and effectively tracked. Managing partial occlusions in different human tracking problems with the PRPF approach will be improved. We also propose the study of PRPF in 3D scenarios for human pose estimation in biomechanics applications. Finally, the authors are working in a robust colour and edge based PRPF applied to tracking articulated objects.

References

[1] Wang, L., Weiming, H., Tieniu, T.: Recent developments in human motion analysis. Pattern Recognition 36 (2003) 585–601
[2] Moeslund, B., Granum, E.: A Survey on Computer Vision-Based Human Motion Capture. Computer Vision and Image Understanding 81 (2001) 231–168
[3] Gavrila, D: The visual analysis of human movement: a review. CVIU 73 1: (1999)
[4] Kakadiaris, I., Sharma, R.: Editorial Introduction to the special issue on human modelling, analysis and syntesis. Machine Vision and Applications Vol. 14 (2003) 197–198
[5] IBV: Biomechanics. http://www.ibv.org/ingles/ibv/index2.html (1992)
[6] Rohr, K.: Human movement analysis based on explicit motion models. Chapter 8 in Motion-based Recognition, Kluwer Academic Publishers (1997) 171–198
[7] Ju, S., Black, M, Yaccob, Y.,: Cardboard people: a parameterized model of articulated image motion. IEEE Int. Conf. on Automatic Face and Gesture Recognition (1996) 38–44
[8] Wachter, S., Nagel, H.-H.: Tracking persons in monocular image sequences. Computer Vision Image Understanding 74 Vol 3 (1999) 174–192
[9] Leung, M.K., Yang, Y.H.: First sight: a human body outline labeling system, IEEE Trans. Pattern Anal. Mach. Intell. 17 (4) (1995) 359–377.
[10] Zotkin, D., Duraiswami, R., Davis, L.: Joint Audio-Visual Tracking Using Particle Filters. EURASIP Journal on Applied Signal Processing, Vol, 11 (2002) 1154–1164
[11] Deutscher, J., Blake, A., Reid, I.: Articulated body motion capture by annealed particle filtering. IEEE Conf. Computer Vision and Pattern Recognition, Vol. 2 (2000) 126–133
[12] MacCormick, J.; Blake, A.: Partitioned sampling, articulated objects and interface-quality hand tracking. Proc European Conf. Computer Vision (2000)
[13] Pantrigo, J.J., Sánchez, A., Gianikellis, K., Duarte, A.: Path Relinking Particle Filter for Human Body Pose Estimation. Accepted paper to appear in LNCS (2004)
[14] Glover, F., Laguna, M., Martí, R.: Scatter Search and Path Relinking: Foundations and Advances Designs. To appear in New Optimization techniques in Engineering (2003)
[15] Glover, F.: A Template for Scatter Search and Path Relinking. LNCS, 1363, (1997) 1-53
[16] Arulampalam, M., et al.: A Tutorial on Particle Filter for Online Nonlinear/Non-Gaussian Bayesian Tracking. IEEE Trans. On Signal Processing, V 50 (2): 174–188 (2002)

Multiple Facial Feature Tracking Using Multi-cue Based Prediction Model

Congyong Su[1], Hong Zhou[2], and Li Huang[1]

[1] College of Computer Science, Zhejiang University
Hangzhou 310027, China
{su,lihuang}@cs.zju.edu.cn

[2] Department of Instrumentation Science and Engineering, Zhejiang University
Hangzhou 310027, China
zhouh@mail.bme.zju.edu.cn

Abstract. It is challenging to track multiple facial features simultaneously in video while rich facial expressions are presented in a human face. To accurately predict the positions of multiple facial features' contours is important and difficult. This paper proposes a multi-cue prediction model based tracking algorithm. In the prediction model, CAMSHIFT is used to track the face in video in advance, and facial features' spatial constraint is utilized to roughly obtain the positions of facial features. Second order autoregressive process (ARP) based dynamic model is combined with Bayesian network based dynamic model. Incorporating ARP's quickness into graphical model's accurateness, we obtain the fusion of the prediction. Finally the prediction model and the measurement model are integrated into the framework of Kalman filter. The experimental results show that our algorithm can accurately track multiple facial features with varied facial expressions.

1 Introduction

Tracking for multiple facial features is challenging in computer vision domain, since they are deformable objects and have complex articulated motions. Kass *et al.* [1] proposed Snakes: Active Contour model to track the contour of lips. Snakes are energy-minimizing splines guided by constraints. It can be used to obtain smooth feature contours. But when the total number of control points is large (e.g., dozen), the dimensionality is too high to track contours efficiently. Furthermore, to allow arbitrary variation in positions of control points over time will lead to instability in tracking. Cootes *et al.*[2] proposed the ASM/AAM algorithms, in which tracking is based on face detection and recognition. However the tracking results depend on the model's initial position and the variations contained in the training set, which makes it difficult to deal with occlusions. Furthermore, during the training, broken line is used to mark the facial features, which is not smooth.

We propose a Bayesian network enhanced prediction model based multiple facial feature tracking algorithm. Our considerations for tracking are as follows:

(1) Choose the B-spline to describe feature's contour. B-spline is smooth. It is better than broken line, since the contour of facial feature is smooth too.
(2) Utilize principal component analysis (PCA) to reduce dimensionality for each facial feature. Therefore un-plausible contours are eliminated by subspace method.
(3) Propose a multi-cue based prediction model.
    a) First, use low-level feature based face tracking algorithm – CAMSHIFT [3] to give an estimation of the face's position. Therefore the search space for observation model is narrowed down.
    b) multiple facial features are tracked simultaneously, spatial constraint among facial features is also taken into account.
    c) We learn the second-order auto regressive process (ARP) based dynamic model for each facial features.
    d) We use graphical model – Bayesian network to enhance the ARP based dynamic model. The Bayesian network in this paper combines the influence on a facial feature in the current time instant contributed by multiple facial features in the previous time instant. In this way, it is more robust than tracking each facial feature independently. We integrate all the above prediction models as multi-cues into prediction model of the Kalman filter.
(4) Finally, the prediction and observation model make up the Kalman filter framework in the standard way.

1.1 Related Work

There is much prior work on tracking the facial feature. The ASM/AAM algorithm is a classic method to track multiple facial features, as discussed in the previous section. Kapoor and Picard [4] proposed an infrared camera based tracking algorithm, which didn't need any manual alignment or calibration. However only upper facial features are tracked. Gu *et al.* [5] proposed an active approach to track facial expressions, and they combined the IR sensor with a Kalman filter, however the local graphes they used to capture the spatial relationship do not fit into a probabilistic framework.

Although there are many facial feature tracking algorithms not mentioned here, in this paper, we are specially interested in Bayesian network, a kind of graphical model's application in multiple facial feature tracking.

This paper is organized as follows. Section 2 describes the representation of facial feature's contour. Dimensionality reduction for facial feature is given in Section 3. Section 4 provides the multi-cue based prediction model in detail. Section 5 presents the measurement model. The experimental results are in Section 6.

2 Representation of Facial Feature's Contour

In this paper, we track the contours of facial features, and use B-spline $(x(u,t), y(u,t))$ to represent facial feature's contour in time instant t. Suppose

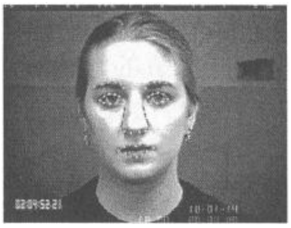


Fig. 1. The contours of facial features are represented by B-spline. Control points are shown as yellow dots, and the B-splines are described by magenta curves

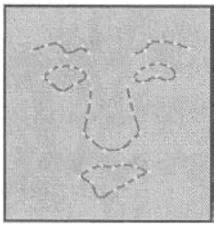


Fig. 2. The results of arbitrarily manipulating spline vectors

there are N spans and N_c control points, we have

$$x(u,t) = B(u)C^x(t), y(u,t) = B(u)C^y(t), 0 \leq u \leq N, \tag{1}$$

where $C^x(t) = [C_1^x(t), ..., C_{N_c}^x(t)]^T$ are the x coordinates for all control points in time instant t, and $C^y(t)$ are the y coordinates. The vector $B(u)$ consists of blending coefficients [6]. The control points of B-spline composite into a spline vector $C(t)$:

$$C(t) = [C^x(t) \;\; C^y(t)]^T. \tag{2}$$

The B-splines that represent facial features are shown in Fig. 1, where the contour of eyebrow is the upper edge of eyebrow; the contour of eye is the boundary between eyelid and eyeball, excluding the eyelid; the contour of nose is the border between nose and the skin of face; the contour of mouth is the edges of upper and lower lips.

If we manipulate the control point's positions arbitrarily, it is easy to generate spline that does not look like facial feature's contour (see Fig. 2). Therefore arbitrarily manipulating the control points may lead to tracking failure.

3 Dimensionality Reduction for Contour of Facial Feature

The movement of facial feature can be decomposed into two parts: rigid motion and no-rigid motion. The rigid motion is caused by the motion of head, while the non-rigid one is the motion of each facial feature that caused by facial expressions. Facial features (eyes, eyebrows, nose, mouth) are generally in the same plane. When rigid motion of head occurs, the contour of facial feature has six degrees of freedom (DOF). For the non-rigid motion of a facial feature, we carry out PCA for the contour of facial feature in the training face image sequence. Let the dimensionality of non-rigid motion is reduced to N_{nr}, and then the total dimensionality of all facial features is $6 + N_{nr}$. Let s_t denote the parameters of state space after the dimensionality reduction, and the spline vector $C(t)$ can be written as

$$C(t) = Ws_t + C_0, \tag{3}$$

where W is a $N_C \times N_S$ shape matrix, N_C denotes the DOF before dimension reduction, and $N_C = 2N_c$. N_S denotes the dimensionality after dimensionality reduction, and $N_S = 6 + N_{nr}$. C_0 is the template of contour, which is usually obtained by manually marking.

4 Multi-Cue Based Prediction Model

In multiple facial feature tracking , we first use ARP based dynamic model to predict the behavior of each facial feature's motion. Furthermore, facial features are related with each other, we use Bayesian network to model the relationship among facial features. Our prediction model combines the quickness of auto regressive process with accuracy of graphical model.

4.1 Second Order Auto Regressive Process Based Prediction Model

The motion of a facial feature's contour can be modelled by a noise driven second-order auto regressive process (ARP), which can be shown as the following second-order linear differential equation:

$$s_t = A_2 s_{t-2} + A_1 s_{t-1} + D_0 + B_0 w_t, \tag{4}$$

where A_1, A_2 and B_0 are matrices, A_1 and A_2 are the deterministic parameters, and B_0 is the stochastic parameter. D_0 denotes a fixed offset, and the distribution of each component of w_t belongs to i.i.d. Gaussian noise. Let $\chi_t = \begin{bmatrix} s_{t-1} \\ s_t \end{bmatrix}$, the Eq. (4) can be written as:

$$\chi_t = A\chi_{t-1} + D + Bw_t, \tag{5}$$

where $A = \begin{bmatrix} 0 & I \\ A_2 & A_1 \end{bmatrix}$, I is an identity matrix, $D = \begin{bmatrix} 0 \\ D_0 \end{bmatrix}$, and $B = \begin{bmatrix} 0 \\ B_0 \end{bmatrix}$. From Eq. (5),we can see that χ_t only depends on χ_{t-1} in the previous time instant. Therefore the dynamic model for one facial feature is actually a Markov chain. But this model doesn't consider the relationship among facial features. Since second-order ARP can describe constant velocity motion, decay and damped oscillation [7], we use it as the plausible dynamic model for each facial feature.

In the second-order ARP dynamic model (see Eq. (5)), the parameters A, D and B are unknown. Although it is possible to specify the parameters empirically, it is more convincible to estimate these parameters from training image sequences. In this paper, we choose a bootstrapping strategy to learn the parameters for dynamic model.

4.2 Using Graphical Model to Model the Relationship Among Facial Features

The dynamic model in Section 4.1 is for one facial feature, and we can build serval dynamic models, each for one different facial feature. Multiple independent ARP

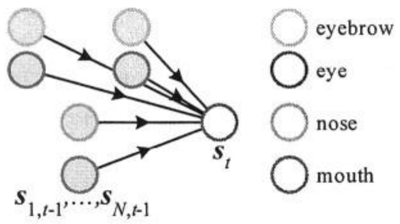


Fig. 3. Bayesian network based dynamic model for multiple facial feature prediction

models based tracking method for multiple facial features is prone to tracking failure, since the inter-relationships among facial features are not taken into account. Actually, the motions of each facial feature relate to each other. For example, when one frowns, his eyes will become smaller; when one surprises with wide open mouth, the eyebrows will move up. It is difficult to describe this kind of interrelationship deterministically. In this paper, we use probabilistic graphical model - Bayesian network to describe it non-parametrically.

Bayesian Network Bayesian network [8, 9, 10] is a directed acyclic graph (DAG). The Bayesian network used in the paper is shown in Fig. 3, where the filled circle denotes observation node, and the empty circle denotes hidden node. The directed edge represents the statistical dependency between two nodes, and the direction is from the parent node to the child node. The intuitive meaning of Fig. 3 is that we can predict the current position of mouth's contour on condition that we have already known the positions of each facial feature's contours in the previous time instant.

Bayesian Network Based Dynamic Model We utilize Bayesian inference to calculate the marginal probability $p(s_{j,t}|\{s_{i,t-1}\}_{i=1}^{N})$. For multiple facial feature tracking, the intuitive meaning is to predict the contour state parameter $s_{j,t}$ in current time instant t on condition that each facial feature's contour state parameters $\{s_{i,t-1}\}_{i=1}^{N}$ are already known. The result of prediction is $\hat{s}_{j,t}$ that maximizes the marginal probability.

$$\hat{s}_{j,t} = \arg\max_{s_{j,t}} p(s_{j,t}|\{s_{i,t-1}\}_{i=1}^{N}) \tag{6}$$

Generally, the Bayesian model based dynamic model can not be decomposed except that $s_{1,t-1},\ldots,s_{i,t-1},\ldots,s_{N,t-1}$ are mutually independent on condition of $s_{j,t}$. But we could use Eq. (7) to approximate the joint marginal probability. Murphy and Weiss proved that it is feasible [9].

$$p(s_{j,t}|\{s_{i,t-1}\}_{i=1}^{N}) = \prod_{i=1}^{N} p(s_{j,t}|s_{i,t-1}) \tag{7}$$

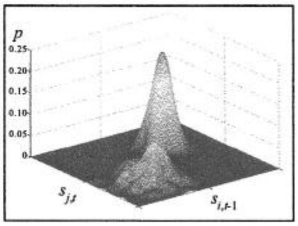


Fig. 4. The sketch map of $p(s_{j,t}, s_{i,t-1})$

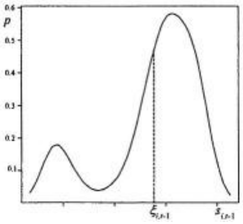


Fig. 5. The sketch map of $p(s_{i,t-1})$

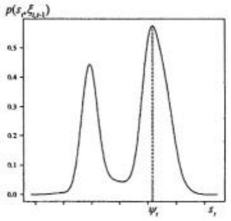


Fig. 6. The sketch map of $p(s_{j,t}, s_{i,t-1})|_{s_{i,t-1}=\xi_{i,t-1}}$

Training Bayesian Network Based Dynamic Model Different from the parametric second-order ARP based dynamic model, Beyesian network based dynamic model $p(s_{j,t}|\{s_{i,t-1}\}_{i=1}^{N})$ is non-parametrical. From Eq. (7), we know that in order to solve the non-parametric dynamic model, the key point is to calculate $p(s_{j,t}|s_{i,t-1})$. From the conditional probability theorem, we have

$$p(s_{j,t}|s_{i,t-1}) = p(s_{j,t}, s_{i,t-1})/p(s_{i,t-1}), \tag{8}$$

where $p(s_{j,t}, s_{i,t-1})$ is joint probability, and $p(s_{i,t-1})$ is the probability of facial feature i in the previous time instant. From the training data, we fit the mixtures of Gaussians to $p(s_{j,t}, s_{i,t-1})$ and $p(s_{i,t-1})$. We can obtain $p(s_{j,t}|s_{i,t-1})$ by evaluating Eq. (8). The sketch maps for $p(s_{j,t}, s_{i,t-1})$ and $p(s_{i,t-1})$ are shown in Fig. 4 and Fig. 5.

Using Bayesian Network Based Dynamic Model to Predict Contour of Feature On condition that the facial feature i's state parameter $s_{i,t-1}$ is equal to $\xi_{i,t-1}$ in the previous time instant, we can predict the state of facial feature j based on Eq. (6) and Eq. (7).

$$\begin{aligned}\hat{s}_{j,t} &= \arg\max_{s_{j,t}} p(s_{j,t}|\{\xi_{i,t-1}\}_{i}^{N}) = \arg\max_{s_{j,t}} \prod_{i=1}^{N} p(s_{j,t}|\xi_{i,t-1}) \\ &= \arg\max_{s_{j,t}} \prod_{i=1}^{N} (p(s_{j,t}, \xi_{i,t-1})/p(\xi_{i,t-1}))\end{aligned} \tag{9}$$

When $s_{i,t-1} = \xi_{i,t-1}$, $p(s_{j,t}, s_{i,t-1})|_{s_{i,t-1}=\xi_{i,t-1}}$ is single variable MOG, its one dimensional sketch map is shown in Fig. 6. From Eq. (9), we know that we need to calculate the maximum of the product of N MOGs. Since it is difficult to obtain the maximum directly, practically approximative methods are used, e.g. starting from an arbitrary point, use gradient descent algorithm to obtain the local maximum; utilizing discretization, draw *ns* samples, then find the maximum probability of them. We tend to use the latter method, since the global maximum can be obtained.

