Lecture Notes in Computer Science 4796

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Michael Lew   Nicu Sebe   Thomas S. Huang
Erwin M. Bakker (Eds.)

Human–Computer Interaction

IEEE International Workshop, HCI 2007
Rio de Janeiro, Brazil, October 20, 2007
Proceedings

Springer

Volume Editors

Michael Lew
Erwin M. Bakker
Leiden University
LIACS Media Lab
Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
E-mail: {mlew, erwin}@liacs.nl

Nicu Sebe
University of Amsterdam
Faculty of Science
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
E-mail: nicu@science.uva.nl

Thomas S. Huang
University of Illinois at Urbana-Champaign
Beckman Institute for Advanced Science and Technology
405 North Mathews Avenue, Urbana, IL 61801, USA
E-mail: huang@ifp.uiuc.edu

Library of Congress Control Number: 2007937397

CR Subject Classification (1998): H.5, H.5.2, I.2.6, H.3, H.2.4

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN	0302-9743
ISBN-10	3-540-75772-4 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-75772-6 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12176533 06/3180 5 4 3 2 1 0

Preface

Welcome to the 4th IEEE International Workshop on Human – Computer Interaction, HCI 2007. Human – computer interaction (HCI) is one of the foremost challenges of our society. New paradigms for interacting with computers are being developed that will define the 21st century and enable the world to communicate and interact effortlessly and intuitively.

The goal of this workshop is to bring together researchers whose work is related to human – computer interaction, share novel ideas, and stimulate new research directions. This year we received 113 submissions from more than 18 countries. Based upon the double-blind reviews of the program committee, 15 papers were accepted for oral presentation and publication in the proceedings.

We would like to thank all of the members of the Program Committee. Their reviews of the submissions as always played a pivotal role in the quality of the workshop. Furthermore, we would like to acknowledge our sponsors and supporters, the Leiden Institute of Advanced Computer Science at Leiden University, the Faculty of Science at the University of Amsterdam, the Beckman Institute at the University of Illinois at Urbana-Champaign, BSIK/BRICKS/FOCUS, the Netherlands National Science Foundation (NWO) and IEEE.

Finally, special thanks go to Bart Thomee, who met the myriad and unexpected challenges of organizing this workshop with generous and tireless dedication.

August 2007

Michael Lew
Nicu Sebe

IEEE International Workshop on Human – Computer Interaction 2007 Organization

Organizing Committee

General Co-chairs	Michael Lew (LIACS Media Lab, Leiden University, The Netherlands) Nicu Sebe (Faculty of Science, University of Amsterdam, The Netherlands)
Program Chair	Erwin M. Bakker (LIACS Media Lab, Leiden University, The Netherlands)
Executive Chair	Thomas S. Huang (Beckman Institute, University of Illinois at Urbana-Champaign, USA)

Program Committee

Kiyo Aizawa	University of Tokyo, Japan
Erwin M. Bakker	Leiden University, The Netherlands
Margrit Betke	Boston University, USA
Alberto del Bimbo	University of Florence, Italy
Edward Chang	Google Research, China
Jeffrey Cohn	University of Pittsburgh, USA
Daniel Gatica-Perez	IDIAP Research, Switzerland
Thomas Huang	Univ. of Illinois at Urbana-Champaign, USA
Alejandro Jaimes	IDIAP Research, Switzerland
Rana Kaliouby	MIT, USA
Ashish Kapoor	Microsoft Research, USA
Stefan Kopp	Univ. Bielefeld, Germany
Michael Lew	Leiden University, The Netherlands
Qiong Liu	FXPAL, USA
Maja Pantic	Imperial College, UK
Ioannis Patras	Queen Mary University, UK
Nicu Sebe	University of Amsterdam, The Netherlands
Peter Sun	Nanjing University, China
Qi Tian	University of Texas at San Antonio, USA
Matthew Turk	University of California, Santa Barbara, USA
Jordi Vitria	Universitat Autonoma de Barcelona, Spain
Guangyou Xu	Tsinghua University, China
Ming-Hsuan Yang	Honda Research Labs, USA
Xiang (Sean) Zhou	Siemens Research, USA

Sponsoring Institutions and Supporters

Leiden Institute of Advanced Computer Science, Leiden University,
The Netherlands
University of Amsterdam, The Netherlands
University of Illinois at Urbana-Champaign, USA
BSIK/BRICKS/FOCUS
Netherlands National Science Foundation (NWO)
IEEE

Table of Contents

Human-Computer Intelligent Interaction: A Survey

Michael Lew[1], Erwin M. Bakker[1], Nicu Sebe[2], and Thomas S. Huang[3]

[1] LIACS Media Lab, Leiden University, The Netherlands
[2] ISIS Group, University of Amsterdam, The Netherlands
[3] Beckman Institute, University of Illinois at Urbana-Champaign, USA

Abstract. Human-computer interaction (HCI) is one of the foremost challenges of our society. New paradigms for interacting with computers are being developed which will define the 21st century and enable the world to communicate and interact effortlessly and intuitively. In this short survey, we explain the major research clusters comprising the state of the art, and indicate promising future research directions.

1 Introduction

Historians will refer to our time as the Age of Information. While information is indeed important, the machine interface, the interactive process between humans and computers will define the 21st century. In this short survey, we are primarily interested in human-computer *intelligent* interaction as opposed to *simple* human-computer interaction. When a user types a document at a word processor, there is a form of simple human-computer interaction - the human types and the computer shows the human the formatted keystrokes composing the document. However, this would not be considered *intelligent* interaction because the computer is not performing any intelligent processing on the keystrokes; there is no meaning extracted from the user's actions. The computer is simply mirroring the user's actions.

From a research perspective, we are interested in *intelligent* interaction where the computer understands the *meaning* of the message of the user and also the *context* of the message. As an example, interaction between humans is typically performed using speech and body gestures. The speech carries the meaning of the message but often not the context. Is the person happy or sad? Is the person serious or joking? To understand the context, it is also necessary to grasp the facial and body gestures. If we are to have truly intuitive communication, computers will need to have their own sense of vision and speech to naturally fit into the world of humans.

How can we achieve synergism between humans and computers? The term "Human-Centered Computing" is used to emphasize the fact that although all existing information systems were designed with human users in mind, many of them are far from being user friendly. Research in face detection and expression recognition [1, 2, 19] will someday improve the human machine synergism. Information systems are ubiquitous in all human endeavors including scientific, medical, military, transportation, and consumer. Individual users use them for learning, searching for

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 1–5, 2007.

information (including data mining), doing research (including visual computing), and authoring. Multiple users (groups of users, and groups of groups of users) use them for communication and collaboration. And either single or multiple users use them for entertainment.

2 Current Research

In Human-Computer Intelligent Interaction, the three dominant clusters of research are currently in the areas of face analysis, body analysis, and complete systems which integrate multiple cues or modalities to give intuitive interaction capabilities to computers. Face and human body analysis includes modeling the face or body and tracking the motion and location. They are considered fundamental enabling technologies towards HCI.

Understanding the dynamics of the human face is certainly one of the major research challenges. The potential of the following work is toward detecting and recognizing emotional states and improving computer to human communication. Valstar and Pantic[3] propose a hybrid technique where a Support Vector Machine (SVM) is used to segment the facial movements into temporal units. Then a Hidden Markov Model is used for classifying temporal actions. Chang et al. [4] propose a framework for using multiple cameras in pose and gaze estimation using a particle system approach.

Another major research challenge is human body analysis. The following research has the potential to endow computers to see where humans are, what their gestures and motions are, and track them over time. Oerlemans, et al.[5] propose a tracking system which uses the multidimensional maximum likelihood approach toward detecting and identifying moving objects in video. Their system has the additional novel aspect of allowing the user to interactively give feedback as to whether segmented objects are part of the background or foreground. Cooper and Bowden[6] address the problem of large lexicon sign language understanding. The method detects patterns of movement in two stages involving viseme classifiers and Markov chains. Angelopoulou et al.[8] are able to model and track nonrigid objects using a growing neural gas network. Jung, et al.[9] use a multi-cue method to recognize human gestures using AdaBoost and Fisher discriminant analysis. Chu and Nevatia[10] are able to track a 3D model of the human body from silhouettes using a particle filtering approach and infra-red cameras.

In addition to novel face and human body analysis, current research is creating systems which are delving into the complete human-computer interaction, thereby endowing computers with new important functionality. State[11] had created a system where a virtual human can maintain exact eye contact with the human user thereby significantly improving the perception of immersion. Rajko and Qian[12] propose an automatic kinematic model building method for optical motion capture using a Markov random field framework. Vural, et al.[13] created a system for detecting driver drowsiness based on facial action units and classified using AdaBoost and multinomial ridge regression. Thomee, et al.[14] used an artificial imagination approach to allow the computer to create example images to improve relevance feedback in image retrieval. Barreto, et al.[15] use physiological responses and pupil dilation to recognize stress using an SVM.

In the area of interfaces, researchers are actively developing new systems. Hua, et al.[16] presented a system for mobile devices for a visual motion perceptual interface. A tabletop interface using camera detected finger interactions was proposed by Song, et. al.[17]. Interaction with a wall sized display was presented by Stødle, et al. [18], whereby the user could interact with the system using arm and hand movements.

3 Future Research Directions

As the current research indicates, we are making rapid progress in face analysis, human body analysis and creating the early generation of complete human-computer interactive systems. Several fundamental directions for the future include (1) User Understanding - learn more about the user [23] and create systems which will adapt appropriately to the user's needs; (2) Authentic emotion recognition - be able to reliably recognize the authentic emotional states of humans [19]; (3) Education and knowledge - develop new paradigms which improve education, learning, and the search for knowledge [20]; (4) New features and similarity measures [7] - for example, development of new texture models [21,22] focusing on HCI tasks such as face and body tracking integration with color and temporal spaces; and (5) Benchmarking in HCI - in many areas such as body tracking there are negligible ground truth sets which would be considered scientifically definitive. We need to decide on the most important HCI tasks and collectively create credible ground truth sets for evaluating and improving our systems.

Acknowledgments

We would like to thank Leiden University, University of Amsterdam, University of Illinois at Urbana-Champaign, the Dutch National Science Foundation (NWO), and the BSIK/BRICKS research funding programs for their support of our work.