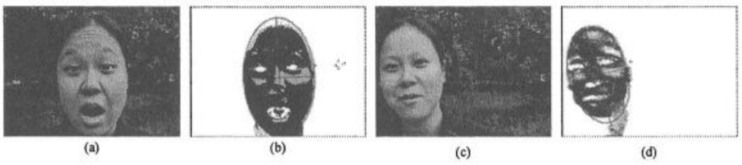


Fig. 7. CAMSHIFT algorithm can be used to obtain the face's position and orientation. (a) one frame contains the frontal face; (b) the probabilistic distribution map of face, the tracked face area is shown in the blue ellipse; (c) one frame contains the face with orientation; (d) the probabilistic distribution map of face, the blue across approximately gives the face's orientation

4.3 Low-Level Feature Based Face Tracking in Advance

The second order ARP based dynamic model is used for each facial feature, not for the whole face. To avoid the tracked facial feature's contour drifting out of the face, it is necessary to track the whole face firstly. Therefore we could narrow down the search range for the facial feature tracking.

In this paper, we use color histogram based CAMSHIFT algorithm [3] to track the whole face, and obtain the location of human face (see Fig. 7(a) and 7(b)). By this means, we set a search range for the observation model of facial feature tracking. Since observation is the most time-consuming part of the facial feature tracking, narrowing down the search range make the tracking more efficient. CAMSHIFT tracking algorithm can also be used to obtain the orientation of face (see Fig. 7(c) and 7(d)), and this make preparation for spatial constraint in the following section.

In the experiments, the training for dynamic model was carried out for frontal faces. When the tracked face has orientation, we use CAMSHIFT algorithm to approximately get its orientation, and then warp the face image into the canonical position. We do the facial feature tracking on the canonical face, we undo the warping after tracking for graphical display.

4.4 Spatial Constraint Among Facial Features

Human face image belongs to a special class. The facial feature position of different person only varies in a local area. If we know the position and orientation of a face, we can use the spatial constraint among facial features to obtain the approximate position of each facial feature. As shown in Fig. 8(a), the human face can be described by an ellipse. The ratio between length and width is roughly 7 : 5, and the distance between two eye's centers is about 2/5 of the width of face. For face with orientation, this spatial constraint still holds (see Fig. 8(b)).

The face position obtained by CAMSHIFT combined with spatial constraint is just an estimation for each facial feature's position. In the previous section, we include rigid motion in second order ARP model to further distill the residue rigid motion for each facial feature.

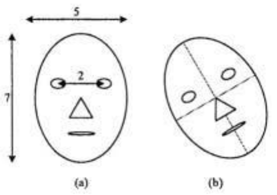


Fig. 8. (a) The spatial constraint among facial features; (b) The spatial constraint also holds for face with orientation

4.5 Multi-Cue Fusion for Prediction

The spatial constraint among facial features can be combined with face tracking to specify the approximate position of facial features in the current time instant. This kind of low-level prediction can be integrated with dynamic model based prediction to improve the accuracy of prediction. The low-level CAMSHIFT algorithm based face tracking and spatial constraint among facial features are the preprocessing for prediction, and they can be easily fused into the prediction model.

Integrate Second-Order ARP Based Dynamic Model with Graphical Model Based One Second-order ARP based dynamic model is very quick to prediction, but it ignores the influence on the facial feature's position in the current time instant caused by the position of other facial features in the previous time instant. The graphical model based prediction can obtain better result than ARP based method theoretically, but its non-parametric property determines that finding the global maximum is time consuming. This paper combines the advantages of these two method. The procedure of the algorithm is as follows:

Step 1. Step1. Based on Eq. (4), we use reject sampling [11] method to draw *ns* samples (we choose *ns* = 20 based on experiments) from w_t. By this way, *ns* ARP based prediction results $s^k_{j,t}$ are generated, where $0 < k < ns$.
Step 2. Step2. Substitute $s^k_{j,t}$ for $s_{j,t}$ in Eq. (9), we can find the best prediction $\hat{s}_{j,t}$ from the *ns* predictions.
Step 3. Step3. Based on Eq. (3), we can solve the contour $\hat{C}(t)$ of current facial feature in time instant t.

In the ASM/ASM based multiple facial tracking algorithms, their dynamic models are only zero-order or first-order linear model, which can only describe uniform motion or uniform acceleration/deceleration motion. Therefore, the prediction based on these dynamic model is not enough. These tracking algorithms usually converge to correct position only when the initial position of facial feature's contour is reasonably appropriate. If the initial position is not very good, the tracker tends to be locked on a local maximum or fails.

5 Measurement Model

After we have the prediction result of the facial feature's contour position, the result should be verified and adjusted by a measurement model. Compared with the prediction model, the measurement model is relatively easy to construct.

On condition that the predicted contour of a facial feature is $\hat{C}(t)$, one measurement in time instant t is to find feature (e.g. edge) along the normal vector $n(pt)$ of one point pt on the contour curve:

$$f(pt,t) = (C(t) - \hat{C}(t)) \cdot n(pt) + g(pt,t), \tag{10}$$

where $g(pt,t)$ is an i.i.d. Gaussian noise, and its variance δ^2 is a constant. The visual effect of measurement along the normal vector is to pull the contour curve $\hat{C}(t)$ along the normal direction based on feature found. Let $\tilde{C}(t)$ denote the contour curve after measurement, and it is the final tracking result in time instant t for current facial feature.

Already have the prediction and measurement model separately, they can be integrated into the Kalman filter framework in a standard manner.

6 Experimental Results

We have implemented a prototype system $\mathrm{MF^2T}$ by Visual C++ and Matlab on Windows platform. $\mathrm{MF^2T}$ tracks 6 contours of facial features, which are left eyebrow (right eyebrow), left eye (right eye), nose and mouth. The contours are all quadratic B-spline, where eyebrow and nose are open B-spline.Other facial features are described by closed B-spline. The number of control points for eyebrow, eye, nose and mouth is 10, 9, 16, 12 respectively (see Fig. 1). The total number of control points is 66, i.e., the total dimensionality is 132.

We choose Cohn-Kanade facial expression database as the training set [12], since it contains a lot of fontal expressive face images, and is stored as 30fps image sequence. The dimensionality reduction and training for prediction model are carried out on this database.

In order to reduce the dimensionality for contour model, we select 100 frame frontal face images from the training set, and these images belong to 48 different persons. We do PCA for each facial feature. After dimensionality reduction, the dimensionality for eyebrow, eye, nose and mouth is 7, 7, 12 and 9 respectively. The total dimensionality is 49, accounting for 99% variations.

In the training for second-order ARP based dynamic model, we obtain 6 dynamic models from image sequences, each for eyebrows (left and right), eyes (left and right), nose and mouth. In the training, we use interactive editing to manually mark feature points in order to get the ground truth. In the training of Bayesian network based non-parametric dynamic model, we use the same image sequence as the second-order ARP model. For image sequence with M frames, there are $(M-1)C_6^2 = 15(M-1)$ pairs of training data. In other words, there are 15 kinds of data for joint probability $p(s_{j,t}, s_{i,t-1})$, and we fit 8 cluster mixtures of Gaussians to them. For the state $p(s_{i,t-1})$ in the previous time instant, there

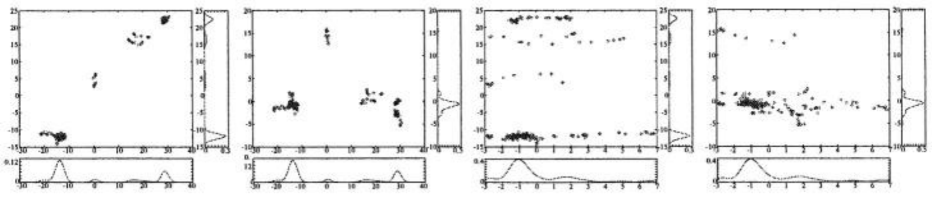


Fig. 9. Joint probabilistic density of 4 pairs of PCA coefficients. The corresponding marginal distribution is shown on each figure's right and lower part. We can see that the relationship between PCA coefficients is multimodal and non-Gaussian

are $C_6^1 = 6$ kinds of data, we also fit 8 cluster mixtures of Gaussians to them. We can calculate conditional probability $p(s_{j,t}|s_{i,t-1})$ from the fitted probabilities.

The reason to use mixture of Gaussian is that the relationship between contour of a facial feature in the previous instant and that in the current time instant is multimodal, and is not Gaussian. We make experiments to prove this. We choose 3 most important PCA coefficients for left eye in the previous time instant, and 3 for mouth in the current time instant. Therefore we obtain 9 kinds of PCA coefficient pairs, and each kind of PCA coefficients' number is $M - 1$. The spatial distribution for 4 different kinds of PCA coefficient pairs is shown in Fig. 9, and we can see that simple Gaussian approximations would obscure this data set's meaningful information.

In the experiments, it turns up that facial expressions change very fast, e.g. it only needs 10 frames to change expressions from neutral to happy (for 30fps video); therefore there are relatively large motion in adjacent two frames. We carry out two kinds of experiments: (1) Tracking multiple facial features in fontal expressive face images in Cohn-Kanade database (640×490, 30fps) (see Fig. 10(2) We use digital video camera to capture face image sequence (640 × 480, 30fps) with expression, orientation and occlusion in the outside. We track these image sequences (see Fig. 11 and Fig. 12, zoom in to see clearly). All the tracked image sequences are not included in the training set. We compare our algorithm's result with that of AAM's.

The tracking result for a surprise expression sequence is shown in Fig. 10. We can see from the result of edge detection (see Fig. 10(c)) that, when mouth is wide open, the teeth and tongue form dense cluster for the contour of mouth. AAM is locked on local maximum, since it treats the contour of teeth as that of lower lip, and regards the dark circles as contour of eyebrows. Our algorithm correctly predicts that the mouth will probably open when the eyebrows are rising and the eyes are opening. The original size of image sequence in Fig. 10 is 640 × 490 pixels, and the tracking is carried out in that size. However for the purpose of display in this paper, we crop the image down to the size of 432 × 490. The frame number is shown in the time code at the bottom of the image.

Our algorithm also can robustly track multiple facial features when face has orientation and size variation (zoom in to see Fig. 11(b)). Since we use CAMSHIFT algorithm to get the position of face in advance, we avoid the AAM

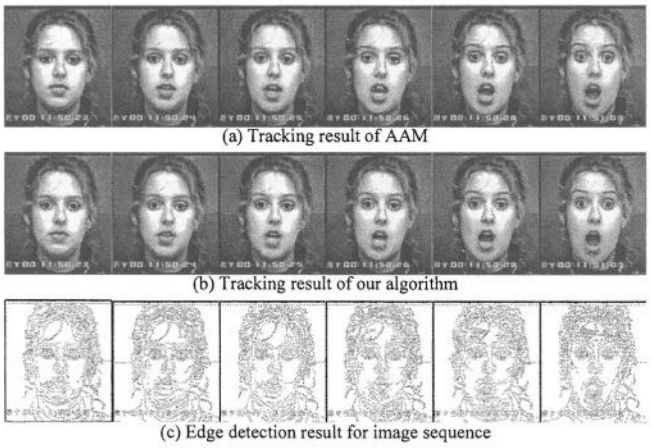

(a) Tracking result of AAM
(b) Tracking result of our algorithm
(c) Edge detection result for image sequence

Fig. 10. Tracking results for a surprise expression sequence. (a) Tracking result of AAM. (b) Tracking result of our algorithm. (c) Edge detection result for image sequence

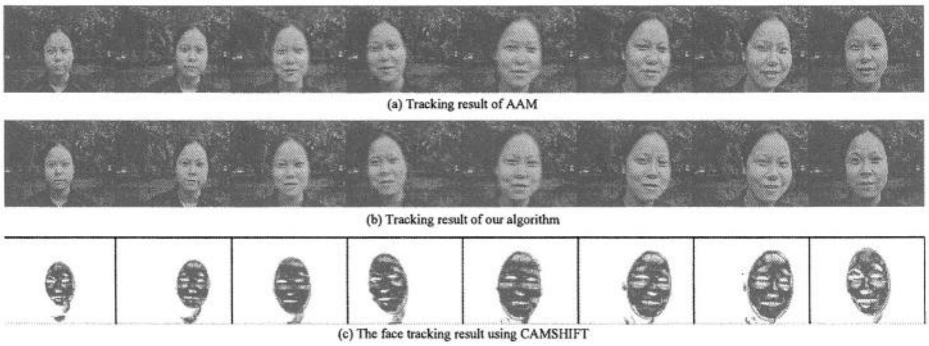

(a) Tracking result of AAM
(b) Tracking result of our algorithm
(c) The face tracking result using CAMSHIFT

Fig. 11. Comparison of tracking results: from far to near, quickly approaching the camera, and with head orientation and face expression. (a) Tracking result of AAM. (b) Tracking result of our algorithm. (c) The face tracking result using CAMSHIFT

tracker's problem that left eyebrow is out of the face (zoom in to see Fig. 11(a)). Furthermore, in the graphical model based prediction, we consider the spatial constraint of facial features, the problem that contour of upper lip overlaps with that of nose is also avoided. In Fig. 11, the frame numbers are 9, 22, 37, 42, 62, 66, 70 and 157.

For the classic AAM algorithm, since occlusions occur in various forms (e.g. occlusions on different facial feature), it is difficult to integrate such negative samples into the training set; therefore the training set for AAM only contains faces without occlusions. It is difficult for AAM to deal with occlusions on facial features in image sequence, and AMM tracker usually fails in such situation (zoom in to see Fig. 12(a)). However our algorithm utilizes the natural rela-

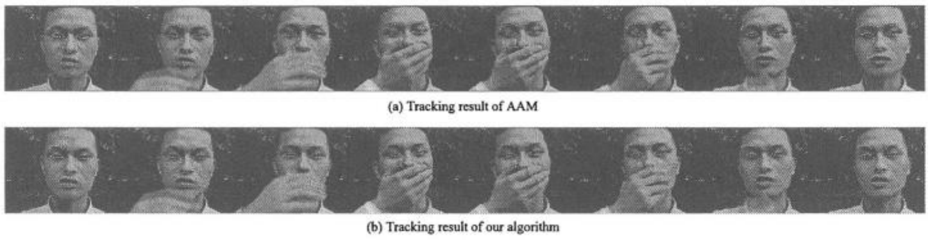

(a) Tracking result of AAM
(b) Tracking result of our algorithm

Fig. 12. Comparison of tracking results with occlusions: hiding mouth by hand. (a) Tracking result of AAM. (b) Tracking result of our algorithm

tionship among facial features, the contour of the occluded facial feature can be inferred by Bayesian network learning (zoom in to see Fig. 12(b)). The comparison results are shown in Fig. 12, where the frame numbers are 0, 1, 4, 23, 51, 54, and 55.

Our algorithm runs at 3 frames per second on a Pentium4 1.8G computer.

7 Conclusions

In this paper, we propose a Bayesian network enhanced prediction model based multiple facial feature tracking algorithm. We combine the second-order ARP based dynamic model with graphical model - Bayesian network based one, and obtain quick and accurate multi-cue based prediction model. The prediction and measurement model are integrated into the Kalman filter framework in a standard way. The experimental results show that our algorithm is effective.

References

[1] Kass, M., Witkin, A., Terzopoulos, D.: Snakes: active contour models. IJCV (1987)

[2] Cootes, T.F., Edwards, G.J., Taylor, C.J.: Active appearance models. IEEE Trans. PAMI (2001)

[3] Bradski, G. R.: Real time face and object tracking as a component of a perceptual user interface. In: Proc. WACV. (1998) 214–219

[4] Kapoor, A., Picard, R. W.: Real-time fully automatic upper facial feature tracking. In: Proc. FG. (2002) 8–13

[5] Gu, H., Ji, Q., Zhu, Z.: Active facial tracking for fatigue detection. In: Proc. WACV. (2002) 137–142

[6] de Boor, C.: A practical guide to splines. Springer-Verlag (1978)

[7] Blake, A., Isard, M.: 3d position, attitude and shape input using video tracking of hands and lips. In: Proc. Siggraph. (1994) 185–192

[8] Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann (1988)

[9] Murphy, K., Weiss, Y.: The factored frontier algorithm for approximate inference in dbns. In: Proc. UAI. (2001) 378–385

[10] Yedidia, J. S., Freeman, W. T.: Understanding belief propagation and its generalizations. In: Proc. IJCAI. (2001)
[11] Press, W., Teukolsky, S. A., Vetterling, W. T., et al.: Numerical recipes in C, 2nd edition. Cambridge University Press (1992)
[12] Kanade, T., Cohn, J., Tian, Y. L.: Comprehensive database for facial expression analysis. In: Proc. FG. (2000) 46–53

A System for Choreographic Simulation of Ballet Using a 3D Motion Archive on the Web

Asako Soga[1], Bin Umino[2], Takami Yasuda[3], and Shigeki Yokoi[3]

[1] Faculty of Science and Technology, Ryukoku University
1-5, Yokotani, Oe-cho, Seta, Otsu, 520-2194, Japan
asako@rins.ryukoku.ac.jp
[2] Faculty of Sociology, Toyo University
umino@toyonet.toyo.ac.jp
[3] Graduate School of Information Science, Nagoya University
{yasuda,yokoi}@info.human.nagoya-u.ac.jp

Abstract. We have developed a system that supports the creation of classical ballet choreography in 3D environments. The system allows ballet teachers to interactively create various choreographies on the Web and to represent them as 3DCG animation. Our research approach focuses on sharing 3D animation data over the Web and reusing motion data effectively. We discuss the standardization of motion data to share them in the Web environment and the creation method of 3DCG animation to simulate choreographies. To develop such a Web-based choreography simulation system, we suggest a method for composing and archiving 3DCG animation on the Web. This allows us to simulate choreography interactively on the Web.

1 Introduction

As 3-dimensional (3D) human animation through motion capture systems is further integrated into various media types, such as movies or TV, the demand for such animation is expected to increase. Our research approach focuses on sharing 3D animation data over the Web and reusing motion data effectively. Sharing motion data on the Internet allows anyone to access various artistic dances all over the world. In addition, these motion data can be applied in online systems for educational and artistic purposes. For example, one can prepare 3D movement catalog, compose pieces by collaborating via the Net, and exhibit pieces on an electronic dance museum.

Our goal is to develop and integrate several modules into a system capable of animating realistic virtual ballet performances on the Web. As a case study of Web3D technology and its application, we have developed a system that supports the creation of classical ballet choreography in 3D environments. In this paper, we discuss our new system, which is called "Web3D Dance Composer."

For many years, various animations have composed by using applications for choreography [1, 2]. Such applications, though widely used up until now, do not work on the Web and they are not suitable for the online application.

Powerful tools for scripting complex movements and behaviors have already been created [3, ?], but coming up with artistic human motions on the Web still remains a challenge.

In the next section, we begin by explaining the concept of our proposed system. In section 3, we explain the standardization method for human motion description. In section 4, we suggest a Web-based choreography simulation system followed by experiments and considerations.

2 Web3D Dance Composer

2.1 Basic Concept

The main goal of our research is to develop a tool for editing classic ballet motions on the Web. Our system offers users a set of powerful functions to direct highly realistic virtual dances. In this work, we focus on a number of issues related to composing and producing various classic ballet stages:

– A Web-based and platform-independent system for high scalability based on VRML and Java applet;
– Standardization of human motion description methods based on H-Anim [6]; and
– Independent motion data sequences that make it easy to access and recompose steps.

One of the contributions of this work is the standardization of human motion description methods. The main program with sharing and reusing dance motion data over the Net is accurately describing human motion. For this purpose, we adapt H-Anim to a human model of a female ballet dancer. We suggest using our original standardization methods.

Another contribution is registering and reusing choreographies. Our approach is registering created choreographies and sharing them on the Web, thus users can refer to archived choreographies and reuse one of them, because it is difficult for beginners to create ballet choreographies, this system is useful for students. It is also helpful for teachers because there is no textbook for classic ballet lessons, and teachers usually create the choreography each time. By using this system, however, they can download and refer to the choreographies made by other teachers or choreographers. Fig. 1 shows an image of sharing motion data.

2.2 System Overview

Web3D Dance Composer (WDC) is a Web-based interactive choreography simulation system for ballet. Using our proposed system, one can easily compose and simulate ballet dance using the motion data captured from a professional dancer. Since classical ballet is a dance that typically consists of established steps, one can compose choreographies by connecting basic steps.

This system allows ballet teachers and students to create various choreographies for lessons in a short time. The creatable choreographies are “allegro for beginners” only, and for lower-body movement only at the present time.

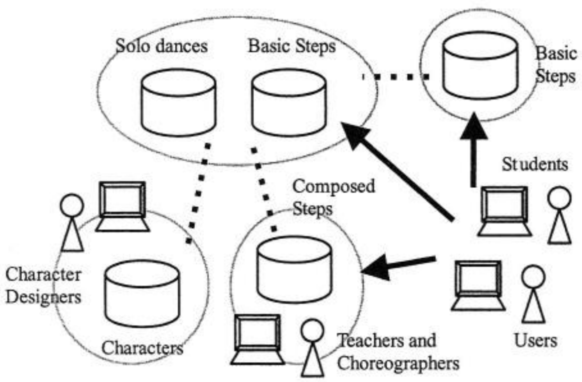


Fig. 1. An image of sharing motion data

3 Standardization of Motion Data

3.1 Recording Motion Data

We constructed a motion archive of basic steps, which were performed by professional ballet dancers. To capture motion data, we used both an optical motion capture system and a magnetic system for six dancers because it is difficult to use the same systems and dancers to gather all steps. We use the EVa System [7] as an optical system, and Motion Star Wireless [8] as a magnetic system.

3.2 Standardization Based on H-Anim

There are various formats for motion capture data, none of which are standardized. This is one of the major problems with sharing and reusing the motion data. To deal with this problem, most CG applications have a tool to adapt it to the final human figure structures. However, using these tools every time is not efficient. In order to solve this problem and to share motion data on the Web, we adopted H-Anim, which is a standard way of representing humanoids on the Web. It defines all names of Joints, Segments, Sites of the human body, and associated hierarchies. It also suggests starting poses for a humanoid animation and a description method for the joint angles.

In this research, we suggest four topics for standardization methods with respect to H-Anim: (a) a hierarchical structure for describing ballet; (b) a description method of joint angles; (c) a body figure of dancers for simulating ballet choreography by 3DCG; and (d) a description method for movement data. Methods (c) and (d) are our original suggestions; (a) also contains an original part, but (b) is H-Anim itself. From the next section, we explain these standardization methods. To edit motion capture data, we use EVa5.20 [9] for the optical system and FiLMBOX3.0 [10] for the magnetic system.

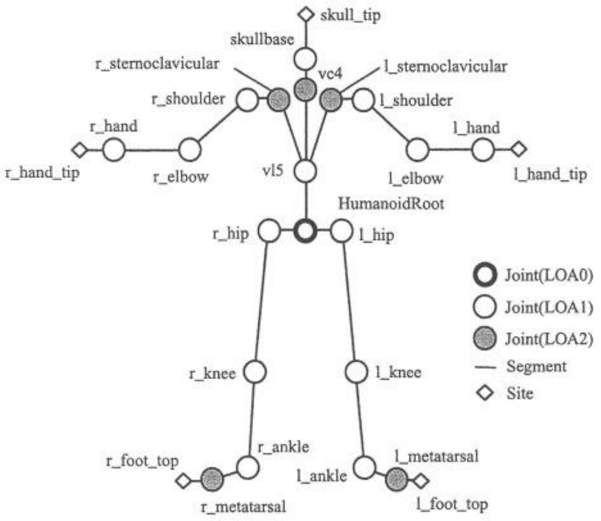


Fig. 2. The human body structure of the standard ballet dancer

3.3 Standardization of Body Structure

In H-Anim, there are 89 defined human joint positions. However, the data from the motion capture system only includes 15-20 of them. In addition, each joint has a different name for different motion capture systems. The names for joints should all be adapted to those of H-Anim. In this process, we prepare “a standard model for a ballet dancer” as a structure for describing ballet. In this model, we define 20 joints that are based on LOA1 (Level of Architecture 1) of H-Anim, and we add the neck, shoulders, and toe joints to it. Fig. 2 shows the human body structure, Fig. 3 is the human body hierarchy of “a standard model for a ballet dancer.”

3.4 Standardization of Description Method for Joint Angles

To share motion data, we unified the starting direction of rotation data. In the case of H-Anim, the default position of the humanoid is defined, where all of the joint angles should be zero. On the other hand, in the case of motion capture data it depends on the system. For example, in the case of the optical system EVa, the starting direction of rotation data is defined as the vector from parent joint to the child joint (Fig. 4(b)). In our research, the joint angles from motion capture data were transformed in the same way to the H-Anim (Fig. 4(a)).

3.5 Standardization of Body Figure for Dancers

Handling data from different motion capture systems, we standardized a human body figure and suggested a CG character for “a standard model for a ballet

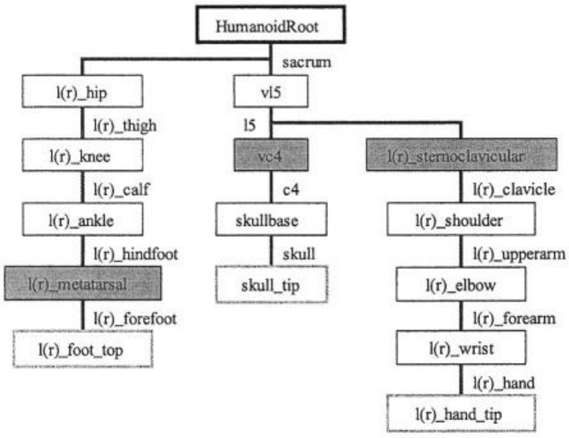


Fig. 3. The human body hierarchy of the standard ballet dancer

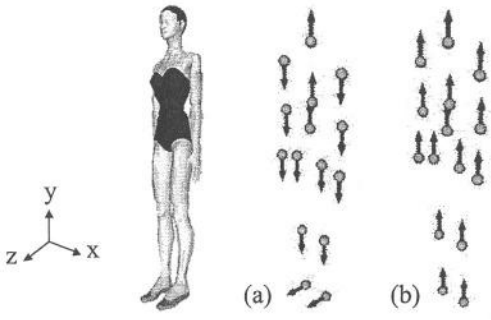


Fig. 4. The initial direction for rotation. (a)H-Anim, (b)EVa System

dancer." This is based on the average of three professional dancers who have similar body figures. We measured their bodies and surmised the joints, and all motion data were adapted to this model. Based on the joints, we produced a 3DCG character for ballet. This character is female, 160 cm tall, and weights 43 kg. She wears a leotard and toe shoes. Fig. 5 shows the appearance of the joints and the 3DCG character, while Table.1 gives the positions of Joint and Site for a standard female ballet dancer.

3.6 Standardization of Description Method for Movement Data

To facilitate simulation and editing of ballet motion, we suggest standardizing the description method for the movement data. In the case of ballet movement, essential attribution is almost always included in the vertical movement, while horizontal movement represents global movement. Moreover, horizontal movement can be modified in order to connect to the next step. Therefore, we segment movement data into vertical and horizontal movements.

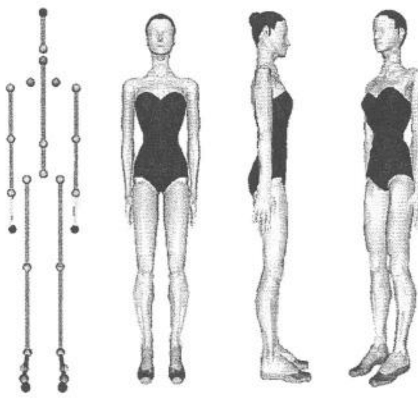


Fig. 5. A 3DCG character for the standard female ballet dancer

4 A Web-Based Choreography Simulation System

4.1 System Architecture and Requirements

Fig. 6 illustrates the system architecture of WDC, which consists of a virtual environment based on VRML and a Java applet running on the WWW browser. We use the External Authoring Interface (EAI) to communicate with the VRML environment and the Java applet, and employ the CGI (Common Gateway Interface) to access the file on the server for archiving choreographies.

The only requirements needed to run this system are a WWW browser and a VRML Plug-in. The motion data of each body part should be prepared on the motion archive on the same server as the Java applet. Other object files, such as background and body shape, may be on the WWW, and users can include more.

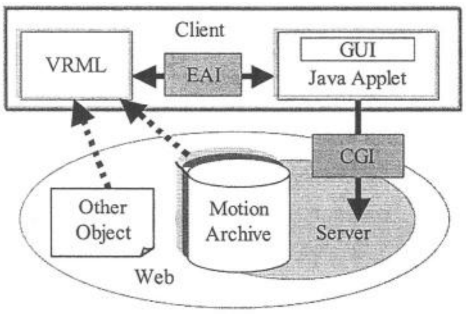


Fig. 6. The system architecture of Web3D Dance Composer

Table 1. The positions of Joint and Site for the standard female ballet dancer

Joint Name	X	Y	Z
HumanoidRoot	0.00	0.89	-0.07
vl5	0.00	1.03	-0.07
skullbase	0.00	1.46	-0.01
l(r)_shoulder	(-)0.15	1.29	-0.03
l(r)_elbow	(-)0.15	1.06	-0.03
l(r)_wrist	(-)0.15	0.82	-0.03
l(r)_hip	(-)0.08	0.86	0.00
l(r)_knee	(-)0.08	0.46	0.00
l(r)_ankle	(-)0.08	0.07	-0.02
vc4	0.000	1.37	-0.02
l(r)_sternoclavicular	(-)0.07	1.31	-0.03
l(r)_metatarsal	(-)0.09	0.01	0.10
Site Name	**X**	**Y**	**Z**
skull_tip	0.00	1.60	0.02
l(r)_hand_tip	(-)0.14	0.65	-0.03
l(r)_foot_top	(-)0.07	0.00	0.17

4.2 Composing Animation

To simulate choreography in real time on the Net, we have developed a Web-based method for composing animation. In the case of using traditional animation software, users should import motion data files and edit part of these to create choreographies. If these systems are applied on the Internet, the process takes a long time and makes many tasks for the network because it is necessary to add and edit a huge amount of animation data.

Our approach to solving this problem is to use EAI and the file reference function of VRML. EAI is an interface for communicating with a Java applet and a VRML, allowing us to access the VRML environment from the Java applet and to edit the VRML description interactively. In addition to this, a VRML scene can import other existing VRML files on the Internet. By using this file reference function, we can easily change the VRML object, such as animation and background, by changing the address of the object file.

Therefore, when adding motion data to the VRML scene, the address of a VRML file can be added instead of animation data itself. Each animation data should be archived on the Web and it can be applied for a 3D environment downloaded by VRML Browser. This permits us to simulate choreography in almost real-time on the Web.

4.3 Archiving Motion Data

We use the browser to download VRML files automatically. This method allows us to adapt animation data to the 3D environment; however, this method does not allow us to edit animation data. Because the animation data of each joint is described in rotation data, these can be used directly. On the other hand, movement data should be edited so as to connect to the following movement.

To solve this problem, we prepared three kinds of files corresponding to the basic steps in the motion archive. One of them is the animation file of each step, which is prepared as a VRML file so as to execute directly after it is loaded. Another is the movement that represents the translation of the X-Z plane, and this is prepared as a text file because it should contain values that can be modified in order to connect the next step. The other is adding information such as step name, starting pose, which is not needed for accumulating animation, but are used for supporting creation.

The animation data is segmented into each step and archived. To archive motion data of basic steps, we edited them as follows. Each step should start from the origin and the front direction. Furthermore, we define default beat for each step. This beat is not related to recording speed because the playing speed is generally decided by beats in a ballet lesson.

4.4 Registering and Reusing Choreographies

Created choreographies by WDC can be registered to the motion archive on the Web. In the present system, 20 composed choreographies for a beginner's lesson that have been choreographed by a ballet teacher are already registered. Previously captured solo dances in classical ballet and characters for the human model that based on VRML H-Anim are also registered. Users can edit these registered choreographies as well as just appreciating them.

By using these registered sequences, users can create a dance performance according to a scenario. Fig. 7 shows such an example of a scenario for a dance performance. The user can post the composed steps to the timeline for each actor, and he/she can simulate different animations for multiple humans. Fig. 8 shows an example of applying different animations for multiple humans corresponding to Fig. 7.

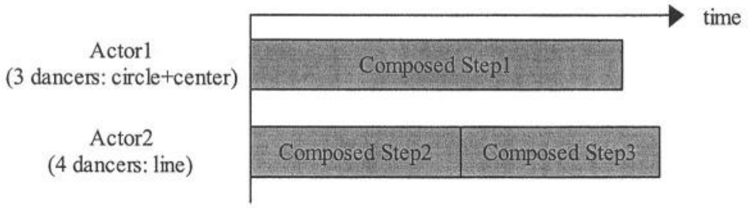


Fig. 7. An example of a scenario

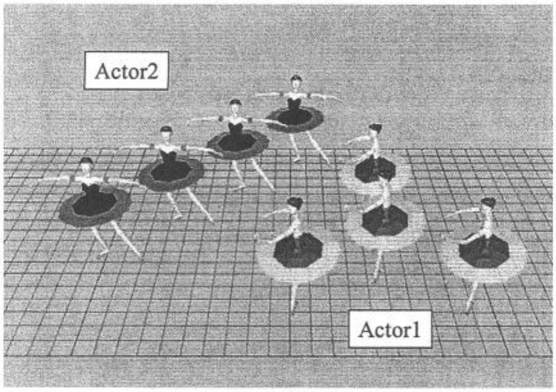


Fig. 8. An example of applying different animations for multiple humans

4.5 GUI Design and Functions of WDC

Fig. 9 shows the User Interface of WDC. It employs only one window, which consists of a virtual environment based on VRML and a Java Applet, including menus for the composing system and the display control system. These menus consist of five panels: a Motion Catalog Panel, a Play Control Panel, a Timeline Panel, an Archiving and Reusing Panel, and a Display Control Panel. All names of steps used for “allegro for biggers” are listed in the Motion Catalog Panel. Composed steps are shown in the Timeline Panel when control buttons are used.

At the beginning, there is a character on the 3D world but no animation. To initiate choreography, the user provides a sequence of steps in Timeline by selecting steps from the Motion Catalog Panel. After selecting a list of steps, one can load the corresponding animation into the character. Once the user loads the animation, he/she can play it anytime. For connecting some steps, each action can be translated and interpolated. The dancer’s character, formation and background can also be changed through the Display Control Panel.