References

1. Cohen, I., Sebe, N., Garg, A., Lew, M.S., Huang, T.S.: Facial expression recognition from video sequences. In: ICME. Proceedings of the IEEE International Conference Multimedia and Expo, Lausanne, Switzerland, vol. I, pp. 641–644. IEEE Computer Society Press, Los Alamitos (2002)

2. Lew, M.S.: Information theoretic view-based and modular face detection. In: Proceedings of the IEEE Face and Gesture Recognition conference, Killington, VT, pp. 198–203. IEEE Computer Society Press, Los Alamitos (1996)
3. Valstar, M.F., Pantic, M.: Combined Support Vector Machines and Hidden Markov Models for Modeling Facial Action Temporal Dynamics. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 118–127. Springer, Heidelberg (2007)
4. Chang, C.-C., Wu, C., Aghajan, H.: Pose and Gaze Estimation in Multi-Camera Networks for Non-Restrictive HCI. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 128–137. Springer, Heidelberg (2007)
5. Oerlemans, A., Thomee, B.: Interactive Feedback for Video Tracking Using Hybird Maximum Likelihood Similarity Measure. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 79–87. Springer, Heidelberg (2007)
6. Cooper, H., Bowden, R.: Large Lexicon Detection of Sign Language. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 88–97. Springer, Heidelberg (2007)
7. Sebe, N., Lew, M.S., Huijsmans, N.: Toward Improved Ranking Metrics. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1132–1143 (October 2000)
8. Angelopoulou, A., Psarrou, A., Gupta, G., Garcia Rodríguez, J.: Nonparametric Modelling and Tracking with Active-GNG. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 98–107. Springer, Heidelberg (2007)
9. Jung, S., Guo, Y., Sawhney, H., Kumar, R.: Multiple Cue Integrated Action Detection. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 108–117. Springer, Heidelberg (2007)
10. Chu, C.-W., Nevatia, R.: Real Time Body Pose Tracking in an Immersive Training Environment. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 146–156. Springer, Heidelberg (2007)
11. State, A.: Exact Eye Contact with Virtual Humans. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 138–145. Springer, Heidelberg (2007)
12. Rajko, S., Qian, G.: Real-time Automatic Kinematic Model Building for Optical Motion Capture Using a Markov Random Field. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 69–78. Springer, Heidelberg (2007)
13. Vural, E., Cetin, M., Ercil, A., Littlewort, G., Bartlett, M., Movellan, J.: Drowsy Driver Detection Through Facial Movement Analysis. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 6–18. Springer, Heidelberg (2007)
14. Thomee, B., Huiskes, M.J., Bakker, E., Lew, M.: An Artificial Imagination for Interactive Search. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 19–28. Springer, Heidelberg (2007)
15. Barreto, A., Zhai, J., Adjouadi, M.: Non-intrusive Physiological Monitoring for Automated Stress Detection in Human-Computer Interaction. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 29–38. Springer, Heidelberg (2007)
16. Hua, G., Yang, T.-Y., Vasireddy, S.: PEYE: Toward a Visual Motion based Perceptual Interface for Mobile Devices. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 39–48. Springer, Heidelberg (2007)
17. Song, P., Winkler, S., Gilani, S.O.: Vision-based Projected Tabletop Interface for Finger Interactions. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 49–58. Springer, Heidelberg (2007)

18. Stødle, D., Bjørndalen, J., Anshus, O.: A System for Hybrid Vision- and Sound-Based Interaction with Distal and Proximal Targets on Wall-Sized, High-Resolution Tiled Displays. In: Lew, M., Sebe, N., Huang, T.S., Bakker, E.M. (eds.) HCI 2007. LNCS, vol. 4796, pp. 59–68. Springer, Heidelberg (2007)
19. Sebe, N., Lew, M.S., Cohen, I., Sun, Y., Gevers, T., Huang, T.S.: Authentic Facial Expression Analysis. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition, Seoul, Korea, pp. 517–522 (May 2004)
20. Lew, M.S., Sebe, N., Djeraba, C., Jain, R.: Content-based Multimedia Information Retrieval: State-of-the-art and Challenges. ACM Transactions on Multimedia Computing, Communication, and Applications 2(1), 1–19 (2006)
21. Sebe, N., Lew, M.S.: Wavelet Based Texture Classification. In: Proceedings of the 15th International Conference on Pattern Recognition, Barcelona, Spain, vol. III, pp. 959–962 (2000)
22. Sidenbladh, H., Black, M.J., Sigal, L.: Implicit Probabilistic Models of Human Motion for Synthesis and Tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2352, pp. 784–800. Springer, Heidelberg (2002)
23. Pantic, M., Pentland, A., Nijholt, A., Huang, T.S.: Human Computing and Machine Understanding of Human Behavior: A Survey. In: ICMI 2006 and IJCAI International Workshops, Banff, Canada, Hyderabad, India, November 3, 2006, January 6, 2007. LNCS, vol. 4451, pp. 47–71. Springer, Heidelberg (2007)

Drowsy Driver Detection Through Facial Movement Analysis

Esra Vural[1,2], Mujdat Cetin[1], Aytul Ercil[1], Gwen Littlewort[2], Marian Bartlett[2], and Javier Movellan[2]

[1] Sabanci University
Faculty of
Engineering and Natural Sciences
Orhanli, Istanbul
[2] University of California San Diego
Institute of
Neural Computation
La Jolla, San Diego

Abstract. The advance of computing technology has provided the means for building intelligent vehicle systems. Drowsy driver detection system is one of the potential applications of intelligent vehicle systems. Previous approaches to drowsiness detection primarily make preassumptions about the relevant behavior, focusing on blink rate, eye closure, and yawning. Here we employ machine learning to datamine actual human behavior during drowsiness episodes. Automatic classifiers for 30 facial actions from the Facial Action Coding system were developed using machine learning on a separate database of spontaneous expressions. These facial actions include blinking and yawn motions, as well as a number of other facial movements. In addition, head motion was collected through automatic eye tracking and an accelerometer. These measures were passed to learning-based classifiers such as Adaboost and multinomial ridge regression. The system was able to predict sleep and crash episodes during a driving computer game with 96% accuracy within subjects and above 90% accuracy across subjects. This is the highest prediction rate reported to date for detecting real drowsiness. Moreover, the analysis revealed new information about human behavior during drowsy driving.

1 Introduction

In recent years, there has been growing interest in intelligent vehicles. A notable initiative on intelligent vehicles was created by the U.S. Department of Transportation with the mission of prevention of highway crashes [1]. The ongoing intelligent vehicle research will revolutionize the way vehicles and drivers interact in the future.

The US National Highway Traffic Safety Administration estimates that in the US alone approximately 100,000 crashes each year are caused primarily by driver drowsiness or fatigue [2]. Thus incorporating automatic driver fatigue detection mechanism into vehicles may help prevent many accidents.

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 6–18, 2007.

One can use a number of different techniques for analyzing driver exhaustion. One set of techniques places sensors on standard vehicle components, e.g., steering wheel, gas pedal, and analyzes the signals sent by these sensors to detect drowsiness [3]. It is important for such techniques to be adapted to the driver, since Abut and his colleagues note that there are noticeable differences among drivers in the way they use the gas pedal [4].

A second set of techniques focuses on measurement of physiological signals such as heart rate, pulse rate, and Electroencephalography (EEG) [5]. It has been reported by researchers that as the alertness level decreases EEG power of the alpha and theta bands increases [6]. Hence providing indicators of drowsiness. However this method has drawbacks in terms of practicality since it requires a person to wear an EEG cap while driving.

A third set of solutions focuses on computer vision systems that can detect and recognize the facial motion and appearance changes occurring during drowsiness [7] [8]. The advantage of computer vision techniques is that they are non-invasive, and thus are more amenable to use by the general public. There are some significant previous studies about drowsiness detection using computer vision techniques . Most of the published research on computer vision approaches to detection of fatigue has focused on the analysis of blinks and head movements. However the effect of drowsiness on other facial expressions have not been studied thoroughly. Recently Gu & Ji presented one of the first fatigue studies that incorporates certain facial expressions other than blinks. Their study feeds action unit information as an input to a dynamic bayesian network. The network was trained on subjects posing a state of fatigue [9]. The video segments were classified into three stages: inattention, yawn, or falling asleep. For predicting falling-asleep, head nods, blinks, nose wrinkles and eyelid tighteners were used.

Previous approaches to drowsiness detection primarily make pre-assumptions about the relevant behavior, focusing on blink rate, eye closure, and yawning. Here we employ machine learning methods to datamine actual human behavior during drowsiness episodes. The objective of this study is to discover what facial configurations are predictors of fatigue. In this study, facial motion was analyzed automatically from video using a fully automated facial expression analysis system based on the Facial Action Coding System (FACS) [10]. In addition to the output of the automatic FACS recognition system we also collected head motion data using an accelerometer placed on the subject's head, as well as steering wheel data.

2 Methods

2.1 Driving Task

Subjects played a driving video game on a windows machine using a steering wheel [1] and an open source multi-platform video game [2] (See Figure 1). The

[1] Thrustmaster® Ferrari Racing Wheel.

[2] The Open Racing Car Simulator(TORCS).

windows version of the video game was maintained such that at random times, a wind effect was applied that dragged the car to the right or left, forcing the subject to correct the position of the car. This type of manipulation had been found in the past to increase fatigue [11]. Driving speed was held constant. Four subjects performed the driving task over a three hour period beginning at midnight. During this time subjects fell asleep multiple times thus crashing their vehicles. Episodes in which the car left the road (crash) were recorded. Video of the subjects face was recorded using a DV camera for the entire 3 hour session.

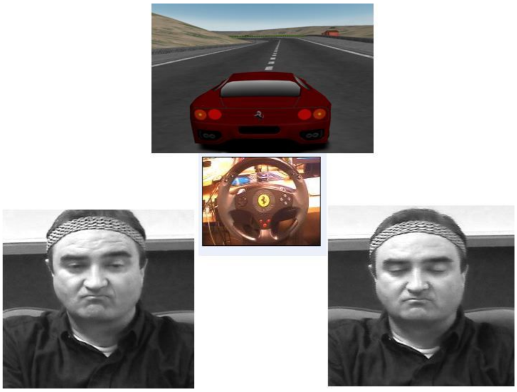


Fig. 1. Driving simulation task

2.2 Head Movement Measures

Head movement was measured using an accelerometer that has 3 degrees of freedom. This three dimensional accelerometer [3] has three one dimensional accelerometers mounted at right angles measuring accelerations in the range of 5g to +5g where g represents earth gravitational force.

2.3 Facial Action Classifiers

The facial action coding system (FACS) [12] is arguably the most widely used method for coding facial expressions in the behavioral sciences. The system describes facial expressions in terms of 46 component movements, which roughly correspond to the individual facial muscle movements. An example is shown in Figure 2. FACS provides an objective and comprehensive way to analyze expressions into elementary components, analogous to decomposition of speech into phonemes. Because it is comprehensive, FACS has proven useful for discovering facial movements that are indicative of cognitive and affective states. In this paper we investigate whether there are Action units (AUs) such as chin raises (AU17), nasolabial furrow deepeners(AU11), outer(AU2) and inner brow raises (AU1) that are predictive of the levels of drowsiness observed prior to the subjects falling sleep.

[3] Vernier®.

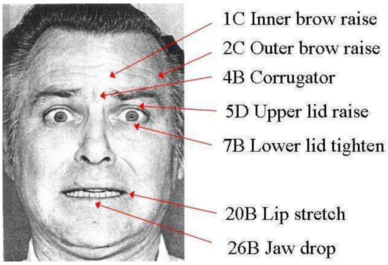


Fig. 2. Example facial action decomposition from the Facial Action Coding System

In previous work we presented a system, named CERT, for fully automated detection of facial actions from the facial action coding system [10]. The workflow of the system is based is summarized in Figure 3. We previously reported detection of 20 facial action units, with a mean of 93% correct detection under controlled posed conditions, and 75% correct for less controlled spontaneous expressions with head movements and speech.