5 Experiments and Discussion

5.1 Experiments

We conducted an experiment on our proposed simulation system to evaluate its utility.

First, a ballet teacher created ten short choreographies from 27 steps that are used in actual lessons for beginners. Second, we recorded as many motion data for these step patterns as possible. We will increase the number to about 100 steps if we consider the different patterns such as starting pose and ending pose. The average number of steps used in each choreography was 15.9, the minimum 11, and the maximum number was 24. Third, we composed and recreated 3DCGs of the same choreographies by using the WDC system.

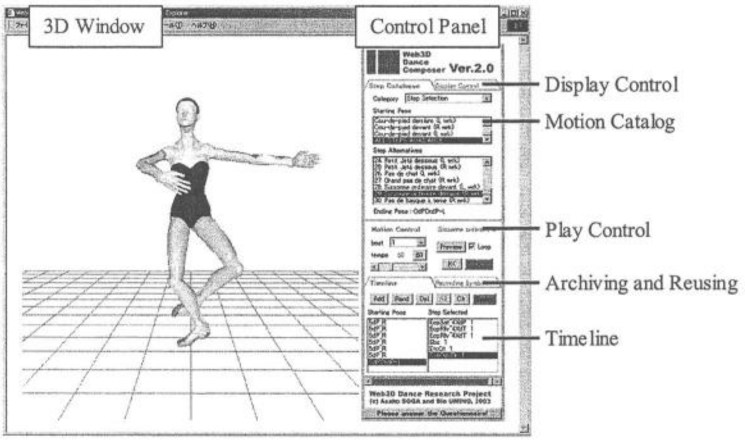


Fig. 9. The user interface of Web3D Dance Composer

Then, ten ballet teachers evaluated these ten choreographies. The average length of teaching experience of these teachers was 9.6 years, with the minimum being 3 years, and maximum 20 years. The criterion for evaluation was whether a 3DCG made by WDC could recreate the original choreography. The evaluation method was as follows. First, a 3DCG made by WDC was shown and teachers remembered the choreography. After that, a movie of the original choreography was shown, then the teachers compared and evaluated these movements. The parts for evaluation were the lower body, the upper body, and rhythm. The evaluation criteria were "recreated exactly," "possible to guess," and "impossible to guess."

Fig. 10 is an example of recreated 3DCG animation choreographed by a teacher.

5.2 Results and Discussion

The evaluation result for the ten choreographies is shown Fig. 11. In the case of the lower body, the sum of "recreated exactly" and "possible to guess" was 65%. Two of the choreographies were strikingly wrong due to a lack of steps. Excluding those two choreographies, the total evaluation late increased to 78%. To increase the percentage, we should record and archive more steps or suggest an effective interpolating method.

Regarding the upper body, the sum of "recreated exactly" and "possible to guess" was 57%. However, we did not choreograph upper body this time, which means the choreographies of the lower body have a relationship with those of the upper body. This is one special characteristic of ballet.

In case of the rhythm, the sum of "recreated exactly" and "possible to guess" was 88%. This means the default beat of each step was correct.

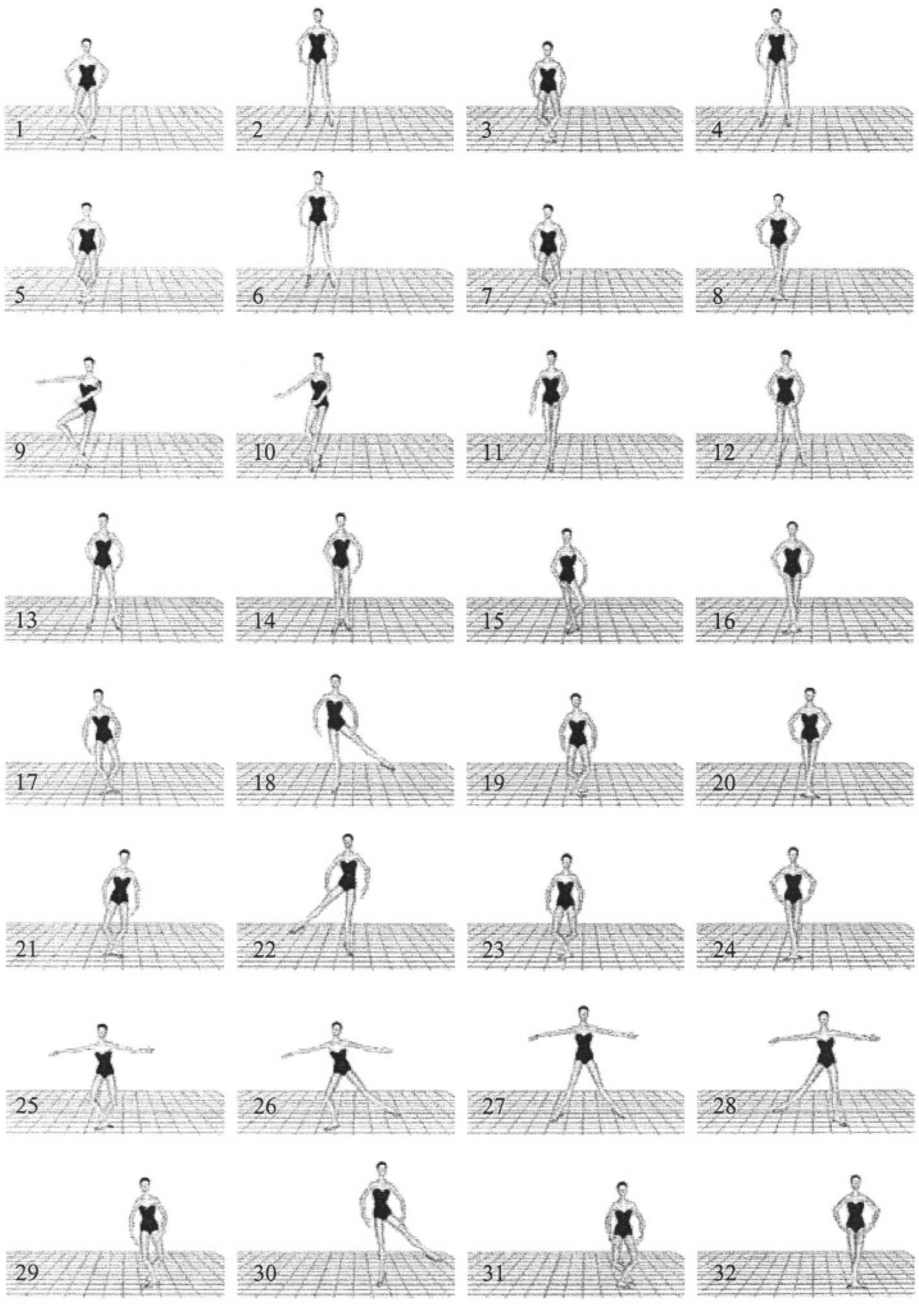


Fig. 10. An example of recreated 3DCG animation choreographed by a teacher

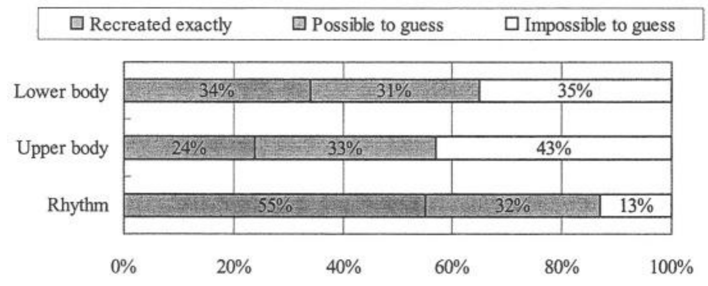


Fig. 11. The evaluation result

6 Conclusion and Future Works

We have developed a system that supports the creation of classical ballet choreography in 3D environments. We suggested the standardization of motion data to share choreography in the Web environment, which provides great opportunities for all ballet dancers and makes available valuable motion data. To develop such a Web-based choreography simulation system, we suggested a method for composing and archiving 3DCG animation on the Web. This allows us to simulate choreography interactively on the Web.

As a result of an evaluation test, we verified that our system could successfully simulate some chains of steps in actual lessons, and we identified some functions requiring further improvement. In future work, we intend to augment the composing system by adding supporting functions such as body-part manipulation. We will also develop an automatic choreography system for actual ballet lessons.

References

[1] Credo Interactive: Life Forms, http://www.credo-interactive.com/
[2] Warabi-za Digital Art Factory, http://www.warabi.or.jp/buyo-fu/
[3] H. Moser and D. Thalmann, “A Rule-Based Interactive Behavioral Animation System for Humanoids,” IEEE Transaction on Visualization and Computer Graphics, Vol.5, No4, pp281-307 (1999)
[4] P. Kalra, N. M. Thalmann, L. Moccozet and G. Sannier, “Real-Time Animation of Realistic Virtual Humans,” IEEE Computer Graphics and Applications, Vol.18, No.5, pp.42-55 (1998)
[5] Sony Creative Products, Paw2 (2001), Japanese, http://www.so-net.ne.jp/paw/
[6] VRML Humanoid Animation Working Group, The VRML Humanoid Animation specification (2001), http://www.h-anim.org/Specifications/H-Anim2001/
[7] Motion Analysis, http://motionanalysis.com/
[8] Innovative Graphics Solutions, http://www.inition.co.uk/
[9] EVa5.20 Reference manual
[10] kaydara, http://www.kaydara.com/

Human Body Analysis with Biomechanics Criteria

J.M. Buades[1], F.J. Perales[1], M. Gonzalez[1], A. Aguiló[2], P. Martinez[2]

[1] Computer Graphics & Vision Group, Universitat de les Illes Balears (UIB)
C/Valldemossa Km. 7.5, 07122, Palma de Mallorca, España
{josemaria.buades,paco.perales,vdmijvg4}@uib.es
http://dmi.uib.es/research/GV/
[2] Nursing and Physical Therapy Department, Universitat de les Illes Balears (UIB)
C/Valldemossa Km. 7.5, 07122, Palma de Mallorca, España
aaguilo@uib.es

Abstract. Today in many applications the study of human movement using a computer vision and graphics techniques is very useful. One of these applications is the three-dimensional reconstruction of the structure of the human body and its movement using sequences of images and biomechanical graphics models. In this paper we present a whole and general system to study the human motion without markers but, in this case, we apply it to high-level competition. This kind of study needs special procedures to do the analysis and reconstruction of the person's body, therefore the virtual human (avatar) must have similar anthropometrical characteristics than the person who is doing the movement.

We define a process to adjust the humanoid to the morphology of the person. It could be very laborious and subjective if done manually or by selection of points, but in this article we present a global human motion system capturing, modeling and matching a semiautomatic process between the real person and the modeled humanoid or synthetic avatar. Semiautomatic means that the computers propose the best matching from previous frames and the user can accept it or not.

1 Introduction

We will divide the general process of the system into four stages: in the first one, we capture images of the person from different points of view and of the background, in particular, up to four IEEE 1934 color cameras are used, that means that a initialization process is needed to know the exact anthropometrical data of the person that is moving in front of the cameras. In the second stage, we select the humanoid with similar characteristics to the original individual, so the result of this initialization process is an avatar with the same segments length of the real person. This process is need because in high-level sport activities the accuracy is very critical (in computer vision tracking the precision or accuracy requirements can be reduced). In the following stage we apply a semiautomatic process to obtain the humanoid adjusted to the person's measurements. The final goal is to reach an automatic recognition process, but this is a challenging task, that we are solving with less accuracy and under controlled environments. In the case of sport activities, a semiautomatic system is a good trade-off solution between full automatic systems and commercial standard systems using markers. The matching criteria propose a future pose of human segments based in previous frames and the user select the best one. Manual joints matching are not being done. The last stage combines the captured images and the generated humanoid to verify the result of the process and study the values we are interested in. We can use numerical data to make a deep study of performance of movements or paint special lines or points in real and synthetic images to help the trainer in the process of coaching to reach the perfect performance of the disabled sport people.

In particular the actual system is being used in indoor spaces and controlled environments. The system is also defined using new IEEE 1934 digital standards with portable and low cost systems. In the following sections we explain the capturing, modeling, and segmentation processes and also the matching criteria. Finally we conclude with an explanation about the biomechanical parameters studied, some interesting results and future work.

2 The Capturing Process and Modeling Humanoids

The tracking algorithm is divided in two parts: skin-color pixel detection and data association. For each frame, first, we segment the skin-color pixels using a previous learned probabilistically skin-color model and a blob analysis. Next, we use a hypothesis based data association algorithm to label the skin-color detected pixels. This algorithm uses a simple prediction step to the state obtained in the previous frame to calculate the new hypothesis.

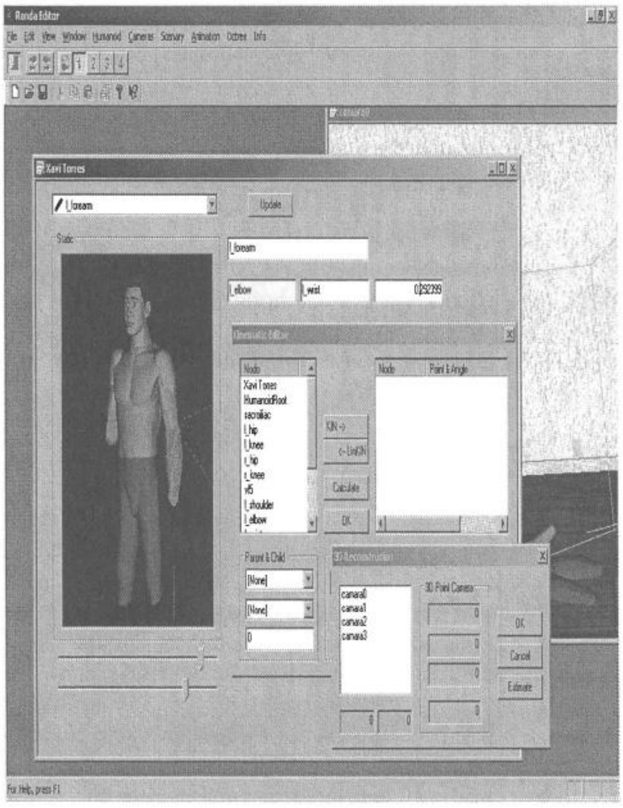
Here we introduce the basic ideas of capturing system and human modeling tools developed specifically for this system. We know that isn't critical point in research, but good implementation can improve runtime requirements. Therefore the results of this implementation have the following properties: versatility, independence of hardware, oriented object programming, efficiency and easily parallelizable.

In the module of human definition, there are various specifications [2] for humanoids; we have chosen the one created by the group h-anim[1] in VRML format for its portability and adaptability to different applications. The humanoid is composed of a collection of joints and segments structured in the form of a tree. Each joint corresponds to an anatomical joint (knee, shoulder, vertebrae), for example with each vertebra hanging from the one above, and the wrist joint hanging from the elbow joint. Each joint has associated with it a segment, it represents, for example, the elbow and the upper arm as its segment.

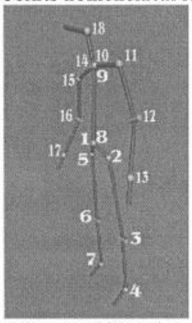
Each segment may have sites and displacers. A site corresponds to an endpoint, such as the fingertips, while a displacer corresponds to an effect applied to the segment. In particular we would like to define a multilevel system. That means that depending on the accuracy of applications, the system may include more degrees of freedom. For human motion recognition task particularly, the model is simpler and for human graphical simulations the system is more complex. Also, a modeling editor has been developed to adapt the human limitations of disabled people in this application. Figure 1 show a snapshot of the biomechanical editor. A simple animation tool is also

[1] www.hanim.org

included to generate simple kind of movements. The final expected result is a virtual model of the real person compliant with H-anim standards.



Joints nomenclature:



1:HumanoidRoot/sacroiliac *(guided)*
2: l_hip *(3 DOF)*
3: l_knee *(1 DOF)*
4: l_ankle *(3 DOF)*
5: r_hip *(3 DOF)*
6: r_knee *(1 DOF)*
7: r_ankle *(3 DOF)*
8: vl5 *(3 DOF)*
9: vc7 *(3 DOF)*
10: l_sternoclavicular *(3 DOF)*
11: l_shoulder *(3 DOF)*
12: l_elbow *(1 DOF)*
13: l_wrist *(3 DOF)*
14: r_sternoclavicular *(3 DOF)*
15: r_shoulder *(3 DOF)*
16: r_elbow *(1 DOF)*
17: r_wrist *(3 DOF)*
18: skullbase *(3 DOF)*

Fig. 1. and 2. H-anim biomechanical editor and Human Structure

Another property of the system is the independence between hierarchical structure and graphical representation. That means, that we can change the graphical primitives and represent the same biomechanical model with wireframe, shapes or volumetric primitives. Of course a more simple biomechanical structure is used in the matching process. In particular, the body structure used has one guided joint, 13 spherical joints and 4 cylindrical joints. That means 43 degrees of freedom in total, but in special cases the joints are simplified. In figure 2 we can see the simplified model.

3 Image Color Segmentation Procedure

Image segmentation is the first step in data extraction for computer vision systems. Achieving good segmentation has turned out to be extremely difficult, and it is a complex process. Moreover, it depends on the technique used to detect the uniformity of the characteristics founded between image pixels and to isolate regions of the image that have that uniformity. Multiple techniques have been developed to achieve this goal, such as contour detection, split and merging regions, histogram thresholding, clustering, etc. A Survey can be found in [7]. In color image processing, pixel color is usually determined by three values corresponding to R (red), G (green) and B (blue). The distinctive color sets have been used with different goals, and specific sets have even been designed to be used with specific segmentation techniques.

We define a color image as a scalar function $\boldsymbol{g} = (\boldsymbol{g}^1, \boldsymbol{g}^2, \boldsymbol{g}^3)$, defined over image domain $\boldsymbol{\Omega} \subseteq \Re^2$ (normally a rectangle), in such a way that $\boldsymbol{g}$: $\boldsymbol{\Omega} > \Re^3$. The image will be defined for three channels, under the hypothesis that they are good indicators of autosimilarity of regions. A segmentation of image g will be a partition of the rectangle in a finite number of regions; each one corresponding to a region of the image where components of g are approximately constant. As we will try to explicitly compute the region boundaries and of course control both their regularity and localization, we will use the principles established in [1, 8] to define a good segmentation. To achieve our goals we consider the functional defined by Mumford-Shah in [6] (to segment gray level images) which is expressed as:

$$E(u,B) = \int_{\Omega} \sum_{i=1}^{3} (u^i - g^i)^2 d\mu + \lambda \ell(B) \tag{1}$$

where B is the set of boundaries of homogenous regions that define a segmentation and u (each $\boldsymbol{u}^k$) is a mean value, or more generally a regularized version of g (of each $\boldsymbol{g}^k$) in the interior of such areas. The scale parameter λ in the functional (1) can be interpreted as a measure of the amount of boundary contained in the final segmentation B: if λ is small, we allow for many boundaries in B, if λ is large we allow for few boundaries.

A segmentation B of a color image g will be a finite set of piecewise affine curves - that is, finite length curves - in such a way that for each set of curves B, we are going to consider the corresponding u to be completely defined because the value of each $\boldsymbol{u}^i$ coordinate over each connected component of $\boldsymbol{\Omega} \setminus \boldsymbol{B}$ is equal to the mean value of $\boldsymbol{g}^i$ in this connected component. Unless stated otherwise, we shall assume that only one $\boldsymbol{u}$ is associated with each B. Therefore, we shall write in this case $E(B)$ instead of $E(u, B)$. A segmentation concept, which is easier to compute, is defined as follows:

Definition 1. A segmentation B is called 2-normal if, for every pair of neighboring regions O_i y O_j, the new segmentation B' obtained by merging these regions satisfies E(B') > E(B).

We shall consider only segmentations where the number of regions is finite, in other words $\boldsymbol{\Omega} \setminus \boldsymbol{B}$ has a finite number of connected components and the regions do not have internal boundaries.

A more detailed explanation of the concepts and their mathematical properties can be consulted in [1, 8] and we can see the properties of the functional in [1,6]. The use of these theoretical concepts in the case of multi-channel images (e.g. color images) can be seen in [1]. We shall use a variation of segmentation algorithm by region merging described in [8] adapted to color images.

The concept of 2-normal segmentations synthesizes the concept of optimal segmentation we are looking for, and it lays on the basis of the computational method we use. The whole process is presented in a previous work [4]. For further explanation please refer to this paper.

The algorithm uses the RGB components because the segmentations obtained are very accurate to our goal. But the system is able to use other color space or color descriptor as we can see in [7]. Moreover, if it is needed it can weigh the channels used in order to obtain the segmentation. After the segmentation process we test every region to detect skin regions. Skin color is a characteristic color that is very different to other colors.

Skin color test is applied, but underwater skin regions are not detected due to water distortion in light spectrum. In any case the results cases are not bad, and we are using only standard images from commercial video cameras. In the future we plan to use special light conditions and they might be infrared images. Some special clothing at end effectors can also help us in the segmentation process.

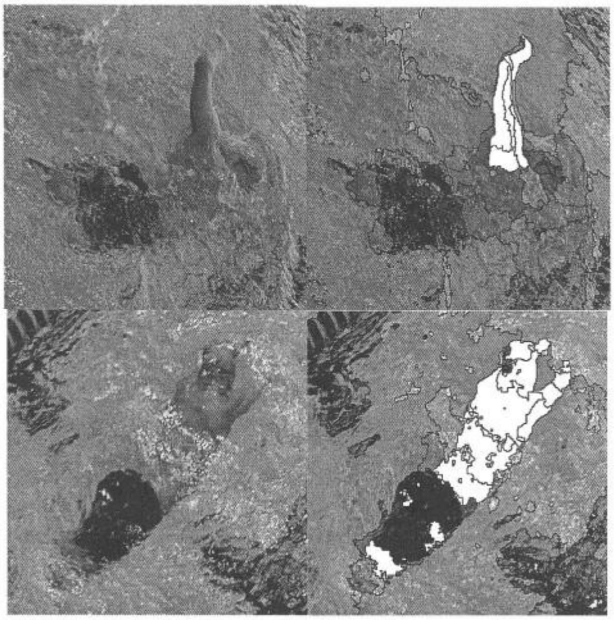


Fig. 3. Arm and skin region segmentation

In the previous figures we can see some color segmentation results. Also we include others segmentation results in Figure 5 with indoor testing images using green chroma key for background.

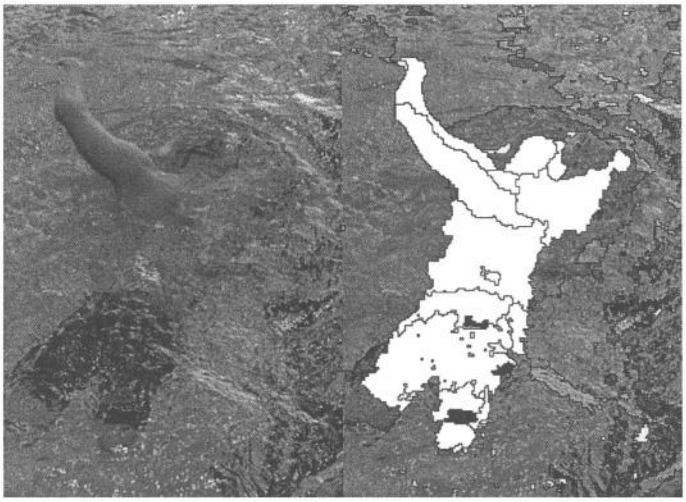


Fig. 4. Whole human body segmentation

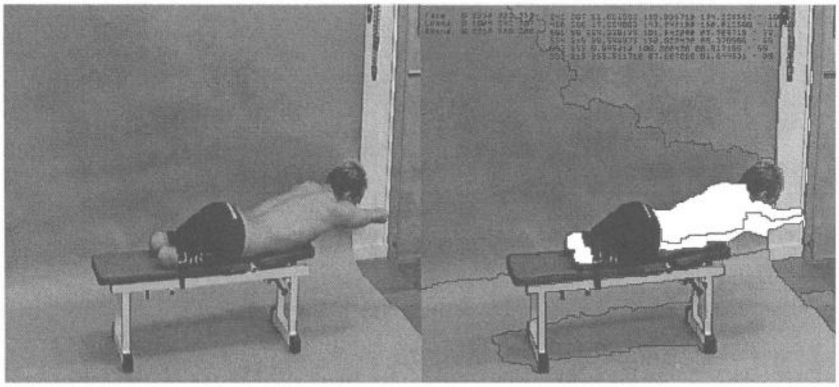


Fig. 5. Human body segmentation with uniform background

4 The Matching Criteria

In the proposed system, Figure 5 display a diagram of the task framework, the matching process is the kernel. Our main objective is to find a one-to-one correspondence between real and synthetic human body segments and joints, in every frame, in 3D space. The humanoid matching is currently done semi automatically; we are working on automatic process for a general indoor environment and, for the moment we have reached promising results in cases where occlusions are not so long in time. A trade off solution is a semi-automatic process that means, a human interaction in matching process, but supervised by computer that avoids wrong biomechanical postures. As we have mentioned above, the system can estimate a posture from previous frames matching in time t_{i-1}. This estimation is based on a function that evaluates visual parameters (contours, regions, color, etc.) and

biomechanical conditions. The general method of this problem is known as *analysis-by-synthesis*. As we know, to search quickly in a 43-dmiensional state is extremely difficult; therefore we propose some set of conditions to reduce our space search.
The information captured comes from the color cameras in three positions. They follow specific anthropometrics criteria, from which we obtain the desired parameters. We use a database of predefined movements and models to help the matching process. We can also use this knowledge to estimate new movements that are not previously recorded. The semiautomatic matching process has certain limitations of precision according to the models and the calibration of the cameras. This process is based on a set of well-defined conditions.

The matching process associates each joint with a 2D point in the image. This process consists on analyzing each image obtained from the cameras in an instant of time *t*. Once we have the articulation located in two or more cameras we estimate the most accurate 3D point; a joint may only be detected in one or none image, therefore the process will have to be completed with contextual information from a higher level. Analyzing the sequence, we are able to apply physical and temporal restrictions, which help us to carry out the matching and to reduce the errors and the search space. This adjustment process is conditioned by a set of conditions, which are optimized in each case and type of movement. The restrictions are:

– Angle and distance limitations of the joints.
– Temporal continuity of the movement in speed or acceleration.
– Prediction of the movement based on a database of known movements, in the case of having non-visible joints.
– Collision of entities.

This matching process works currently in an automatic way for simple movement parallel to the camera plane and in a semiautomatic way in complex environments, thus making that it is possible to work at different levels of precision depending on the application the results are required for.

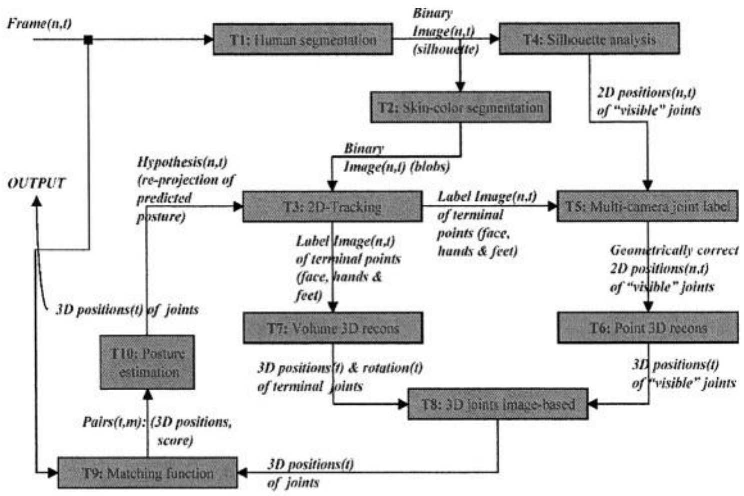

Frame(n,t)
T1: Human segmentation
Binary Image(n,t) (silhouette)
T4: Silhouette analysis
T2: Skin-color segmentation
2D positions(n,t) of "visible" joints
Binary Image(n,t) (blobs)
OUTPUT
Hypothesis(n,t) (re-projection of predicted posture)
T3: 2D-Tracking
Label Image(n,t) of terminal points (face, hands & feet)
T5: Multi-camera joint label
Label Image(n,t) of terminal points (face, hands & feet)
Geometrically correct 2D positions(n,t) of "visible" joints
3D positions(t) of joints
T7: Volume 3D recons
T6: Point 3D recons
T10: Posture estimation
3D positions(t) & rotation(t) of terminal joints
3D positions(t) of "visible" joints
T8: 3D joints image-based
Pairs(t,m): (3D positions, score)
T9: Matching function
3D positions(t) of joints

Fig. 6. Our proposed system and task framework

In Figure 7 we display some matching results. In this case, the matching is done semi-automatically and without calibration.

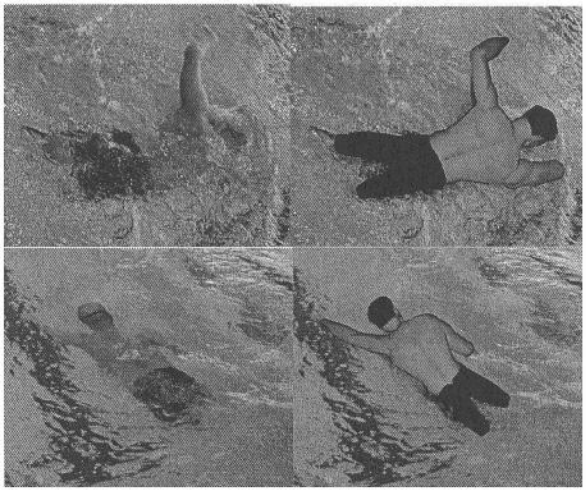


Fig. 7. Matching examples with non-calibrated cameras

5 Biomechanical Parameters Studied

To study a specific swimming human model we need a set of well-defined parameters. The following list includes the most important ones. The final aim of this work is to obtain all these parameters in an automatic way. Unfortunately we know that this task is very complex and challenging, so at the moment only a subset are reached with computer vision segmentation and analysis techniques:

- Defining the skew of upper limbs in order to make a difference in the biomechanics performance between swimmers.
- Calculating and observing the movement of the gravity center
- Defining the anthropometrics parameters of several segments:

  a) Length of both lower limbs,
  b) Length of both upper limbs,
  c) Total length of each half-body,
  d) Distance between both acromioclavicular articulations,
  e) Distance from each lower limbs to the vertex,
  f) Length of trunk: distance between the suprasternal point and the middle point between both hips,
  g) Length of the three segments of the left upper limb: From the acromioclavicular to the olecranon, g.2) From olecranon to the styloid apophysis and g.3) From the styloid apophysis to the distal phalanx of the finger.

At the moment, all these parameter are computed in a manual or semiautomatic way. Only a subset is computed automatically (for example: center mass point).

6 Conclusions

In this paper we have presented a whole system to analysis and synthesize human movements. The main outstanding research and contributions are: a) Robust programming of a digital IEEE1934 synchronized and portable low cost capturing system (that's isn't a non trivial task!), b) Biomechanical compliant human modeling from our vision oriented model to an H-Anim, c) Robust color segmentation process using solid mathematical background theory, d) Semiautomatic matching process based on rules from images features and biomechanical conditions e) Good accuracy application with supervised matching criteria in real sport activities and handicapped users. The system isn't really full automatic, but much better than manual systems (in time processing and accuracy). In general the final system should allow us to detect, follow, and recognize the activities of one or more people in a scene [4]. The part presented allows editing the humanoid only under supervision, but a possible extension would be obtaining cinematic and dynamic means with a view to more sophisticated applications. In fact, a determined level of precision will be defined according to the application.

This work is framed within a more general project of analysis and synthesis of human movement using techniques of digital image processing and computer vision. Is very important to remark that the system proposed is non <u>invasive</u> and <u>does not use markers</u> on people. The biomechanical model is also over placed on real images and we don't need individual manual digitalization for every frame and an interactive feedback with the model is possible in real time processing.

The last part presents the adaptation of the proposed system to the analysis of sports motion in all possible variations, in particular for people with several physical limitations. Obviously each discipline has its parameters of interest, which will be defined by the user or specialized technician. The precision of the system is adequate for indoor sequences. We have problems with water interferences in segmentation process and we plan to include underwater images in near future. We are working to improve the segmentation process and reach the biomechanical data as automatically as we can. The manual interaction must be avoided to reach more precision in human matching. We are working with filtering estimation using predictive methods and dynamic and kinetic control of biomechanical solution proposed.

Acknowledgements

This work is subsidized by the European project HUMODAN IST-2001-32202 and Spanish Contract TIC2003-0931 & TIC2002-10743-E. Also we would like to express ours thanks to all members of Computer Graphics and Vision group by their explicit and implicit support to develop this work. We also want to make a special mention to Mr. Xavi Torres, a para-olimpic swimmer who has collaborate with us in the experiments `http://www.xavitorres.com/`.

References

[1] J.M. Morel and S. Solimini. "Variational Methods for Image Segmentation", Birkhauser Verlag. 1995

[2] N. Badler, C. Phillips, B. Webber. *Simulating Humans. Computer Graphics Animation and Control.* Oxford University Press, 1993.

[3] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland. "Pfinder: Real-Time Tracking of the Human Body". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 7, 1997, pp 780-785.

[4] J.M. Buades, M. Gonzalez, F.J. Perales. "Face and Hands Segmentation in Color Images and Initial Matching" International Workshop on Computer Vision and Image Analysis. Palmas de Gran Canaria. Dec. 2003. pp 43-48

[5] Jessica K.Hodgins, Nancy S. Pollard. "Adapting Simulated Behaviors For New Characters". Computer Graphics Proceedings, 1997pp. 153-162

[6] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and variational problems", Communications on Pure and Applied Mathematics, XLII(4), 1989

[7] H.D. Cheng, X.H. Jiang, Y. Sun, JinGli Wang "Color Image Segmentation: Advances and Prospects", Journal of Pattern Recognition 34, (2001), pp. 2259-2281

[8] G. Koepfler, J.M. Morel, and S. Solimini, "Segmentation by minimizing a functional and the merging methods", SIAM J. on Numerical Analysis, Vol 31, No 1, Feb. 1994

[9] H. Seo, N. Magnenat Thalmann."An Example-Based Approach to Human Body Manipulation", Graphical Models, Academic Press, Vol. 66, No. 1, pp. 1~23., January 2004

[10] H. Seo, N. Magnenat-Thalmann, "An Automatic Modeling of Human Bodies from Sizing Parameters", ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics, pp19-26, pp234, 2003

[11] L.Vacchetti, V.Lepetit, G.Papagiannakis, M.Ponder, P.Fua, N.Magnenat-Thalmann, D.Thalmann "Stable Real-Time Interaction Between Virtual Humans and Real Scenes", 3DIM, pp. 449-456, Banff, Alberta, Canada, 2003

[12] Mun Wai Lee and Isaac Cohen, Human Upper Body Pose Estimation in Static Images, Institute for Robotics and Intelligent Systems, Integrated Media Systems Center, University of Southern California Los Angeles, CA 90089-0273, USA

[13] I.A. Kakadiaris, D. Metaxas, 3D Human body model acquisition from multiple views, in: Proceedings of the Fifth International Conference on Computer Vision, pp. 618–623, Boston, MA, June 20–23,1995.