For this project we used an improved version of CERT which was retrained on a larger dataset of spontaneous as well as posed examples. In addition, the system was trained to detect an additional 11 facial actions for a total of 31 (See Table 1). The facial action set includes blink (action unit 45), as well as facial actions involved in yawning (action units 26 and 27). The selection of this set of 31 out of 46 total facial actions was based on the availability of labeled training data.

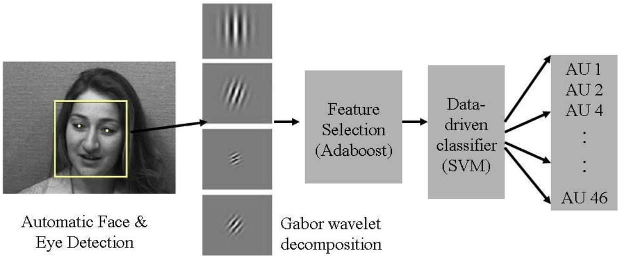


Fig. 3. Overview of fully automated facial action coding system

The facial action detection system was designed as follows: First faces and eyes are detected in real time using a system that employs boosting techniques in a generative framework [13]. The automatically detected faces are aligned based on the detected eye positions, cropped and scaled to a size of 96 × 96 pixels and then passed through a bank of Gabor filters. The system employs 72 Gabor spanning 9 spatial scales and 8 orientations. The outputs of these filters are normalized and then passed to a standard classifier. For this paper we employed

Table 1. Full set of action units used for predicting drowsiness

AU	Name
1	Inner Brow Raise
2	Outer Brow Raise
4	Brow Lowerer
5	Upper Lid Raise
6	Cheek Raise
7	Lids Tight
8	Lip Toward
9	Nose Wrinkle
10	Upper Lip Raiser
11	Nasolabial Furrow Deepener
12	Lip Corner Puller
13	Sharp Lip Puller
14	Dimpler
15	Lip Corner Depressor
16	Lower Lip Depress
17	Chin Raise
18	Lip Pucker
19	Tongue show
20	Lip Stretch
22	Lip Funneller
23	Lip Tightener
24	Lip Presser
25	Lips Part
26	Jaw Drop
27	Mouth Stretch
28	Lips Suck
30	Jaw Sideways
32	Bite
38	Nostril Dilate
39	Nostril Compress
45	Blink

support vector machines. One SVM was trained for each of the 31 facial actions, and it was trained to detect the facial action regardless of whether it occurred alone or in combination with other facial actions. The system output consists of a continuous value which is the distance to the separating hyperplane for each test frame of video. The system operates at about 6 frames per second on a Mac G5 dual processor with 2.5 ghz processing speed.

Facial expression training data. The training data for the facial action classifiers came from two posed datasets and one dataset of spontaneous expressions. The facial expressions in each dataset were FACS coded by certified FACS coders. The first posed datasets was the Cohn-Kanade DFAT-504 dataset [14]. This dataset consists of 100 university students who were instructed by an experimenter to perform a series of 23 facial displays, including expressions of seven basic emotions. The second posed dataset consisted of directed facial actions from 24 subjects collected by Ekman and Hager. Subjects were instructed by a FACS expert on the display of individual facial actions and action combinations, and they practiced with a mirror. The resulting video was verified for AU content by two certified FACS coders. The spontaneous expression dataset consisted of a set of 33 subjects collected by Mark Frank at Rutgers University. These subjects underwent an interview about political opinions on which they felt strongly. Two minutes of each subject were FACS coded. The total training set consisted of 6000 examples, 2000 from posed databases and 4000 from the spontaneous set.

3 Results

Subject data was partitioned into drowsy (non-alert) and alert states as follows. The one minute preceding a sleep episode or a crash was identified as a non-alert state. There was a mean of 24 non-alert episodes with a minimum of 9 and a maximum of 35. Fourteen alert segments for each subject were collected from the first 20 minutes of the driving task.[4] Our initial analysis focused on drowsiness prediction within-subjects.

3.1 Facial Action Signals

The output of the facial action detector consisted of a continuous value for each frame which was the distance to the separating hyperplane, i.e., the margin. Histograms for two of the action units in alert and non-alert states are shown in Figure 4. The area under the ROC (A') was computed for the outputs of each facial action detector to see to what degree the alert and non-alert output distributions were separated. The A' measure is derived from signal detection theory and characterizes the discriminative capacity of the signal, independent of decision threshold. A' can be interpreted as equivalent to the theoretical maximum percent correct achievable with the information provided by the system when using a 2-Alternative Forced Choice testing paradigm. Table 2 shows the actions with the highest A' for each subject. As expected, the blink/eye closure measure was overall the most discriminative for most subjects. However note that for Subject 2, the outer brow raise (Action Unit 2) was the most discriminative.

3.2 Drowsiness Prediction

The facial action outputs were passed to a classifier for predicting drowsiness based on the automatically detected facial behavior. Two learning-based classifiers, Adaboost and multinomial ridge regression are compared. Within-subject prediction of drowsiness and across-subject (subject independent) prediction of drowsiness were both tested.

[4] Several of the drivers became drowsy very quickly which prevented extraction of more alert segments.

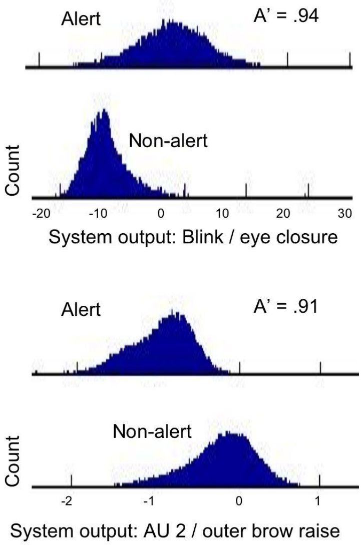


Fig. 4. Histograms for blink and Action Unit 2 in alert and non-alert states. A' is area under the ROC

Within Subject Drowsiness Prediction

For the within-subject prediction, 80% of the alert and non-alert episodes were used for training and the other 20% were reserved for testing. This resulted in a mean of 19 non-alert and 11 alert episodes for training, and 5 non-alert and 3 alert episodes for testing per subject.

The weak learners for the Adaboost classifier consisted of each of the 30 Facial Action detectors. The classifier was trained to predict alert or non-alert from each frame of video. There was a mean of 43,200 training samples, $(24+11)\times 60\times 30$, and 1440 testing samples, $(5+3)\times 60\times 30$, for each subject. On each training iteration, Adaboost selected the facial action detector that minimized prediction error given the previously selected detectors. Adaboost obtained 92% correct accuracy for predicting driver drowsiness based on the facial behavior.

Classification with Adaboost was compared to that using multinomial ridge regression (MLR). Performance with MLR was similar, obtaining 94% correct prediction of drowsy states. The facial actions that were most highly weighted by MLR also tended to be the facial actions selected by Adaboost. 85% of the top ten facial actions as weighted by MLR were among the first 10 facial actions to be selected by Adaboost.

Across Subject Drowsiness Prediction

The ability to predict drowsiness in novel subjects was tested by using a leave-one-out cross validation procedure. The data for each subject was first normalized to zero-mean and unit standard deviation before training the classifier. MLR was trained to predict drowsiness from the AU outputs several ways.

Table 2. The top 5 most discriminant action units for discriminating alert from non-alert states for each of the four subjects. A' is area under the ROC curve.

	AU	Name	A'
Subj1	45	Blink	.94
	17	Chin Raise	.85
	30	Jaw sideways	.84
	7	Lid tighten	.81
	39	Nostril compress	.79
Subj2	2	Outer brow raise	.91
	45	Blink	.80
	17	Chin Raise	.76
	15	Lip corner depress	.76
	11	Nasolabial furrow	.76
Subj3	45	Blink	.86
	9	Nose wrinkle	.78
	25	Lips part	.78
	1	Inner brow raise	.74
	20	Lip stretch	.73
Subj4	45	Blink	.90
	4	Brow lower	.81
	15	Lip corner depress	.81
	7	Lid tighten	.80
	39	Nostril Compress	.74

Table 3. Performance for drowsiness prediction, within subjects. Means and standard deviations are shown across subjects.

Classifier	Percent Correct	Hit Rate	False Alarm Rate
Adaboost	.92 ±.03	.92±.01	.06±.1
MLR	.94 ±.02	.98±.02	.13±.02

Performance was evaluated in terms of area under the ROC. For all of the novel subject analysis, the MLR output for each feature was summed over a temporal window of 12 seconds (360 frames) before computing A'. MLR trained on all features obtained an A' of .90 for predicting drowsiness in novel subjects.

Action Unit Predictiveness. In order to understand the action unit predictiveness in drowsiness MLR was trained on each facial action individually. Examination of the A' for each action unit reveals the degree to which each facial movement is associated with drowsiness in this study. The A's for the drowsy and alert states are shown in Table 4. The five facial actions that were the most predictive of drowsiness by *increasing* in drowsy states were 45, 2 (outer brow raise), 15 (frown), 17 (chin raise), and 9 (nose wrinkle). The five actions that were the most predictive of drowsiness by *decreasing* in drowsy states were 12 (smile), 7 (lid tighten), 39 (nostril compress), 4 (brow lower), and 26 (jaw drop). The

Table 4. MLR model for predicting drowsiness across subjects. Predictive performance of each facial action individually is shown.

More when critically drowsy

AU	Name	A'
45	Blink/eye closure	0.94
2	Outer Brow Raise	0.81
15	Lip Corner Depressor	0.80
17	Chin Raiser	0.79
9	Nose wrinkle	0.78
30	Jaw sideways	0.76
20	Lip stretch	0.74
11	Nasolabial furrow	0.71
14	Dimpler	0.71
1	Inner brow raise	0.68
10	Upper Lip Raise	0.67
27	Mouth Stretch	0.66
18	Lip Pucker	0.66
22	Lip funneler	0.64
24	Lip presser	0.64
19	Tongue show	0.61

Less when critically drowsy

AU	Name	A'
12	Smile	0.87
7	Lid tighten	0.86
39	Nostril Compress	0.79
4	Brow lower	0.79
26	Jaw Drop	0.77
6	Cheek raise	0.73
38	Nostril Dilate	0.72
23	Lip tighten	0.67
8	Lips toward	0.67
5	Upper lid raise	0.65
16	Lower lip depress	0.64
32	Bite	0.63

high predictive ability of the blink/eye closure measure was expected. However the predictability of the outer brow raise (AU 2) was previously unknown.

We observed during this study that many subjects raised their eyebrows in an attempt to keep their eyes open, and the strong association of the AU 2 detector is consistent with that observation. Also of note is that action 26, jaw drop, which occurs during yawning, actually occurred *less* often in the critical 60 seconds prior to a crash. This is consistent with the prediction that yawning does not tend to occur in the final moments before falling asleep.