[14] Raquel Urtasun and Pascal Fua, 3D Human Body Tracking Using Deterministic Temporal Motion Models, Computer Vision Laboratory, EPFL, CH-1015 Lausanne, Switzerland

A Comparison of Algorithm Design Paradigms in Active Contours for Muscle Recognition

A. Caro[1], P.G. Rodríguez[1], M.L. Durán[1], M.M. Ávila[1],
T. Antequera[2], and R. Gallardo[3]

[1] University of Extremadura, Computer Science Dept.
Escuela Politécnica, Av. Universidad s/n, 10071 Cáceres, Spain
{andresc,pablogr,mlduran,mmavila}@unex.es
[2] University of Extremadura, Food Technology Dept.
Facultad Veterinaria, Av. Universidad s/n, 10071 Cáceres, Spain
tantero@unex.es
[3] "Infanta Cristina" University Hospital, Radiology Service
Badajoz, Spain

Abstract. Active Contours constitute a widely used Pattern Recognition technique. Classical active contours are based on different methodologies. This paper reviews the algorithm paradigms most frequently utilized in active contours (variational calculus, dynamic programming and greedy algorithm) in a practical application, comparing chemical data with computer vision results. An experiment has been designed to recognize muscles from Magnetic Resonance (MR) images of Iberian ham at different maturation stages in order to calculate their volume change, using different active contour approaches. Our main findings can be summarized as two: The feasible application of active contours to recognize muscles in MR images, and the early way to automate the ripening process for Iberian ham.

1 Introduction

Active Contours (or snakes) is a low-level processing technique widely used to extract boundaries in many pattern recognition applications [3, 7]. In their formulation, active contours are parameterized curves, defined by an energy function. By minimizing this energy function, the contour converges, and the solution is achieved. An Active Contour is represented by a vector, *v(s)*, which contains all of the *n* points of the snake. The functional energy of this snake is given by:

$$E = \int \left[E_{\text{int}}(v(s)) + E_{image}(v(s)) \right] ds \tag{1}$$

$$E_{\text{int}} = \alpha(s)\left|v_s(s)\right|^2 + \beta(s)\left|v_{ss}(s)\right|^2 \tag{2}$$

$$E_{image} = -\gamma(s)\left|\nabla I(v(s))\right|^2 \tag{3}$$

where v_s and v_{ss} are the first and second derivatives of *v(s)*, with respect to *s*, respectively, and ∇I is the gradient of the image *I*.

Energy-minimizing Active Contour models were proposed by Kass *et al.* [9]. They developed a controlled continuity spline which can be operated upon by internal contour forces, images forces, and external forces which are supplied by an interactive user, or potentially by a higher level process. An algorithmic solution involves derivation of this objective function and optimization of the derived equation for finding an appropriate solution. However, in general, variational approaches do not guarantee global optimality of the solution [1].

On the other hand, Amini *et al.* [1] also proposed a dynamic programming algorithm for minimizing the functional energy. However, the proposed algorithm is slow, having a great complexity. Williams and Shah [13] developed a greedy algorithm which has performance comparable to the dynamic programming and variational calculus approaches.

Active Contours could be used as a pattern recognition technique. A practical application of them could be employed to recognize muscles of Iberian ham images [4, 5]. Particularly, Iberian ham images have been processed in this research in order to find out some characteristics and reach conclusions about this excellent product. The evolution of the ripening of this meat product has been studied, acquiring images in different stages during the maturation process. The ripening of the Iberian ham is a lengthy procedure (normally 18-24 months). Physical-chemical and sensorial methods are required to evaluate the different parameters in relation with quality, being generally tedious, destructive and expensive [2]. Traditionally, the maturation time is fixed when the weight loss of the ham is approximately 30% [6]. So, other methodologies have long been awaited by the Iberian ham industries.

The use of image processing to analyze Iberian products is quite recent. Processing images from Iberian ham slices taken by a CCD camera is a first approach. However, although Computer Vision is essentially a non-destroying technique, ham pieces must be destroyed to obtain images using these techniques.

On the other hand, MRI *(Magnetic Resonance Imaging)* offers great capabilities to non-invasively look inside the bodies. It is widely used in medical diagnosis and surgery. It provides multiples planes (digital images) of the body or piece. Its application to the Food Technology is still recent and it is confined for researching purposes.

In this work, three algorithm paradigms in active contours have been developed, *variational calculus, greedy algorithm* and *dynamic programming,* in order to recognize the two main muscles structures in the Iberian ham (*biceps femoris* and *semimembranosus* muscles). The experiment has been designed having Magnetic Resonance (MR) images from four different stages during the maturation of the ham *(raw, post-salting, semi-dry and dry-cured).* The main aim is to recognize muscles processing MR images, to determine the volume of ham, and to study their changes during the ripening process. This main objective is an attempt to provide a computer vision alternative to the traditional methods of determining the optimal ripening time. A second goal is to find out if there is a method which does not work fine, and, eventually, to compare the physical and computer vision results.

2 Data Set

The presented research is based on MRI sequences of Iberian ham images. A technique to recognize the main muscle form *(bicepsfemoris* and *semimembranosus)* is employed. Six Iberian hams have been scanned, in four stages during their ripening time. The images have been acquired using an MRI scan facilitated by the "Infanta Cristina" Hospital in Badajoz (Spain). The MRI volume data set is obtained from sequences of T1 images with a FOV (*field-of view*) of 120x85 mm and a slice thickness of 2 mm, i.e. a voxel resolution of 0.23x0.20x2 mm. As a result, a great image data base is obtained, selecting from it a subset of images which contains both muscles of interest. The total number of images of the obtained database is 600 for the *biceps femoris,* and 504 for the *semimembranosus* muscle. So, the total amount of images is 1104.

3 Methodology

Three different algorithm paradigms in active contours (*variational calculus, greedy algorithm* and *dynamic programming*) have been proven in order to obtain results using several methodologies (section 3.1). Afterwards, the practical application of these three methodologies over MR Iberian ham images from different maturation stages will be shown in 3.2.

3.1 Classical Active Contour Approaches

An Active Contour is represented by a vector, v, of points [3]. The complete energy functional in all the developed methods is given by:

$$E = \sum \{\alpha E_{cont} + \beta E_{curv} + \gamma E_{image}\} \tag{4}$$

$$E_{cont} = \left| \bar{d} - |v_i - v_{i-1}| \right| \tag{5}$$

$$E_{curv} = |v_{i-1} - 2v_i + v_{i+1}|^2 \tag{6}$$

$$E_{image} = -\frac{|\nabla I|}{e_{\max}} \tag{7}$$

The internal energy of the contour consists of continuity energy (E_{cont}) plus curvature energy (E_{curv}). E_{image} represents the proper energy of the image [10, 14]. All the energy terms are normalized to produce similar effects to the solution. α, β and γ are values chosen to control the influence of the three energy terms [11, 12]. Table 1 shows the different values used in each method. These final values have been calculated empirically, and the most important detail to highlight is the image energy is always higher than the sum of the other two energy terms.

Table 1. Parameters used to weigh up the energy terms

Method	α	β	γ
Variational	0.4	0.4	1.2
Dynamic	0.4	0.4	2.4
Greedy	0.4	0.4	1.2

Although most of the active contour algorithm and practical approaches consider a fixed number of *n* points for the snake, the three different active contour approaches developed in this work maintain a variable number, *n,* of points of the snake. This is the best way to ensure the finest muscle recognition. To achieve that, two distances have been consider, d_{max} and d_{min}. When two consecutive points of the snake exceed the maximum distance between points d_{max}, a new point appears between the two original points. In such a way, when two consecutive points go beyond the minimal distance d_{min}, one of them disappears. As a result, the total number of points of the snake varies from one iteration to another, adjusting perfectly the final snake to the pattern.

The **variational calculus** approach is based on the finite difference method proposed by Kass et al. [9]. By approximating the derivatives in equation (1) with finite differences, and converting to the vector notation described in [14], this equation is written in a compact matrix form. This one can be solved iteratively by matrix inversion, using the *Lower and Upper triangular decomposition,* a well-known technique in linear algebra. A detailed description of the principle behind this numerical method is described in [8, 14]. Moreover, as external force, the Gradient Vector Flow proposed in [14] has been used, which computes as a diffusion of the gradient vectors of a gray-level map derived from the image. In this way, a first active contour method has been implemented, achieving the initial group of results. If there are *n* points for the snake, and *m* directions at each point (neighborhood), each iteration has complexity $O(nm^2)$.

On the other hand, the developed **dynamic programming** approach is based on the method proposed by Amini [1]. Although it is natural to view energy minimization as a static problem, to compute the local minima of a functional such as equation (1) it is possible to construct a dynamical system that is governed by the function and allow the system to evolve to equilibrium. The efficiency of this technique consist in solving subtasks just the once, storing theis partial solutions in a table for its future reutilization. Following these ideas, a dynamic programming active contour has been designed and used to reach a set of results, having complexity $O(nm^3)$.

Finally, the implemented **greedy algorithm** approach is based on the Williams and Shah's proposals [13]. This technique achieve the final solution by steps, trying to take always an optimal decision for each stage. The algorithm is iterative, and it considers a square $m\text{x}m$ neighborhood for each point of the snake. The energy function is computed for the current location of v_i and each of its neighbors. The location having the smallest value is chosen as the new position of v_i. Using this methodology, the last active contour approach has been developed, with a complexity $O(nm)$, obtaining the last set of results.

3.2 Practical Application: Active Contours on Iberian ham MRI

An experiment was designed using the three considered methods above to study the ripening process of the hams. Four stages were selected: *raw, post-salting, semi-dry* and *dry-cured,* acquiring MR images of the hams in each of the stages. The initial analysis began with 15 hams in the raw stage. Moreover, three hams were destroyed and the muscle pieces were extracted by an expert in each stage to realize the chemical analysis on *biceps femoris* and *semimembranosus* muscles. For this reason, there only were 12 hams in the post-salting stage, 9 in the semi-dry stage, and finally 6 hams in the dry-cured stage. Therefore, there only are MR images in the four stages for the six considered Iberian hams.

As a previous step, a **pre-processing stage** is introduced, in order to obtain the values used as image energy. A 7x7 Gausian filter has been used to smooth the images at this stage, as well as a 3x3 Sobel operator. The resulting images were equalized and converted to binary images, removing noisy pixel from them and calculating the potential fields used as external force for the snakes. These potential fields were calculated as a degradation (diffusion) of the binary edge map derived from the image.

In addition, the **initial snakes** for all the images have been previously calculated too. Searching in the potential fields matrixes in order to find the points with the smallest values is required. The key is to distribute all the points of the contour surrounding all those points of the image with smallest potential field values. In this manner, it is ensured that the snake will evolve towards the edges of the object, searching for points with levels of energy smaller than the energy values of the points in the initial snake.

Once the complete database of images and the initial values of the snakes for these images are obtained, the **application of Active Contours** to compute the area of the muscle is needed. Every of the three active contours methods exposed above (variational calculus, greedy algorithm and dynamic programming) have been developed, obtaining three different sets of results.

Each of the obtained final snakes determines the surface of the muscle over the image. The final step computes **surfaces and volumes** for the extracted muscles. The surface for all the final snakes of the ham has been calculated using the classical methods in analytical geometry. Knowing the surfaces for the ham images and the distance between slices is possible to determine the volume for the whole muscle.

Eventually, in average, the error made has been estimated at less than 10%, considering the manual expert delineation of the muscles compared with the final area of the snake-segmented muscle.

Figure 1 contains MR images with the final snake for both *bicepsfemoris* (a) and *semimembranosus* (b) muscles. Figure 2 corresponds to the stages of the designed experiment.

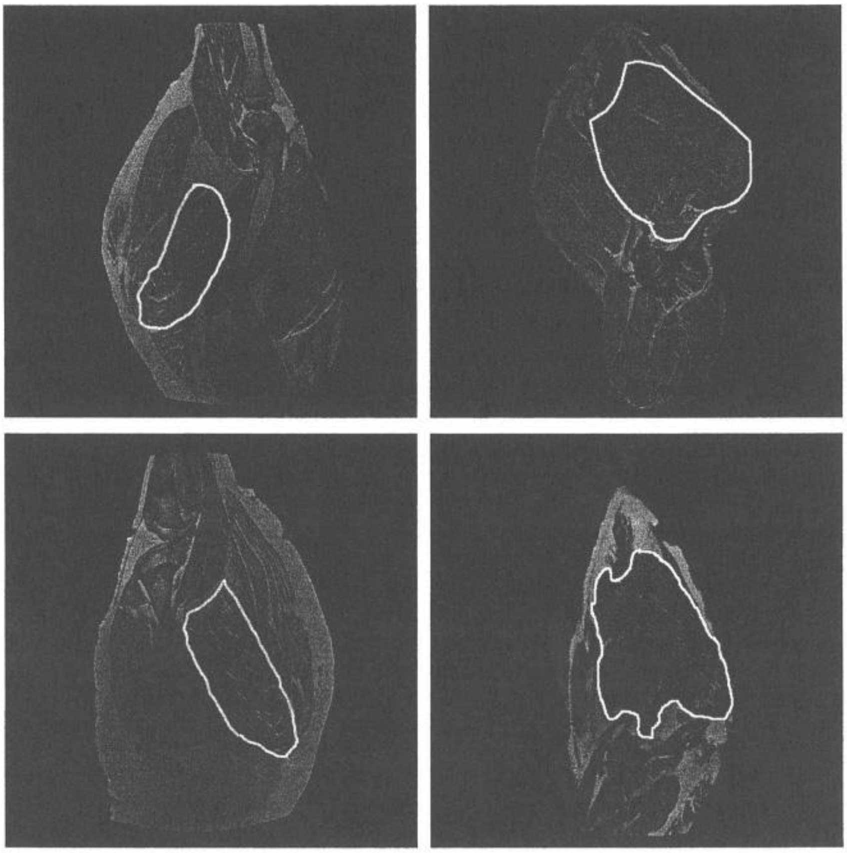


Fig. 1. Illustration of Iberian ham MR images, which include the detection of the muscles

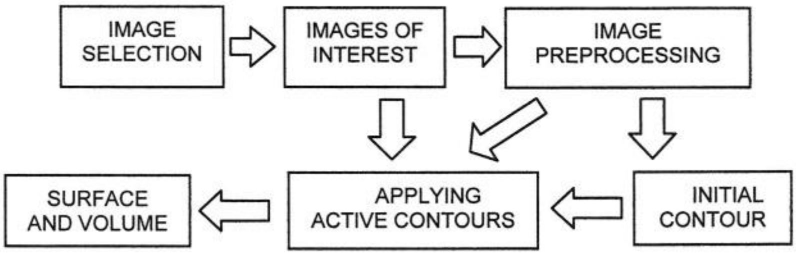
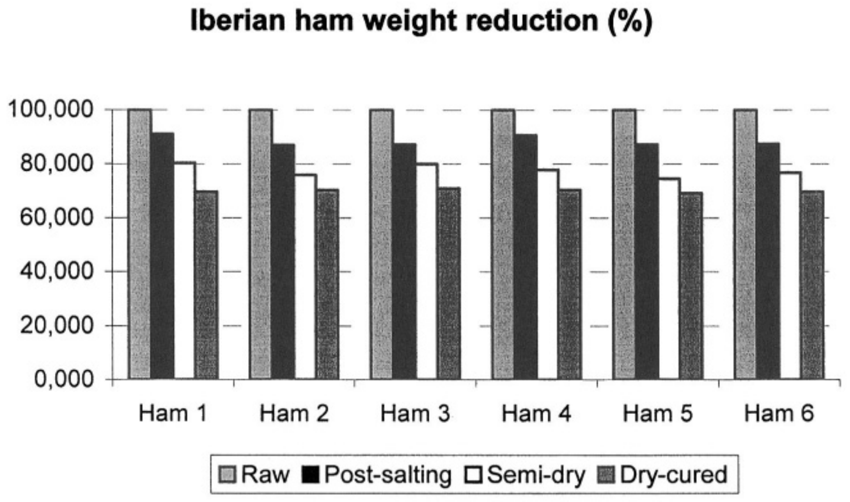


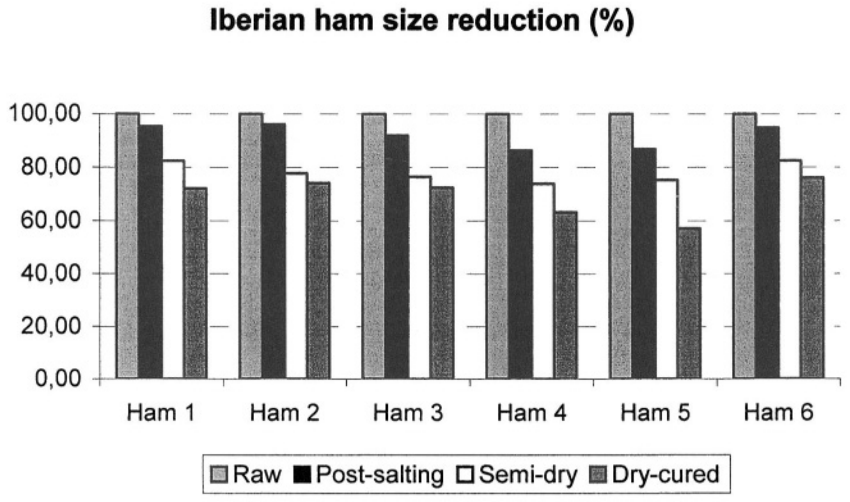
Fig. 2. Stages of the practical application

4 Results and Discussion

The practical application of the three active contour methods shows the volume reduction of the Iberian ham *biceps femoris* and *semimembranosus* muscles during its ripening stages.



a) Physical results, obtained by weighing manually the hams



b) Computer vision results, obtained by applying active contours automatically

Fig. 3. Weight (a) and size (b) reduction during ripening time in Iberian hams

It is expected the ham to be cured when its weight loss is approximately 30%, percentage established by tradition. Fig. 3.-a shows how all the six hams ensure this traditional rule. These results have been obtained weighing manually the hams.