Finally, a new MLR classifier was trained by contingent feature selection, starting with the most discriminative feature (AU 45), and then iteratively

Table 5. Drowsiness detection performance for novel subjects, using an MLR classifier with different feature combinations. The weighted features are summed over 12 seconds before computing A'.

Feature	A'
AU45	.9468
AU45,AU2	.9614
AU45,AU2,AU19	.9693
AU45,AU2,AU19,AU26	.9776
AU45,AU2,AU19,AU26,AU15	.9792
all the features	.8954

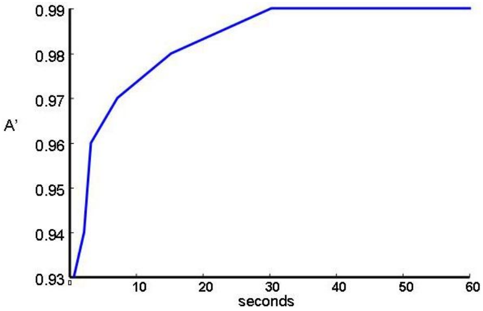


Fig. 5. Performance for drowsiness detection in novel subjects over temporal window sizes

adding the next most discriminative feature given the features already selected. These features are shown at the bottom of Table 5. Best performance of .98 was obtained with five features: 45, 2, 19 (tongue show), 26 (jaw drop), and 15. This five feature model outperformed the MLR trained on all features.

Effect of Temporal Window Length. We next examined the effect of the size of the temporal window on performance. The five feature model was employed for this analysis. The performances shown to this point in the paper were for temporal windows of one frame, with the exception of the novel subject analysis (Tables 4 and 5), which employed a temporal window of 12 seconds. The MLR output in the 5 feature model was summed over windows of N seconds, where N ranged from 0.5 to 60 seconds. Figure 5 shows the area under the ROC for drowsiness detection in novel subjects over time periods. Performance saturates at about 0.99 as the window size exceeds 30 seconds. In other words, given a 30 second video segment the system can discriminate sleepy versus non-sleepy segments with 0.99 accuracy across subjects.

3.3 Coupling of Steering and Head Motion

Observation of the subjects during drowsy and nondrowsy states indicated that the subjects head motion differed substantially when alert versus when the driver was about to fall asleep. Surprisingly, head motion increased as the driver became drowsy, with large roll motion coupled with the steering motion as the driver became drowsy. Just before falling asleep, the head would become still.

We also investigated the coupling of the head and arm motions. Correlations between head motion as measured by the roll dimension of the accelerometer output and the steering wheel motion are shown in Figure 6. For this subject (subject 2), the correlation between head motion and steering increased from 0.33 in the alert state to 0.71 in the non-alert state. For subject 1, the correlation between head motion and steering similarly increased from 0.24 in the alert state to 0.43 in the non-alert state. The other two subjects showed a smaller coupling effect. Future work includes combining the head motion measures and steering correlations with the facial movement measures in the predictive model.

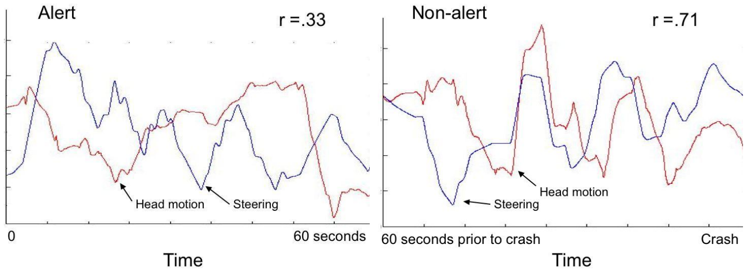


Fig. 6. Head motion and steering position for 60 seconds in an alert state (left) and 60 seconds prior to a crash (right). Head motion is the output of the roll dimension of the accelerometer.

4 Conclusion

This paper presented a system for automatic detection of driver drowsiness from video. Previous approaches focused on assumptions about behaviors that might be predictive of drowsiness. Here, a system for automatically measuring facial expressions was employed to datamine spontaneous behavior during real drowsiness episodes. This is the first work to our knowledge to reveal significant associations between facial expression and fatigue beyond eyeblinks. The project also revealed a potential association between head roll and driver drowsiness, and the coupling of head roll with steering motion during drowsiness. Of note is that a behavior that is often assumed to be predictive of drowsiness, yawn, was in fact a negative predictor of the 60-second window prior to a crash. It appears that in the moments before falling asleep, drivers yawn less, not more, often. This highlights the importance of using examples of fatigue and drowsiness conditions in which subjects actually fall sleep.

5 Future Work

In future work, we will incorporate motion capture and EEG facilities to our experimental setup. The motion capture system will enable analyzing the upper torso movements. In addition the EEG will provide a ground-truth for drowsiness. The new sample experimental setup can be seen in Figure 7.

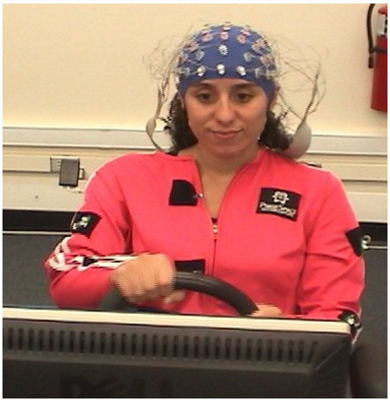


Fig. 7. Future experimental setup with the EEG and Motion Capture Systems

Acknowledgements. This research was supported in part by NSF grants NSF-CNS 0454233, SBE-0542013 and by a grant from Turkish State Planning Organization. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. DOT: Intelligent vehicle initiative. United States Department of Transportation, `http://www.its.dot.gov/ivi/ivi.htm./`
2. DOT: Saving lives through advanced vehicle safety technology. USA Department of Transportation, `http://www.its.dot.gov/ivi/docs/AR2001.pdf`
3. Takei, Y., Furukawa, Y.: Estimate of driver's fatigue through steering motion. In: Man and Cybernetics, vol. 2, pp. 1765–1770. IEEE Computer Society Press, Los Alamitos (2005)
4. Igarashi, K., Takeda, K., Itakura, F., Abut, H.: DSP for In-Vehicle and Mobile Systems. Springer, US (2005)
5. Cobb, W.: Recommendations for the practice of clinical neurophysiology. Elsevier, Amsterdam (1983)
6. Hong Chung, K.: Electroencephalographic study of drowsiness in simulated driving with sleep deprivation. International Journal of Industrial Ergonomics 35(4), 307–320 (2005)

7. Gu, H., Ji, Q.: An automated face reader for fatigue detection. In: FGR, pp. 111–116 (2004)
8. Zhang, Z., shu Zhang, J.: Driver fatigue detection based intelligent vehicle control. In: ICPR 2006. Proceedings of the 18th International Conference on Pattern Recognition, Washington, DC, USA, pp. 1262–1265. IEEE Computer Society Press, Los Alamitos (2006)
9. Gu, H., Zhang, Y., Ji, Q.: Task oriented facial behavior recognition with selective sensing. Comput. Vis. Image Underst. 100(3), 385–415 (2005)
10. Bartlett, M.S., Littlewort, G.C., Frank, M.G., Lainscsek, C., Fasel, I., Movellan, J.R.: Automatic recognition of facial actions in spontaneous expressions. Journal of Multimedia 1(6), 22–35
11. Orden, K.F.V., Jung, T.P., Makeig, S.: Combined eye activity measures accurately estimate changes in sustained visual task performance. Biological Psychology 52(3), 221–240 (2000)
12. Ekman, P., Friesen, W.: Facial Action Coding System: A Technique for the Measurement of Facial Movement. Consulting Psychologists Press, Palo Alto, CA (1978)
13. Fasel, I., Fortenberry, B., Movellan, J.R.: A generative framework for real-time object detection and classification. Computer Vision and Image Understanding (2005)
14. Kanade, T., Cohn, J.F., Tian, Y.: Comprehensive database for facial expression analysis. In: FG 2000. Proceedings of the fourth IEEE International conference on automatic face and gesture recognition, Grenoble, France, pp. 46–53. IEEE Computer Society Press, Los Alamitos (2000)

An Artificial Imagination for Interactive Search

Bart Thomee, Mark J. Huiskes, Erwin M. Bakker, and Michael S. Lew

LIACS Media Lab, Leiden University
{bthomee,Mark.Huiskes,erwin,mlew}@liacs.nl

Abstract. In this paper we take a look at the predominant form of human computer interaction as used in image retrieval, called interactive search, and discuss a new approach called *artificial imagination*. This approach addresses two of the grand challenges in this field as identified by the research community: reducing the amount of iterations before the user is satisfied and the small sample problem. Artificial imagination will deepen the level of interaction with the user by giving the computer the ability to think along by synthesizing ('imagining') example images that ideally match all or parts of the picture the user has in mind. We discuss two methods of how to synthesize new images, of which the *evolutionary synthesis* approach receives our main focus.

Keywords: Human computer interaction, Content-based image retrieval, Interactive search, Relevance feedback, Artificial imagination, Synthetic imagery, Evolutionary algorithms.

1 Introduction

In the early years of image retrieval – the mid '90s – searches were generally performed using only a single query [3]. It was soon realized that the results could be significantly improved by applying interactive search and hence it did not take long before interactive search methods [4-9, 11] were introduced into the field of image retrieval – or multimedia retrieval in general.

Interactive search, also known as relevance feedback, was initially developed to improve document retrieval [1]. Under this paradigm, the retrieval system presents a ranked set of objects relevant to the user's initial query and from thereon iteratively solicits the user for feedback on the quality of these objects and uses the feedback to compose an improved set of results. But, as keywords in a query can be matched one-on-one with text in a document and therefore good results can be obtained in a single step, its use has remained limited. In image retrieval, however, a user's query cannot be directly mapped onto items in the database. Interactive search turned out to be especially well suited for this problem: by interacting with the user, the system can learn which (features, parts of) images the user is interested in; the feedback resolves many of the uncertainties that arise as the system tries to learn what the user is looking for.

Despite the progress made, finding images of interest remains a major problem. Recent literature (e.g. [9], [11]), regards the following issues as the grand challenges in this field:

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 19–28, 2007.

1. *Bridging the semantic gap* through improved concept detection techniques. Since users think in terms of high-level semantic concepts and not in low-level image features as available to the system, it is very important to select the most useful image descriptors, to help narrow this so-called semantic gap.
2. *Overcoming the curse of dimensionality* by selecting only the most optimal features. It is also essential to use suitable multi-dimensional indexing techniques for an efficient search in high-dimensional feature space, especially considering that performance quickly suffers with an increase in dimensionality [12].
3. *Reducing the amount of iterations* before the user is satisfied. Requiring a minimal amount of effort a user has to invest is key in relevance feedback: if too much involvement is demanded, the user will be reluctant to use the system.
4. *Solving the small sample problem*, which is the issue that the user will only label a few images while the amount of dimensions in feature space is huge, making it very difficult to discover the user's interest.