On the other hand, considering the results obtained by the three different active contour techniques, it is particularly interesting to emphasize that they are certainly similar, independently of what method is used. Fig. 3.-b shows the volumetric evolution of the hams in the same four maturation stages, using the results obtained by computer vision techniques. It is significant the results have not been substantially different between both physical and computer vision results.

Figure 4-b shows the average of three *bicepsfemoris* weight reduction during the ripening process in each stage. The obtained results using the three active contour methods are shown in figure 4-a. Both physical and computer vision results are certainly comparable. Similarly, figure 5 shows the same results corresponded to *semimembranosus* muscle. The physical extraction of this last muscle by Food Technology experts is a complex procedure. For this reason, it is assumed the produced inaccuracy between the post-salting and semi-dry stages (figure 5-b). However, from a positive point of view, the obtained results by the three active contour methods coincide with what were expected (figure 5-a).

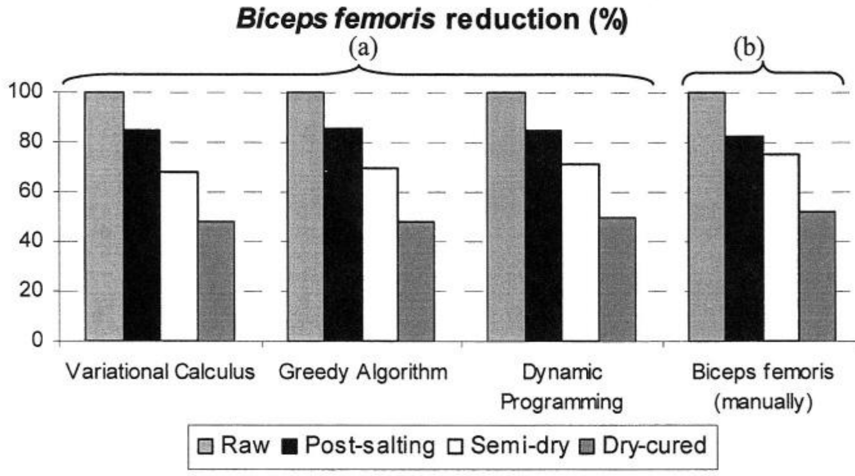


Fig. 4. Size (a) and weigh (b) reduction during ripening time for the biceps femoris muscle

The results presented in figures 4 and 5 demonstrate a size reduction of up to 50% as an average score going from the initial phase (raw) to the last stage (dry-cured), 21 months after the initial stage.

Food Technology specialists have estimated the total weight decrease in the Iberian ham to be 30% during the same time. Thus, a relationship between the ham weight decrease (30%) and muscle reduction (50%) could be first established for maturation as an approximation. A more detailed study is carried out in our approach: Weight decreases may be caused by the loss of water during maturation time. Optimal ripening time could not be the same for different Iberian hams. In addition, by studying the percentage rate of volume reduction during the ripening process, it was possible to predict the optimal maturation point.

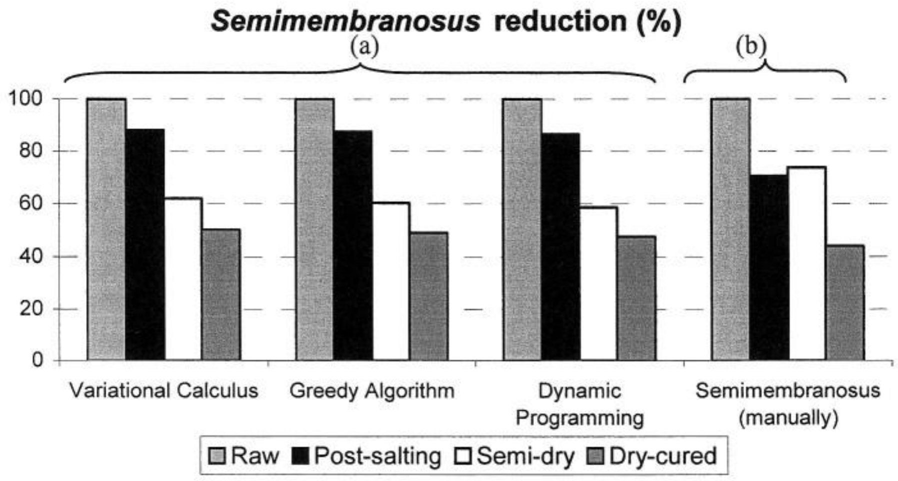


Fig. 5. Size (a) and weigh (b) reduction during ripening time for the *semimembranosus* muscle

5 Conclusions

This paper has proved the real viability of the three classical active contour approaches for muscle recognition in MRI. In addition, a comparative study between physical (chemical and sensorial analysis) and computer vision results have been achieved, verifying the robustness of the employed methodology. Moreover, the practical feasibility of applying Computer Vision techniques, in conjunction with MRI, to automate optimal ripening time in Iberian hams, constitutes a key finding in our research. Such computer vision techniques may introduce new and alternative methods for future work, which together with traditional chemical processes, may serve to evaluate different physical properties that characterize the Iberian ham.

Acknowledgements

The authors wish to acknowledge and thank the support of the "Dehesa de Extremadura" brand name and the "Hermanos Alonso Roa" company from Villar del Rey (Badajoz) to our study. In addition, this research has been funded by the Junta de Extremadura (Regional Government Board) under the IPR98A03P- and 2PR01C025-labeled projects.

References

[1] Amini, A.A., Weymouth, T.E. and Jain, R.: Using Dynamic Programming for Solving Variational Problems in Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, 855-867, (1990)

[2] Antequera, T., López-Bote, C.J., Córdoba, J.J., García, C., Asensio, M.A., Ventanas, J. and Díaz, Y.: Lipid oxidative changes in the processing of Iberian pig hams, Food Chem., 54, 105, (1992)

[3] Blake, A. and Isard, M.: Active Contours. Springer, London – UK, (1998)

[4] Caro, A., Rodríguez, P.G., Cernadas, E., Durán, M.L., Antequera, T.: Potencial Fields,as an External Force and Algorithmic Improvements in Deformable Models, Electronic Letters on Computer Vision and Image Analisys, Vol. 2(1), 23-34 (2003)

[5] Caro, A., Rodríguez, P.G., Ávila, M., Rodríguez, F., Rodríguez, F.J.: Active Contours Using Watershed Segmentation, IEEE **9th** Int. Workshop on Systems, Signal and Image Processing, Manchester - UK, 340-345, (2002)

[6] Cava, R. and Ventanas, J., Dinámica y control del proceso de secado del jamón ibérico en condic. naturales y cámaras climatizadas, T. jamón ibérico, Mundi Prensa, 260-274, (2001)

[7] Cohen, L.D.: On Active Contour Models and Balloons, Computer Vision, Graphics and Image Processing: Image Understanding, Vol. 53(2), 211-218, (1991)

[8] Courant, R. and Hilbert, D., Methods of Mathematical Physics, Interscience, Vol. 1, New York, (1953)

[9] Kass, M., Witkin, A. and Terzopoulos, D.: Snakes: Active Contour models, Proceedings of First International Conference on Computer Vision, London, 259-269, (1987)

[10] Kichenassamy, S., Kumar, A., Olver, P., Tannenbaum, A. and Yezzi, A.: Gradient Flows and Geometric Active Contour Models, Proc. Int. Conf. on Computer Vision, Cambridge, (1995)

[11] Larsen, O.V., Radeva, P. and Martí, E.: Guidelines for Choosing Optimal Parameters of Elasticity for Snakes, Proc. Int. Conf. Computer Analysis and Image Proces., 106-113, (1995)

[12] Ranganath, S.: Analysis of the effects of Snake Parameters on Contour Extraction, Proc. Int. Conference on Automation, Robotics, and Computer Vision, 451-455, (1992)

[13] Williams, D.J. and Shah, M.: A Fast Algorithm for Active Contours and Curvature Estimation, C. Vision, Graphics and Im. Proc.: Im. Understanding, Vol. 55, 14-26, (1992)

[14] Xu, C. and Prince, J. L.: Gradient Vector Flow: A New External Force for Snakes, IEEE Proc. on Computer Vision and Pattern Recognition, (1997)

Real Time Segmentation and Tracking of Face and Hands in VR Applications

José M. Buades, Francisco J. Perales, and Javier Varona

Unidad de Gráficos y Visión por Ordenador, Universitat de les Illes Balears (UIB)
C/Valldemossa Km. 7.5, 07122 - Palma de Mallorca - España
{josemaria.buades,paco.perales,vdmijvg4}@uib.es
http://dmi.uib.es/research/GV/

Abstract. We describe a robust real-time 3D tracking system of the extreme limbs of the upper human body, i.e., the hands and the face. The goal of the system is that it can be used as a perceptual interface for virtual reality activities in a workbench environment. The whole system includes an input capture and calibration module, a real time color segmentation module, a data association and tracking module and finally a visualization VRML and H-ANIM procedure. The results of our probabilistically skin-color segmentation are skin-color blobs. Then, for each frame of the sequence our algorithm labels the blobs' pixels using a set of object state hypothesis. This set of hypothesis is built from the results of previous frames. The 2D tracking results are used for the 3D reconstruction of limbs position in order to obtain the H-ANIM visualization results. Several results are presented to show the algorithm performance.

1 Introduction

In actual computer systems the interaction is going to a non-contact devices, by means of perceptual and multimodal user interfaces (PUIs and MUIs). That's means that the system allows the user to interact without physical contact with the machine; this communication can be carried out with voice or user gesticulation capture. We are especially interested in visual information, so recognize the human presence in color video images. Exist many precedent work developed in this hot topic: in [3] including finger detection with multi-scale color approach, in [8] in real time for virtual reality applications but in 2D, in [9] to human grasping but using infrared images including a hand model with 15 joints (20 D.O.F.). For our purposes, we would like to define a general, robust and efficient system that can be used with non-expensive analogical or digital cameras (using day light spectrum) and can recover 3D hands and face pose of the human person in real time. Capture is carried out from digital IEEE 1394 color cameras. The process is applied to stereo cameras to recover 3D positions. In the presented paper is used a high quality workbench but the system is adaptable to others less expensive systems.

The global process must detect a new user entering the system and analyze him/her to determine parameters such as hair color and clothes. Once the user who is going to interact with the machine has been detected, the system starts to track interesting regions such as the head, hands, body and joints, using information obtained in the user detection task. The input data for the gesture interpretation process are the position and orientation of these regions. This process will determine which gesture the user has carried out. Next, these gesture data are sent to the execution process, which ends the process by performing the action that has been specified, and so completing the feedback process. This is a very complex and challenging task, so we isolate sub problems to make it more tractable. Then we present in this paper the face and hands segmentation and tracking.

Our previous work, define a segmentation algorithm based in functional minimization procedure [10, 11] but this robust method is time consuming and non applicable to virtual reality application where the delay time must be very short (less 40 ms). So in this paper we propose a more simple segmentation algorithm and heuristic classification and tracking procedure based in a set of rules.

In the following section, we explain briefly the tracking method proposed including the skin color pixel detection and training. Also we present our hypothesis generation and applications and finally we conclude with some examples and visualization results in VRML format and H-Anim [12] avatar compliant. This work is an adapted version from a more computational cost version of [5, 10] improving the computational efficiency to apply in virtual reality environments.

2 Tracking of Face and Hands

The tracking algorithm is divided in two parts: skin-color pixel detection and data association. For each frame, first, we segment the skin-color pixels using a previous learned probabilistical skin-color model and a blob analysis. Next, we use a hypothesis based data association algorithm to label the skin-color detected pixels. This algorithm uses a simple prediction step to the state obtained in the previous frame to calculate the new hypothesis.

2.1 Skin-Color Pixel Detection

As we have explained before, we have another proposed method based in more solid mathematical background but are very expensive in computational cost. So we would like to define new systems that can be run in real time and oriented to virtual reality applications, in particular interactive workbench activities with no contact devices.
The assumption that color can be used as a cue to detect faces and hands has been proved in several publications [1,2,10,11]. Usually, it is necessary to model the actor's skin-color in a previous step. In our work we use only one image of the actor to build this model. The user selects manually the regions of the image that contains skin-color pixels, also this procedure can be done in a automatic way, we are working in this process. Next, we transform these pixels from the RGB-space to HSL-space to take the Hue and the Saturation values for each pixel, that is, the chroma information. These values for the selected pixels are the data samples used to learn the skin-color model:

$$x = (x_1, \dots, x_n),$$

where n is the number of samples and $\mathbf{x}_i = (\mathbf{h}_i, \mathbf{s}_i)$. We have proved several statistical models and finally the best results has been obtained using a Gaussian model:

$$\mu = \frac{1}{n}\sum_i \mathrm{x}_i, \qquad \Sigma = \frac{1}{n}\sum_i (\mathrm{x}_i - \mu_i)\cdot(\mathrm{x}_i - \mu_i).^T \tag{1}$$

Once the skin-color model is built we can calculate the probability that a pixel is skin colored:

$$P(x) = \frac{1}{\sqrt{(2\pi)^2 |\Sigma|}} e^{\frac{1}{2}(x-\mu)(x-\mu)^T}. \tag{2}$$

Then, in order to detect the skin-color pixels in each frame of the sequence we compute the probability for all the image pixels. In Fig. 1 can be show some results of this process, it can be seen how this model performs well in several environmental conditions.

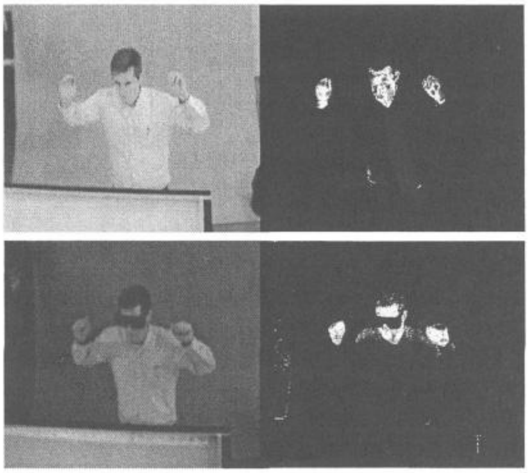


Fig. 1. Examples of skin-color probabilities using the model proposed.In the first row we show one frame for a sequence with normal illumination, and in second row with a proper illumination to correctly see the workbench screens

However, it is necessary to refine the results of pixel probabilities to obtain an image with only the pixels belonging to the interesting blobs: the face and the hands. Therefore, we define a posterior hysteresis process to obtain the skin-color blobs. All the image pixels with probability $P>T_{max}$ are considered as being skin colored. These pixels constitute the seeds of potential blobs. More specifically, image pixels with probability $P>T_{min}$, where $T_{min}< T_{max}$, that are immediate neighbours of skin-color pixels are recursively added to each blob. We actually use $T_{min}=0.05$ and $T_{max}=0.15$. In Fig. 2 we show the final results of skin-color detection.

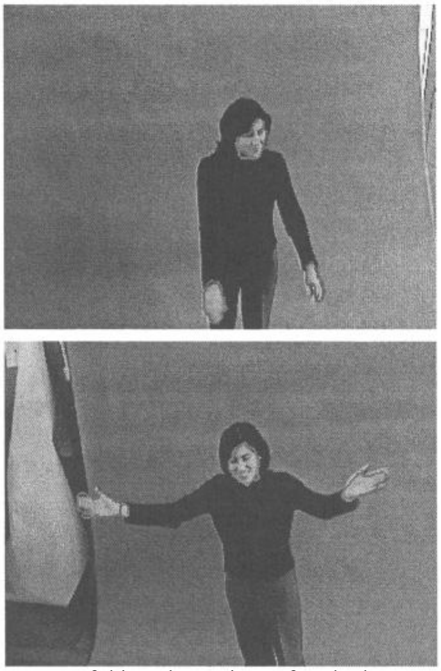


Fig. 2. Contours of skin color regions after the hysteresis process

The final step in the skin-color pixel detection is the computation of the attributes of each blob (centroid and size) that will be used in the tracking process to represent the state of actor's face and hands.