Note that solving (or at least alleviating) the small sample problem will have a direct effect on the amount of iterations needed: if more samples are labeled by the user, the retrieval system obtains better insight into the user's interests and consequently will be able to return better images, thus more quickly satisfying the user.

Several methods have been proposed that address the small sample problem. In [13] it was found that combining multiple relevance feedback strategies gives superior results as opposed to any single strategy. In [14], Tieu and Viola proposed a method for applying the AdaBoost learning algorithm and noted that it is quite suitable for relevance feedback due to the fact that AdaBoost works well with small training sets. In [15] a comparison was performed between AdaBoost and SVM and found that SVM gives superior retrieval results. Good overviews can also be found in [16] and [11].

In section 2 we will address the third and fourth of above challenges by a new approach we call *artificial imagination*, which will deepen the level of interaction with the user by giving the computer the ability to think along by synthesizing ('imagining') example images that ideally match all or parts of the picture the user has in mind. In section 3 we will demonstrate our initial implementation of this approach.

2 Artificial Imagination

Our visual imagination allows us to create newly synthesized examples based on our memories and experiences. When we are learning new visual concepts, we often construct such examples based on real objects or scenes to help understand or clarify the primary features which are associated with the concept. One example from real life is when a journalist is looking for a photo to accompany his article and asks an archivist to find it, see Figure 1. *Artificial imagination* is the digital analogy of our own visual

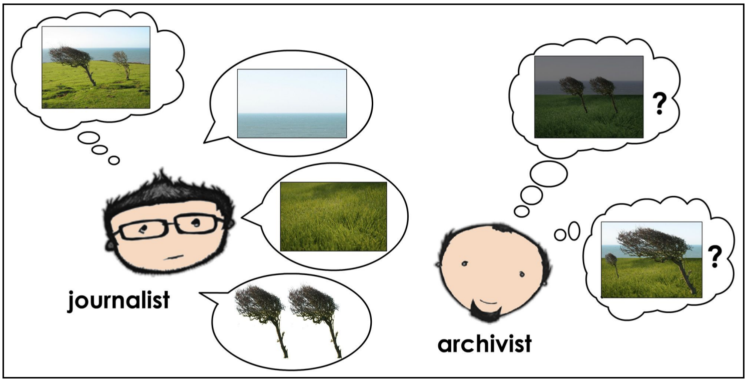


Fig. 1. An example of visual imagination. The journalist has a scene on its mind and tells the archivist what it looks like (sky, grass, trees). The archivist imagines scenes that contain the concepts mentioned; however because the image concepts might not be perfectly transferred, the imagined scenes do not necessarily initially resemble the image the journalist is thinking of (top-right imagined image). After exchanging image concepts (‘night-time or day-time?’, ‘trees equal size or one larger than the other?’) and obtaining more detailed information, the archivist is then better able to imagine the scene (bottom-right imagined image) and consequently is better able to return suitable images.

imagination. The computer is endowed with the ability to intelligently synthesize images and to present them to the user to ask whether or not they are relevant. These synthesized images are constructed in such a way that they target one or more particular features that are important to the query. Our idea is that the generated images (that are not in the database) are more in line with the user’s thoughts and consequently the user will be able to select more images as relevant during an iteration. We can then compare the results obtained from search queries without using synthesized images to search queries including the synthesized images and see to what extent our approach improves the results.

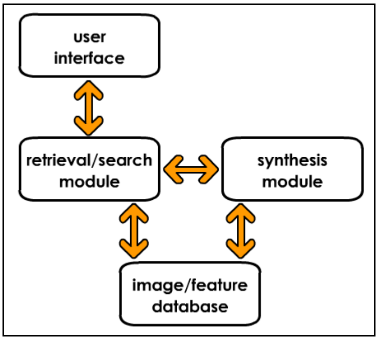


Fig. 2. Diagram of our retrieval system

A content-based image retrieval system employs methods that analyze the pictorial content of the images in the database and performs similarity comparisons to determine which images the user most likely is interested in. Generally, the pictorial content is translated to a set of image features and, based on these, each image is then placed at the appropriate location in the high-dimensional feature space. The similarity comparisons are performed directly in this space (e.g. [1]) or after mapping this space to a lower (e.g. [17]) or higher (e.g. [18]) dimensionality.

As we mainly want to focus on the feedback-based generation of examples, we use the classic and well-known relevance feedback method proposed by Rocchio [1], where the simple idea is to move a query point toward the relevant examples and away from the irrelevant examples. The Rocchio algorithm has the advantage of working relatively well when few examples are available. However, one challenging limitation of the Rocchio algorithm is that the single query point can necessarily refer to only a single cluster of results.

In Figure 2 a simplified system diagram is depicted of our content-based image retrieval system that uses synthetic imagery. It is important to realize that in our implementation the synthesis and retrieval/search aspects of the retrieval system are separate from each other: the feature space used to perform similarity comparisons does not necessarily have to be the same feature space used to synthesize images.

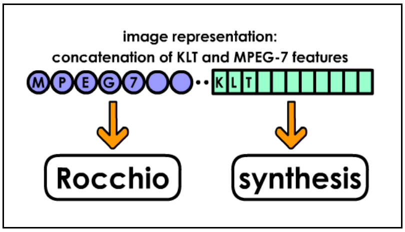


Fig. 3. The features used to represent an image serve different purposes

In our current system, we use several MPEG-7 [19] features (edge histogram, homogeneous texture, texture browsing and color layout) for the space in which we determine the similarity between images and in which we discover more relevant images by using the Rocchio method; for synthesis we use the feature space formed by taking the Karhunen-Loeve Transform (KLT, e.g. [20]) of each image in the database and using *N* coefficients from its KLT representation. Section 2.2 discusses in more detail how this is done. Thus, each image – real or synthesized – is associated with both an MPEG-7 and a KLT feature vector. This is illustrated in Figure 3.

The artificial imagination paradigm introduces several novel challenges, of which the most important ones are (i) which locations in feature space will be optimal candidates for synthesis and (ii) how do we synthesize an image given a point in feature space. In the following sections we suggest possible solutions to these two challenges.

2.1 Optimal Synthesis Locations

Here we describe two different methods to determine locations in the KLT feature space that are likely to result in suitable images when synthesized.

2.1.1 Inter-/Extrapolation of Feedback

By analyzing what effect the feedback has on the movement of the Rocchio query point over time, we can infer one or more locations where the query point likely will move after the next iteration; these points can therefore be synthesized, see Figure 4.

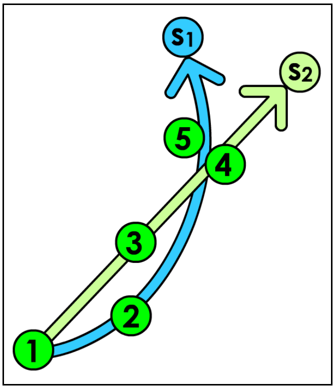


Fig. 4. Inferring likely future query points for synthesis: if over time the query point moved from point 1 and ending at 5, two likely future query points could be s_1 and s_2

2.1.2 Evolution of Feedback

Using evolutionary algorithms, points in feature space can be determined that are likely to maximize the 'fitness' (suitability, relevance) of the synthetic image. See Table 1 for the steps in our algorithm. After step 4 the algorithm loops back to step 2. An illustration of the crossover and mutation steps are shown in Figure 5.

Table 1. Evolutionary synthesis algorithm

step 1: **starting population**	random selection of images
step 2: **crossover**	sub-sampling: after feedback has been received, take subsets of the positive examples and mix their feature vectors to yield new points in feature space
step 3: **mutation**	negative examples and previous feedback can be used to move the points generated in step 2 to an adjusted location, or new random elements can be introduced into the syntesized images
step 4: **survival**	the user determines the fitness of the synthesized images by providing relevance feedback

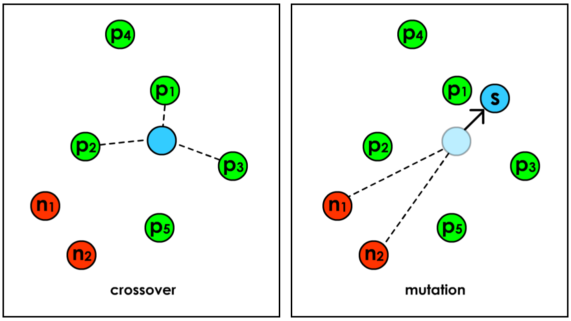


Fig. 5. Crossing over positive points p_1, p_2 and p_3 and mutating the resulting point through influence of negative points n_1 and n_2 to arrive at point s, from which a synthetic image can be generated

2.2 Image Synthesis

A point in KLT feature space is a set of coefficients weighting the eigenvectors of the KLT representation. We can thus synthesize the corresponding image by the standard method of linear reconstruction using the coefficients and corresponding eigenvectors. The MPEG-7 features of this synthesized image can be easily extracted by considering the image as any regular image and applying the MPEG-7 feature extractors to it, see Figure 6.

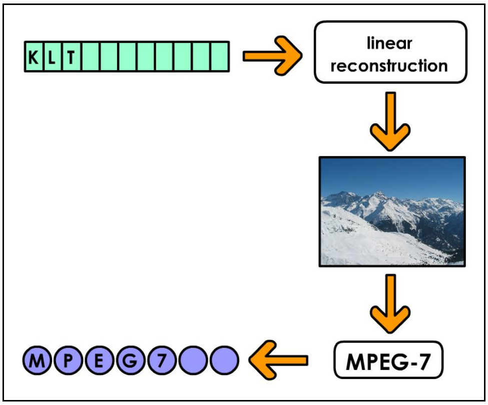


Fig. 6. From KLT feature point to synthetic image, followed by determining its MPEG-7 features. The KLT and MPEG-7 features together enable this new image to be used in the feedback process as if it is a real existing image from the image database

For clarity, we give a detailed example in pseudocode of the process of finding similar images and subsequently synthesizing a few images with the evolutionary synthesis algorithm. In this example, the user only selects relevant (positive) images from the initial random selection.

```
'preprocessing
load all N images from disk into I[n], where n = 1...N
for each n, do
   compute MPEG-7 features of image I[n] and store in M[n]
   compute KLT coefficients of image I[n] and store in K[n]
end

'initial selection
select R random images from I; present to user

wait for user to perform relevance feedback
store positive examples in P[i], i = 1...NP

'analyze feedback
given relevant images P[i] do
   determine Rocchio query point q = mean(M[P[i]])
   select most similar images W to q using distance to M[n]
   'crossover
   for each nonempty subset $A_k \subset P[i]$, k = 1...$(2^{NP}-1)$ do
      'synthesize new feature vector
      K[N+k] = mean(K[P[$A_k$]])
      synthesize image I[N+k] by inverse KLT on K[N+k]
      compute MPEG-7 features M[N+k] of image I[N+k]
   end
   return W and S = (I[N+1]...I[N+$2^{NP}$-1]) to the user
end
```

One should realize that the synthesis of an image using a feature space other than KLT may not be as straightforward. With KLT, a direct one-on-one mapping exists between the coefficients/eigenvectors and the pixels through linear reconstruction. This is generally not the case with other feature spaces where image reconstruction is not well-defined. For instance, suppose that color histograms are used as features. Given a color histogram, it is not possible to synthesize a unique image. We have information about how often each color should appear in the image, but we do not know where the colors should be located, and many images have the same histogram.

3 Results and Examples

We have developed a system for the retrieval of color texture images which uses 1000 textures taken from the Corel database. The images are represented by means of a decomposition in terms of “eigen-textures” obtained through the KLT transform and are additionally associated with several MPEG-7 features (edge histogram, homogeneous texture, texture browsing and color layout). The synthetic images we show are created using our evolutionary algorithm approach.

The following example provides a first proof-of-concept: using our current implementation, we typically observe small improvements in the relevance rankings after incorporating one or a small number of generated synthetic images as positive relevance feedback. As illustration we describe two image query sequences aimed at finding flowers/leaves that contain purple and green. In the first sequence no synthetic images are used in the process, while in the second sequence they are. The initial set of images that we indicate as relevant are shown in Figure 7.

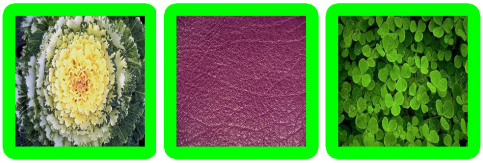


Fig. 7. Initial selection of relevant images

The results that are returned after submitting this query are shown in Figure 8. When not using any synthetic images that are generated, the user selects the five relevant images (shown with the green borders in Figure 8), and submits this modified query. The result from this second query is that the system is unable to return any new relevant images.

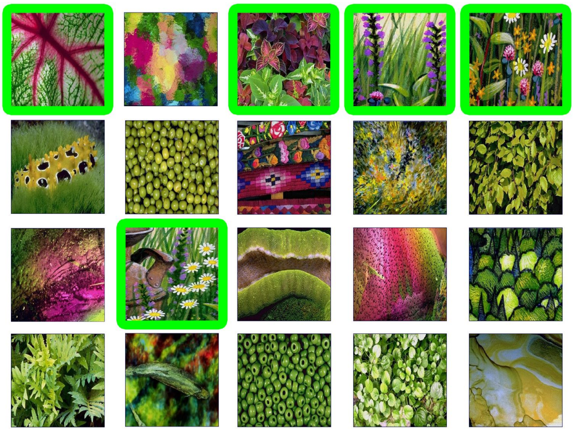


Fig. 8. Image ranking after the first step

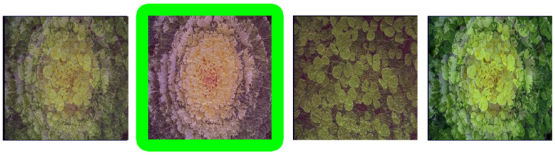


Fig. 9. Synthesized images by applying the evolutionary algorithm to the five relevant images that were selected in the initial screen

However, if the user had included the most relevant image from the set of synthesized images – the purple flower with the green border in Figure 9 – then the results would have improved with an additional two relevant images, see Figure 10.

Although the incorporation of synthetic images seems to only have modest positive effects, their use is nonetheless promising. The synthesized examples tend to show meaningful similarities with the positive examples, whereas the Rocchio query point often has a large distance to individual positive examples and sometimes centers on an undesirable cluster of images that show none or few of the desired image

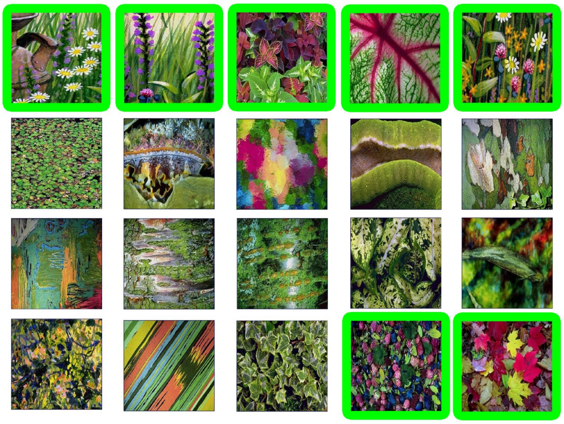


Fig. 10. Ranking after two iterations using the synthesized image

characteristics. In this case the synthetic images may offer valuable examples to steer the search to more relevant regions.

4 Conclusions and Future Work

Artificial imagination is a promising paradigm that can significantly enhance the level of interaction of the retrieval system with the user. By giving the retrieval system the power to imagine, it will more quickly understand what the user is looking for. Our evolutionary synthesis algorithm shows much potential for the generation of synthetic imagery. In future work we intend to explore other classification/synthesis methods such as wavelets [10], more advanced relevance feedback strategies such as [18], methods for dealing explicitly with the partial relevance of specific image aspects [8], and other focused areas such as face analysis and recognition [2]. Also, to apply the concept of artificial imagination to the case of general images (as opposed to textures only), we will work on the generation of collages, where the system can combine image concepts/objects by placing them in a single image.

Acknowledgments

We would like to thank Leiden University, University of Amsterdam, University of Illinois at Urbana-Champaign, the Dutch National Science Foundation (NWO), and the BSIK/BRICKS research funding programs for their support of our work.

References

1. Rocchio, J.J.: Relevance Feedback in Information Retrieval. In: Salton, G. (ed.) The Smart Retrieval System: Experiments in Automatic Document Processing, Prentice-Hall, Englewood Cliffs (1971)
2. Cohen, I., Sebe, N., Garg, A., Lew, M.S., Huang, T.S.: Facial expression recognition from video sequences. In: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME). Lausanne, Switzerland, vol. 1, pp. 641–644. IEEE Computer Society Press, Los Alamitos (2002)

3. Bach, J.R., Fuller, C., Gupta, A., Hampapur, A., Horowitz, B., Humphrey, R., Jain, R., Shu, C.-F: Virage Image Search Engine: An Open Framework for Image Management. In: Proceedings of the SPIE Storage and Retrieval for Still Image and Video Databases, pp. 76–87 (1996)
4. Chang, S.-F., Chen, W., Sundaram, H.: Semantic Visual Templates: Linking Visual Features to Semantics. In: Proceedings of the IEEE International Conference on Image Processing, pp. 531–535. IEEE Computer Society Press, Los Alamitos (1998)
5. Chen, Y., Zhou, X.S., Huang, T.S.: One-class SVM for Learning in Image Retrieval. In: Proceedings of IEEE International Conference on Image Processing, pp. 815–818. IEEE, Los Alamitos (2001)
6. Haas, M., Rijsdam, J., Thomee, B., Lew, M.S.: Relevance Feedback: Perceptual Learning and Retrieval in Bio-computing, Photos, and Video. In: Proceedings of the 6th ACM SIGMM International Workshop on Multimedia Information Retrieval, pp. 151–156 (October 2004)
7. He, X., Ma, W.-Y., King, O., Li, M., Zhang, H.: Learning and Inferring a Semantic Space from User's Relevance Feedback for Image Retrieval. In: Proceedings of the ACM Multimedia, pp. 343–346. ACM Press, New York (2002)
8. Huiskes, M.J.: Aspect-based Relevance Learning for Image Retrieval. In: Leow, W.-K., Lew, M.S., Chua, T.-S., Ma, W.-Y., Chaisorn, L., Bakker, E.M. (eds.) CIVR 2005. LNCS, vol. 3568, pp. 639–649. Springer, Heidelberg (2005)
9. Lew, M.S., Sebe, N., Djeraba, C., Jain, R.: Content-based Multimedia Information Retrieval: State of the Art and Challenges. ACM Transactions on Multimedia Computing, Communications, and Applications 2(1), 1–19 (2006)
10. Sebe, N., Lew, M.S.: Wavelet Based Texture Classification. In: Proceedings of the International Conference on Pattern Recognition, vol. III, pp. 959–962 (2000)
11. Zhou, X.S., Huang, T.S.: Relevance Feedback in Image Retrieval: A Comprehensive Review. Multimedia Systems Journal 8(6), 536–544 (2003)
12. Böhm, C., Berchtold, S.: Searching in High-Dimensional Spaces: Index Structures for Improving the Performance of Multimedia Databases. ACM Computing Surveys 33(3), 322–373 (2001)
13. Yin, P.-Y., Bhanu, B., Chang, K.-C., Dong, A.: Integrating Relevance Feedback Techniques for Image Retrieval Using Reinforcement Learning. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(10), 1536–1551 (2005)
14. Tieu, K., Viola, P.: Boosting Image Retrieval. International Journal of Computer Vision 56(1), 17–36 (2004)
15. Guo, G., Zhang, H.-J., Li, S.Z.: Boosting for Content-Based Audio Classification and Retrieval: An Evaluation. In: Proceedings of the IEEE Conference on Multimedia and Expo, IEEE Computer Society Press, Los Alamitos (2001)
16. Muller, H., Muller, W., Marchand-Maillet, S., Pun, T., Squire, D.: Strategies for Positive and Negative Relevance Feedback in Image Retrieval. In: Proceedings of 15th International Conference on Pattern Recognition, pp. 1043–1046 (2000)
17. Lin, Y.-Y., Liu, T.-L., Liu, C.H.-T.: Semantic manifold learning for image retrieval. In: Proceedings of the 13th annual ACM international conference on Multimedia, pp. 249–258. ACM Press, New York (2005)
18. Tong, S., Chang, E.: Support Vector Machine Active Learning for Image Retrieval. In: Proceedings of the 9th ACM International Conference on Multimedia, pp. 107–118. ACM Press, New York (2001)
19. Manjunath, B.S., Salembier, P., Sikora, T.: Introduction to MPEG-7: Multimedia Content Description Interface. John Wiley & Sons, Chichester (2002)
20. Therrien, C.: Decision, Estimation, and Classification. John Wiley & Sons, Chichester (1989)

Non-intrusive Physiological Monitoring for Automated Stress Detection in Human-Computer Interaction

Armando Barreto[1,2], Jing Zhai[1], and Malek Adjouadi[1,2]

[1] Electrical and Computer Engineering Department
[2] Biomedical Engineering Department
Florida International University
Miami, Florida, USA
{barretoa,jzhai002,adjouadi}@fiu.edu

Abstract. Affective Computing, one of the frontiers of Human-Computer Interaction studies, seeks to provide computers with the capability to react appropriately to a user's affective states. In order to achieve the required on-line assessment of those affective states, we propose to extract features from physiological signals from the user (Blood Volume Pulse, Galvanic Skin Response, Skin Temperature and Pupil Diameter), which can be processed by learning pattern recognition systems to classify the user's affective state. An initial implementation of our proposed system was set up to address the detection of "stress" states in a computer user. A computer-based "Paced Stroop Test" was designed to act as a stimulus to elicit emotional stress in the subject. Signal processing techniques were applied to the physiological signals monitored to extract features used by three learning algorithms: Naïve Bayes, Decision Tree and Support Vector Machine to classify relaxed vs. stressed states.