2.2 Data Association

At each frame, we represent the state of the face and hands using a set of *hypothesis*. Each hypothesis is composed by the position and size of the actor's face or hand (from now, limb). The proposed method to track a limb operates as follows: for the frame of time t the aim is to associate the skin-color blobs with the limb hypothesis in time *t-1*. The objective of this association is to propagate in time the labels that represent the face, the left hand and the right hand, and to detect when a limb appears in the scene and disappears from the field of view.

We assume that at tune *t*, *M* blobs have been detected using the skin-color model,

$$B=\{b_1,\ldots,b_j,\ldots,b_M\}.$$

Each blob b_j, corresponds to a set of connected skin-color pixels. Note that a blob may correspond to one of many limbs. As an example, two crossing hands are two different limbs that appear as one blob when one occludes the other. Let *N* the number of visible limbs present in the viewed scene at time *t* (*N<4*). We assume that an ellipse can approximate the spatial distribution of the pixels.
Then, the state of a limb *i* can be represented by:

$$h_i = \left(c_{x_i}, c_{y_i}, \alpha_i, \beta_i\right),$$

where $\left(c_{x_i}, c_{y_i}\right)$ is its center and (α_i, β_i) are the lengths of semiaxis. The union of limb states is denoted by:

$$H = \{h_i\}, \quad i \leq 3.$$

Tracking amounts to determining the relation between limb states (*h*$_i$) and observations (*b*$_j$) in time. The first subproblem can be present when a new limb appears in the field of view. In order to cope with this problem, we define the distance $D(p,h)$ of a pixel $p = (x,y)$ from a ellipse (i.e., limb state) as follows:

$$D(p,h) = \sqrt{v \cdot v^T}, \tag{3}$$

where

$$v = \left(\frac{x - c_x}{\alpha}, \frac{y - c_y}{\beta}\right). \tag{4}$$

From the definition of D(p, h) it turns out that its value is less than 1 if *p* is inside ellipse *h,* and greater than 1 if it is outside. Considering a state *h* and a point *p* belonging to a blob *b,* if distance is less than 1, we conclude that the blob *b* support the existence of the part hypothesis *h* and that part hypothesis *h* predicts blob *b.* Now, if a new part appears in the scene implies that none of the existing hypothesis predicts the existence of the corresponding blob *b*:

$$\forall p \in b, \quad \min_{h \in H}\{D(p,h)\} > 1. \tag{5}$$

The Eq.5 describes a blob with empty intersection with all ellipses of the existing limb hypothesis. Algorithmically, at each time t, all detected blobs are tested against the criterion of Eq.5. If a blob not belongs to any hypothesis, a new limb hypothesis is created and their corresponding parameters correspond to the blob parameters.

After appearing limbs have been detected, all the remaining blobs must support the existence of limb hypothesis. The main task of the tracking algorithm is to associate blobs to limb hypothesis. We use two rules to make this association:

- **Rule 1:** if a pixel *p* of a blob is inside of a limb hypothesis then this pixel is labeled with the hypothesis number. Formally

$$R_1 = \{p \in B \mid D(p,h) < 1\}. \tag{6}$$

- **Rule 2:** if a pixel *p* of a blob is outside of all the ellipses, then it is labeled to the hypothesis number that is closer to it. Formally

$$R_2 = \left\{p \in B \,\middle|\, D(p,h) = \min_{k \in H}\{D(p,k)\}\right\}.. \tag{7}$$

These two rules permit the treatment of limbs occlusion if pixels belonging to a blob can be labeled with more than one hypothesis number using the rule 2, see Fig. 3.

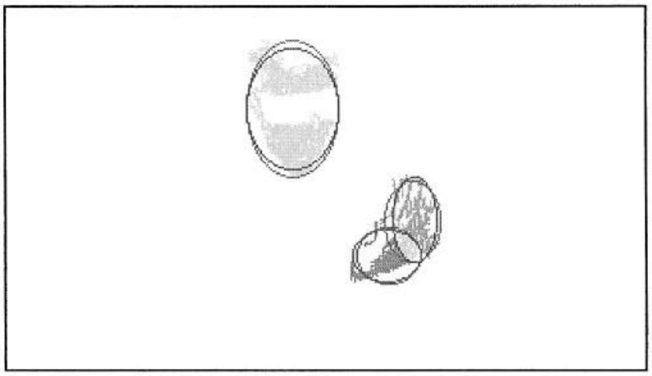


Fig. 3. Occlusion treatment using multiple labeling and rule 2. Object hypothesis are depicted using ellipses

Another interesting case can happen when finish the occlusion and then a hypothesis is supported by more than one blob (split case). In this case, the hypothesis is assigned to the blob with which it shares the largest number of pixels. Finally, a limb hypothesis should be removed either when the limb moves out of scene. Following our algorithm, a hypothesis is removed when:

$$\forall p \in B, D(p,h) > 1 \ . \tag{8}$$

To conclude, once all the above case have been treated for a frame, the resulting limb hypothesis are used to maintain the limb states, that is to know the position and size of head and hands. Once the state has been estimated, they are propagated in time to the next frame using a linear scheme of prediction:

$$\begin{aligned} \hat{C}_i(t) &= C_i(t) + \Delta C_i(t), \\ \Delta C_i(t) &= C_i(t) - C_i(t-1), \end{aligned} \tag{9}$$

where $C_i(t) = \left(c_{x_i}, c_{y_i}\right)$. The above equations say that a part wills maintain the same velocity on the image plane. Some experimental results showed in Fig. 4.

2.3 3D Position Estimation

At the initialization step a calibration process is carried out using OpenCV library functions and a check board as calibrator object. The calibration process is showed in Fig. 5.

Finally, we use the calibration parameters computed and the state of each limb, see Fig. 6, in order to estimate their 3D positions.

The complete procedure can be seen in Fig. 7.

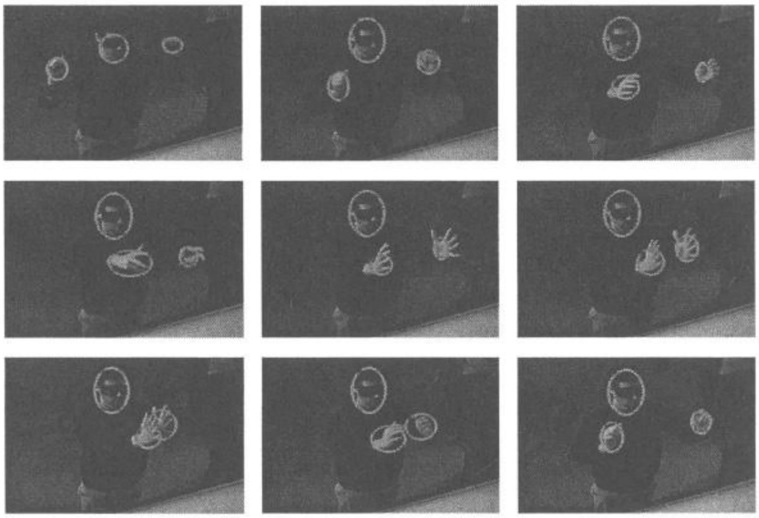


Fig. 4. Experimental results of 2D-Tracking of head and hands

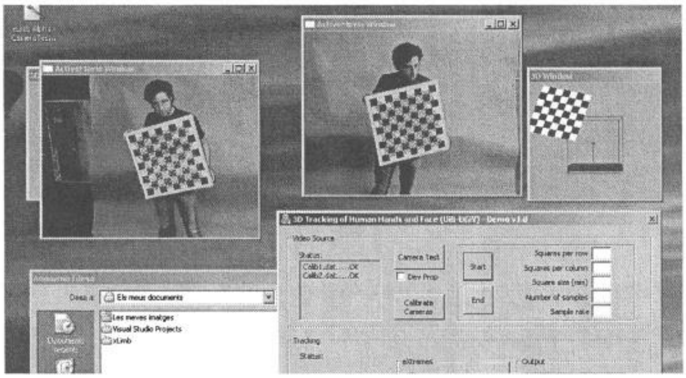


Fig. 5. Calibration process in the initialization phase

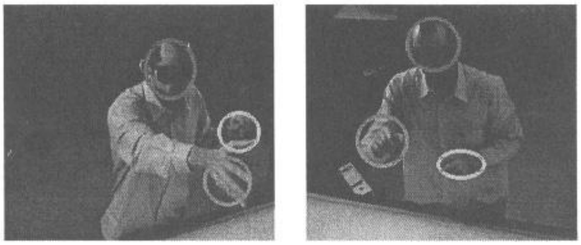


Fig. 6. Extreme limbs states used for 3D estimation in a workbench session

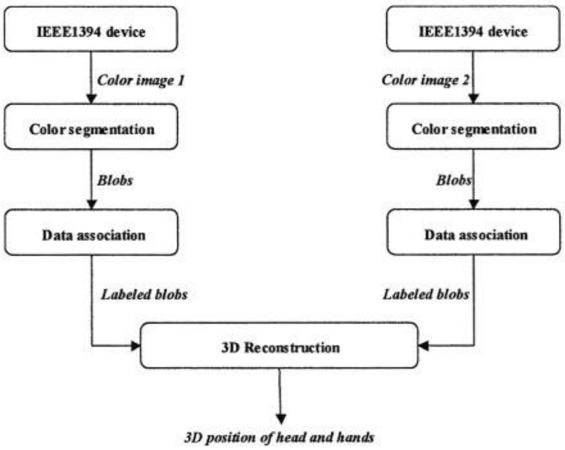


Fig. 7. Complete procedure: color segmentation, data association and 3D reconstruction

3 Visualization Using VRML and H-Anim Avatar on Workbench

Hands and head tracking system obtain 2D coordinates from two stereo cameras. Due to a previous calibration process, is able to compute 3D position. This information is needed to display the data in real time on a Barco Consul Workbench. That's mean that a human virtual avatar can be reproduced in a remote workbench. The final objective of this process is to recover in high precision the 3D position and orientation of the hands and face in the interactive applications. In the human model representation we use the H-Anim standard that's mean that we can collaborate with standard VRML models.

At the moment we use IEEE1394 digital videocameras and high quality workbench but the system must be portable in near future to low cost systems (web cams, and domestic virtual reality environments).

3D position is computed for every blob, projecting the centroid of the blob on each image to infinity and compute 3D coordinate as the nearest point to this two lines.

Hands and head are modeled with a VRML file, extracted from a H-Anim humanoid, see two different reconstruted views in Fig. 8.

In the web site http://dmi.uib.es/research/GV/amdoResults2004 you can view and download some video examples of captured data and VRML demos. Also we can combine the synthetic avatar with real human in workbench in real time, as we can see in Fig 9.

At the moment only 3d position is really computed, so we must to consider the 3D orientation and check the precision of this reconstruction against commercial tracking systems based in ultrasound technology. This comparison must be evaluated in relation commercial cost/precision results.

Also we plan to reconstruct the fingers positions to understand the actions that the user plans to do in relation with the interactive process in workbench. So a more carefully analysis must be done [9].

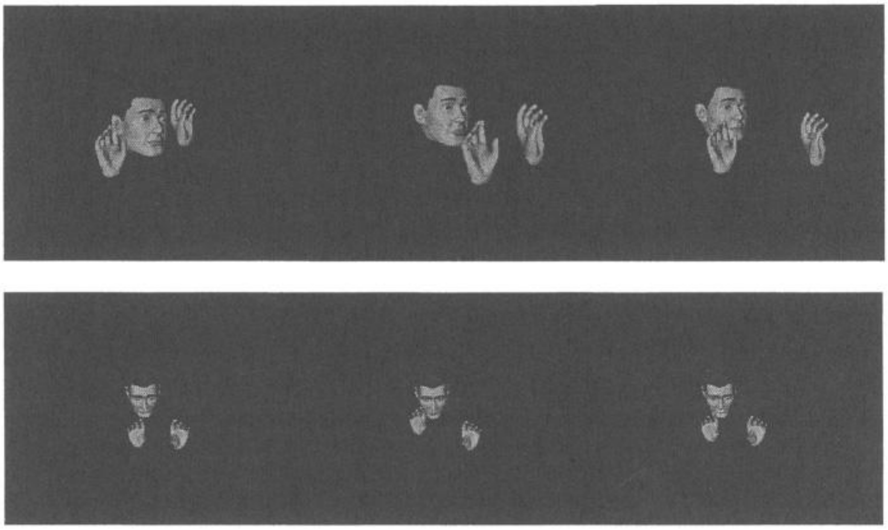


Fig 8. 3D Reconstruction in VRML

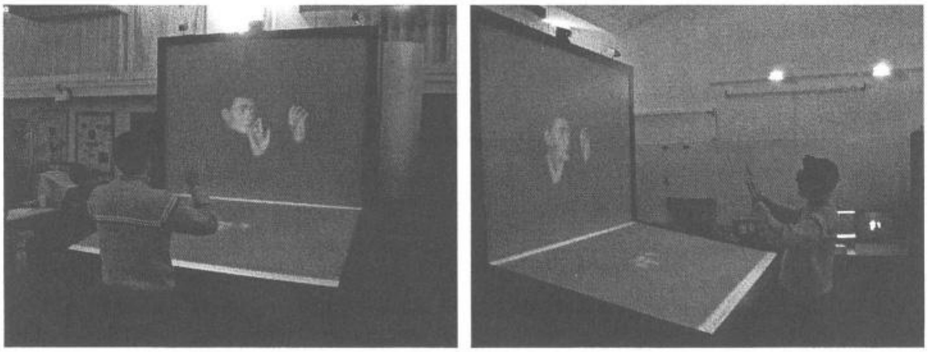


Fig 9. An example of HCI in virtual reality applications

4 Conclusions and Future Work

In this paper we have proposed a new system for 3D tracking of human extreme limbs, hands and face, for HCI in real time. Moreover, it analyses the user to determine parameters that will be useful for tracking process that‘s include an heuristic method based in a well define set of rules. The pixel segmentation process is based on the colour pixel classification hypothesis based data association algorithm. Besides, the process is carried out in real time. The software implementation is efficient and OOP. The result of this process is the tracking and reconstruction of face and hands in 3D to be able a human computer interaction system for virtual reality environments.

It remains as future work to do tracking of interesting body parts and to interpret movements in order to carry out action recognition that the user is performing. So we must to improve the precision results and more precise primitives must be defined to track fingers and face parts to know exactly eyes directions and gaze orientation.

Acknowledgements

The projects TIC2003-0931 and TIC2002-10743-E of MCYT Spanish Government and European Project HUMODAN 2001-32202 from UE V Program-IST have subsidized this work. Also we would like to express our gratitude to all members of Computer Graphics and Vision group at UIB by their collaboration.

References

[1] Bradski G.R.; “Computer video face tracking for use in a perceptual user interface”. Intel Technology Journal, Q2’98, 1998

[2] Comaniciu, D.; Ramesh, V.; “Robust detection and tracking of human faces with an active camera”. Proceedings of the Third IEEE International Workshop on Visual Surveillance, 2000; Page(s): 11-18.

[3] Lars Bretznerl, 2 Ivan Laptev,1 Tony Lindeberg1. Hand Gesture Recognition using Multi-Scale Colour Features, Hierarchical Models and Particle Filtering, Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR.02), 0-7695-1602-5/02 © 2002 IEEE

[4] H.D. Cheng, X.H. Jiang, Y. Sun, JinGli Wang “Color Image Segmentation: Advances and Prospects”, Journal of Pattern Recognition 34, (2001), pp. 2259-2281

[5] M. Gonzalez “Segmentación de imágenes en Color por método variacional”. Proc. Del XIV C.E.D.Y.A. y IV C.M.A. pp 287-288, 1995.

[6] I. Haritaoglu, “W4: Real-Time Surveillance of People and Their Activities” IEEE. Transactions on Pattern Analysis and Machine Intelligence, vol 22 No8, pp 809-830, 2000

[7] H. Sidenbladh, M.J. Black and D.J. Fleet “Stochastic Tracking of 3D Human Fig.s Using 2D Image Motion” ECCV 2000.

[8] Kenji Okay Yoichi Satoy Hideki Koikez, Real-time Tracking of Multiple Fingertips and Gesture Recognition for Augmented Desk Interface Systems, Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (FGR.02), 0-7695-1602-5/02, 2002 IEEE

[9] Koichi Ogawara, Kentaro Hashimoto, Jun Takamatsu, Katsushi Ikeuchi, Grasp Recognition using a 3D Articulated Model and Infrared Images, Institute of Industrial Science, Univ. of Tokyo, Tokyo, Japan, Institute of Industrial Science, Univ. of Tokyo, Tokyo, Japan, Fuji Xerox Information Systems Co.,Ltd. Tokyo, 150–0031, Japan

[10] J.M. Buades, M. Gonzalez, F.J. Perales. “A New Method for Detection and Initial Pose Estimation based on Mumford-Shah Segmentation Functional”. IbPRIA 2003. Port d’Andratx. Spain. June 2003. pp 117-125, LNCS 2652.

[11] J.M. Buades, M. Gonzalez, F.J. Perales. “Face and Hands Segmentation in Color Images and Initial Matching” International Workshop on Computer Vision and Image Analysis. Palmas de Gran Canaria. Dec. 2003. pp 43-48.

[12] HANIM 1.1 Compliant VRML97. http://ece.uwaterloo.ca/~h-anim/index.html

Author Index

This page intentionally left blank

This page intentionally left blank

This page intentionally left blank