Keywords: Stress Detection, Affective Computing, Physiological Sensing, Bio-signal Processing, Machine Learning.

1 Introduction

New developments in human-computer interaction technology seek to broaden the character of the communication between the humans and computers. Picard [1] has pointed out the lack of responsivity to the affective states of users in contemporary human-computer interactions. Affective Computing concepts seek to empower computer systems to react appropriately to the affective states of the user. This, however, requires de development of methods to obtain reliable real-time assessment of the affective states experienced by the user. Several approaches for measuring affective states in the users have been tried, such as the identification of facial expressions, in isolation, or in combination with speech understanding and body gesture recognition [2]. Another approach for recognizing affect is through the monitoring of physiological signals [3]. Some previous attempts to recognize emotions from physiological changes have focused on variables that can be monitored in non-invasive and non-intrusive ways. Ark, Dryer and Lu, and the IBM "Blue Eyes" team (http://www.almaden.ibm.com/cs/BlueEyes/), developed the "emotional

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 29–38, 2007.

mouse”, an emotion recognition system based on a mouse-like device. However, one physiological variable that has not been studied extensively for the purpose of on-line affect recognition is the pupil dilation. In particular, the joint analysis of pupil diameter changes along with other physiological signals to indicate the emotional state of a computer user has not been fully investigated.

2 Affective Appraisal Through Physiological Signals

The rationale for monitoring physiological signals to investigate affective states in a subject is based on known associations between the divisions of the Autonomic Nervous System (ANS) and numerous physiological processes around the body. The ANS influences the cardiovascular, respiratory, digestive, urinary and reproductive functions. The Parasympathetic Division of the ANS stimulates visceral activity and promotes a state of “rest and repose” in the organism, conserving energy and fostering sedentary “housekeeping” activities, such as digestion. In contrast, the Sympathetic Division of the ANS prepares the body for heightened levels of somatic activity that may be necessary to implement a reaction to stimuli that disrupt the “rest and repose” of the organism, stimulating tissue metabolism and increasing alertness. When fully activated, this division produces a “flight or fight” response, which readies the body for a crisis that may require sudden, intense physical activity [4]. Therefore, we can expect that sympathetic activation under stress will modify multiple physiological signals in the subject undergoing such stress.

We have investigated the changes in a reduced set of physiological signals influenced by the ANS in computer users faced with a task known to elicit moderate levels of stress, and explored methods to analyze those signals for identification of the presence of stress in the subject. We monitored the Galvanic Skin Response (GSR), the Blood Volume Pulse (BVP) signal and the Skin Temperature (ST) in the extremities (in our case in the thumb), as well as the Pupil Diameter (PD) of the left eye of the subjects as they performed the prescribed computer task. We extracted several features from these four signals and applied machine learning algorithms to classify these groups of features towards the detection of instances that corresponded to segments of the records associated with the presence of stress in the subject.

When a subject experiences stress and nervous tension, the palms of his/her hands become moist, as increased activity in the Sympathetic Division of the ANS will cause increased hydration in the sweat ducts and on the surface of the skin. The resulting drop in skin resistance (increase in conductance) is recorded as a change in electrodermal activity (EDA), also called Galvanic Skin Response (GSR). The GSR is measured by passing a small current through a pair of electrodes placed on the surface of the skin and measuring the conductivity level. We used a GSR2 module, by Thought Technology LTD (West Chazy, New York) for that purpose. The resistance found between the two electrodes determines the oscillation frequency of a square-wave oscillator inside this device. We fed the square wave signal into a “frequency-to-voltage converter” integrated circuit (LM2917N), and calibrated this module by connecting several resistors of known resistance to it and measuring the output voltage from the module in each case.

The measurements of Blood Volume Pulse (BVP) in this project were obtained using the technique called photoplethysmography (PPG), to measure the blood volume in skin capillary beds, in the finger. PPG is a non-invasive monitoring technique that relies on the infrared light absorption characteristics of blood. We measured the BVP signal through a UFI model 1020 PPG finger plethysmograph. The analog BVP signal was digitized at 360 samples/second and used for the estimation of the heart-rate variability, which is another indicator of user affective state that can be utilized in the context of human-computer interaction [5].

The diameter of the pupil (normally between 1.5 mm and 9 mm) is determined by the relative contraction of two opposing sets of muscles within the iris, the sphincter and the dilator pupillae, and is determined primarily by the amount of light and accommodation reflexes [6]. The pupil dilations and constrictions are governed by the ANS. Previous studies have suggested that pupil size variation is also related to cognitive information processing. This, in turn, relates to emotional states (such as frustration or stress) since the cognitive factors play an important role in emotions [7]. Partala and Surakka have found, using auditory emotional stimulation, that the pupil size variation can be seen as an indication of affective processing [8]. In this research, we attempt to use the pupil size variation to detect affective changes during human-computer interactions. Currently, automatic instruments, such as infrared eye-tracking systems, can be used to record the eye information including pupil diameter and point of gaze. In our study, the subject's left eye was monitored with an Applied Science Laboratories model 504 eye tracking system running on a PC computer to extract the values of pupil diameter. The sampling rate of the system is 60 samples/second. To minimize the unwanted potential impact of illumination changes on the subject's pupil diameter, the lighting of the experimental environment and the average brightness of the stimulus computer were kept constant during the complete experimental sequences and across all the subjects.

Changes of acral skin blood flow may reflect sympathetic response to various stimuli. Under stress conditions the blood volume in the finger vessels are expected to decrease [9]. This leads to a transient decrease in skin temperature. In our experiment, the subject's skin temperature was measured with an LM34 integrated circuit (capable of providing a linear output between –50 ^{o}F and 300 ^{o}F), attached to the distal phalanx of the left thumb finger with the help of Velcro. The signal was amplified and recorded at the sampling rate of 360 samples/second. The experiments were performed in an air-conditioned room, to minimize the unwanted potential impact of environmental temperature changes on this experimental variable.

3 Experimental Setup Design

We developed a hardware / software setup to: Provide an appropriate stimulus, capable of eliciting stress in the subjects participating in the experiment, provide synchronization signals for the rest of the instrumental setup, and record the GSR, BVP, PD and ST signals with all the necessary time markers. Thirty two healthy subjects recruited from the student body at Florida International University (ages 21-42) participated in this study.

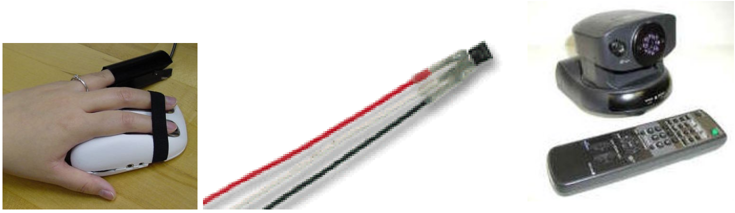


Fig. 1. GSR & BVP sensors (left), LM34 IC (center) and Eye Gaze Tracking system (right)

3.1 Stress Elicitation and Time Stamping of Protocol Landmarks

A computer application based on the ‘Stroop Test’ was designed to elicit mental stress at pre-defined intervals, while the subject interacted with the computer. The Stroop Color-Word Interference Test [10], in its classical version, demands that the color font of a word designating a different color be named. This task has been widely utilized as a psychological or cognitive stressor to induce emotional responses and heightened levels of physiological, (especially autonomic) reactivity [11]. The classical Stroop test was adapted into an interactive version that requires the subject to click on the button displaying the correct answer, which is one of five buttons shown on the screen, instead of stating it verbally. Task pacing was added to the Stroop test, by only waiting a maximum of 3 seconds for a user response, to intensify the physiological stress responses [11]. Typical examples of this test interface are shown on Figure 2. This interaction environment was also programmed to output bursts of a sinusoidal tone through the sound system of the laptop used for stimulation, at selected timing landmarks through the protocol, to time-stamp the physiological signals recorded at those critical instants. As a preliminary stage, the baseline level of the signals was recorded while 30 still, emotionally-neutral pictures were presented to the subject in order to have him/her rest for about 5 minutes. Afterwards, the actual interactive session proceeded through three consecutive and identical sections, shown in Figure 3. In turn, each section includes: 1) ‘IS’ - an introductory segment to let the subject get used to the interface environment, in order to establish an appropriate initial level for the psychological experiment, according to the law of initial values (LIV); 2) ‘C’ - a congruent segment, in which the font color and the meaning of the words presented to the user match. In each congruent segment, 45 trials are presented to the computer users. During congruent segments the expectation is that subjects will not experience stress; 3) ‘IC’ - an incongruent segment of the Stroop test in which the font color and the meaning of the words presented differ. There are 30 trials in this segment, and this is the segment where a stress response is expected; and 4) ‘RS’ - a resting segment to let the subject relax for about one minute.

The binary numbers shown at the bottom of Figure 3 indicate the output of sinusoidal bursts on the left channel (‘01’), the right channel (‘10’) , or both channels (‘11’), of the soundcard in the laptop used to present the Stroop stimuli to the subjects. External hardware, shown in Figure 4, was used to convert these sinusoidal bursts into signals used to time-stamp the digital recordings of GSR, BVP and ST, as well as the PD records, when 1) the user starts the session, 2) each of the congruent and incongruent segments begin, and 3) the user concludes the session.

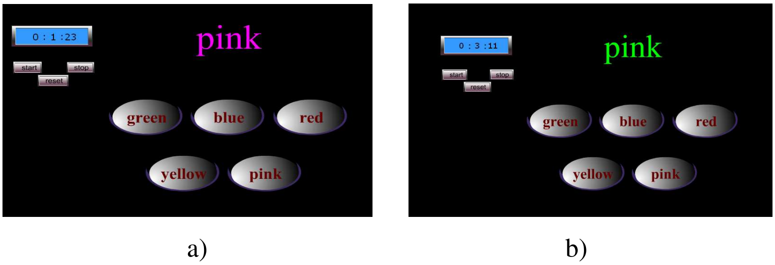


Fig. 2. Sample Stroop trials: a) Congruent: “pink” in pink; b) Incongruent: “pink” in green

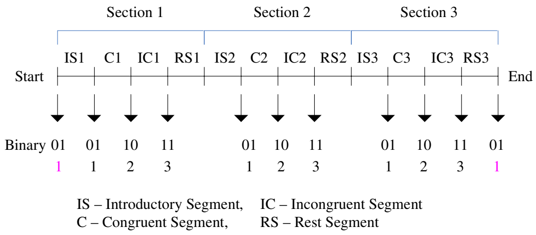


Fig. 3. Experimental sequence, identifying the audio output codes used as time stamps

3.2 Hardware Setup Design

The complete instrumental setup developed for this research [12] is illustrated in Figure 4. The stimulus program (Paced Stroop Test) described above runs in a laptop PC. While playing the Stroop test, the subject has the GSR, BVP and ST sensors attached to his/her left hand. All these three signals are converted, using a multi-channel data acquisition system, NI DAQPad-6020E for USB (National Instruments Corp.), to be read into Matlab® directly at a rate of 360 samples/sec. Additionally, the eye gaze tracking system (ASL-504) records PD data to a file on its own interface PC, at a rate of 60 samples/sec. Thanks to the common event markings in the BVP/GSR/ST and PD files, they can be time-aligned, offline. At this point the pupil diameter data can be upsampled (interpolated) by six, to achieve a common sampling rate of 360 samples/sec for all four measured signals.

4 Feature Extraction

For the purpose of this study, it was considered that each Stroop incongruent segment, in its entirety, would be associated with a “stressed” state in the subject. Similarly, each complete congruent Stroop segment was associated with a “non-stressed” state in the subject. In order to detect the stress (Stroop incongruent) segments during the experimental session, 11 features were derived from the four physiological signals measured during each segment. The digitized GSR signal, g[n], is characterized by a number of transient increases, or “GSR responses”. These responses were isolated by calculating the second derivative of the raw GSR signal and thresholding g”[n] to

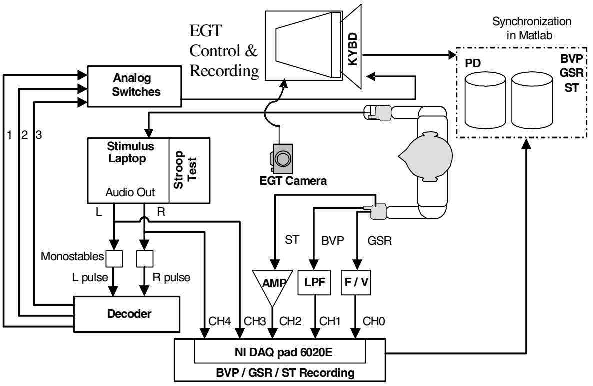


Fig. 4. Instrumental Setup

demarcate individual responses in the GSR signal. The number of the responses and the mean value in each segment could then be calculated. Additionally, the total area under the rising half of each GSR response is treated as the GSR response energy. So, the following five features were evaluated for each Stroop segment (congruent or incongruent): *Number of GSR Responses*; *Mean value of GSR*; (average) *Amplitude of GSR individual responses*; (average) *Rising time of individual responses*; and (average) *Energy of the individual Reponses*.

Heart rate variability (HRV) has been extensively studied to understand the function of the ANS, and has shown a close connection to the emotional state of the subject [5]. From the BVP signal, each heart beat was first separated to track the BVP period, also called interbeat interval (IBI), defined as the time in milliseconds between two normal, consecutive peaks in the BVP signal, and the beat amplitudes. The IBI series is frequently analyzed in the frequency domain, focusing on three specific bands: Very Low Frequency (VLF) (0.00-0.04Hz), Low Frequency (LF) (0.05-0.15Hz) and High Frequency (HF) (0.16-0.40Hz). The low frequency band reflects sympathetic activity with vagal modulation, and the high frequency band reflects parasympathetic activity. When a healthy person experiences mental stress, the sympathetic activity of his/her heart increases and the parasympathetic activity decreases, so the LF/HF ratio can be calculated as a single number estimate of mental stress. Therefore, four features were derived from the BVP signal in each Stroop segment: The *LF/HF ratio* (as described above); the *mean IBI*; the *standard deviation of IBI* and the *mean amplitude of individual BVP beats*.

The amplified skin temperature (ST) signal was first filtered to remove noise. *The average value of the filtered skin temperature* in each segment was then used as a feature of this signal, and was expected to decrease under stress.

The raw pupil diameter (PD) signal was recorded separately, as previously described. The artifact gaps due to blinking were filled by interpolation. The feature extracted from the pupil diameter is a simple parameter: *the mean value of PD*, which was expected to increase under stress. Eleven numerical features were defined to represent each Stroop segment (congruent or incongruent).

5 Data Normalization

For each subject, the previous feature extraction techniques were applied to the congruent (C) and incongruent (IC) segments in each of the three sections shown in Figure 3, and also to the relaxation interval when the subjects watched still pictures before the experiment started. Let X_s represent the feature value for any of the 11 features extracted from the signals which were recorded during congruent and incongruent segments of the protocol. Let X_r represent the corresponding feature value extracted from the signals which were recorded during the relaxation period, before the interactive segments started. To eliminate the initial level due to the individual differences, Equation (1) was first applied to get the corrected feature signals for each of the subjects.

$$Y_s = \frac{X_s}{X_r} \tag{1}$$

For each subject, there were three C segments and three IC segments. Therefore, six values of any of the features were obtained from the signals recorded during these segments. Equation (2) normalizes each feature value dividing it by the sum of all six segment values.

$$Y_s' = \frac{Y_{si}}{\sum_{i=1}^{6} Y_{si}} \tag{2}$$

These two stages of normalization proved essential for minimizing the impact of individual subject responses in the training of the learning systems used in our work. After this pre-processing, all features were normalized to the range of [0, 1] using max-min normalization, as shown in Equation (3), to be fed into the three learning systems described in the following section.

$$Y_{norm} = \frac{Y'_s - Y'_{s\min}}{Y'_{s\max} - Y'_{s\min}} \tag{3}$$

6 Stress Recognition

After all the features were extracted and normalized, they were provided as input to three types of learning classifiers: Naïve Bayes Classifier, Decision Tree Classifier, and Support Vector Machine (SVM), to differentiate the stress and non-stress states.

6.1 Learning Classifiers Used

The naïve Bayes classifier is a statistical learning algorithm that applies a simplified version of Baye's rule in order to compute the posterior probability of a category given the input attribute values of an example situation. This classifier is based on probability models that incorporate class conditional independence assumptions. The method computes the conditional probabilities of the different classes given the values of attributes of an unknown sample and then the classifier will predict that the sample belongs to the class having the highest posterior probability. The conditional probabilities are obtained from the estimates of the probability mass function using training data.

The decision tree classifier uses a ‘divide-and-conquer’ approach. It has a flow-chart-like tree structure, where each internal node involves testing a particular attribute, each branch represents an outcome of the test, and leaf nodes represent classes or class distributions. The basic idea involved is to break up a complex decision into a union of several simpler decisions, hoping the final solution obtained this way resembles the intended desired solution. To classify an unknown sample, it is routed down the tree according to the values of the attributes tested in successive nodes, and when a leaf is reached the instance is classified according to the class assigned to the leaf. A path is traced from the root to a leaf node which holds the class prediction for that sample. A J48 decision tree [13] is used for this classification task.

Support vector machines (SVMs) are learning classifiers that use a hypothesis space of linear functions in a high dimensional feature space to perform supervised classification. The support vector machine constructs a discriminant function for the data points in feature space in such a way that the feature vectors of the training samples are separated into classes, while simultaneously maximizing the distance of the discriminant function from the nearest training set feature vector. SVM classifiers also allow for non-linear discriminant functions by mapping the input vectors into a different feature space using a mapping function Φ: $xi \rightarrow \Phi(xi)$, and using the vectors, $\Phi(xi)$, $xi \in X$, as the feature vectors. The SVM classifier for this project used the sigmoid kernel. All classifiers were implemented using the Weka software [13].

6.2 Performance Measurements and Comparisons

The detection capability of the classifiers tested was assessed following the strategy of dividing input samples into training and test sets, as used in k-fold cross validation techniques. This strategy eliminates the need to test on unknown physiological signal samples whose labels (targets) may be uncertain. In this study we were particularly interested in investigating the added recognition capability that was achieved by including the pupil diameter measurements as part of the signals monitored. As a point of comparison, we also studied the level of recognition capability provided by skin temperature measurements. As such we have proceeded to perform the classification phase of our project under three conditions: using all the features extracted from all four signals monitored (BVP, GSR, ST and PD); excluding the PD signal; and excluding the ST signal. It should be noted that when each of the PD or ST signals were removed, all 3 learning classifiers were retrained and retested using 20-fold cross validation.

6.3 Results

Signals from 32 experimental subjects were collected and 192 feature vectors were extracted, since each participant generated data under three non-stress (Congruent Stroop) segments and three stress (Incongruent Stroop) segments. Eleven attributes were determined for each data entry (i.e., each segment). The prediction performance was evaluated using 20-fold cross validation: 20 samples were pulled out as the test samples, and the remaining samples were used to train the classifiers.

Performance of classification systems, such as the ones used here is commonly summarized in a confusion matrix. The results from our experiments are shown in that fashion in Table 1. It should be noted that this table has three sets of two rows, which represent the results for each of the three classification conditions that we used: involving information from all the sensors ("11 Features"); removing the information obtained from PD; and removing the information from ST.

Table 1. Confusion matrix for the three classification systems used (Shaded cells are the number of correctly recognized samples. NS = Non-stress)

			Classification					
			Naïve Bayes		Decision Tree		SVM	
			NS	Stress	NS	Stress	NS	Stress
Actual	11 Features	NS	66	30	85	11	88	8
		Stress	11	85	12	84	11	85
	PD Removed	NS	27	69	38	58	57	39
		Stress	20	76	26	70	40	56
	ST Removed	NS	79	17	85	11	88	8
		Stress	16	80	11	85	11	85

To evaluate the recognition ability of the classifiers, the total recognition accuracy, which is the number of correctly classified samples divided by the number of total samples, was calculated for each system. The overall accuracy reached in each case is listed in Table 2. The SVM has the highest recognition accuracy of the three approaches, in the 3 scenarios tested.

Table 2. Stress classification accuracies with three classifiers using physiological features

	Naïve Bayes	Decision Tree	SVM
11 Features	**78.65 %**	**88.02 %**	**90.10 %**
PD Removed	**53.65%**	**56.25%**	**58.85 %**
ST Removed	**82.81 %**	**88.54%**	**90.10 %**

7 Discussion and Conclusions

The first row of Table 2 allows a comparison of the performance exhibited by the three learning classifiers explored using the complete set of 11 features available for classifications of stressed and non-stressed segments. The SVM classifier, which can deal with more complex pattern distributions, shows the best performance of the three classifiers implemented.

It is also observed (second row of the table) that, with the PD signal excluded, the recognition rate dramatically dropped down, even for the SVM classifier (58.85%). The interaction of the information provided by the PD signal with the remaining signals is particularly interesting. By involving the pupil dilation signal into the affective recognition, the system gives much better performance (up to 90.10%). The potential importance of the PD signal for affect recognition may prompt its inclusion in future studies, particularly at a point in time when "web cams" on PCs are becoming ubiquitous and may evolve to a stage in which they might be used for pupil diameter assessment in standard PCs.

In contrast to the performance difference observed when the PD information was removed from the analysis, suppression of the single feature (mean value) extracted from ST did not seem to produce any noticeable change in classification performance.

While it is clear that the conditions for the testing reported here are not typical of human-computer interaction (controlled environmental conditions, prolonged blocks of congruent and incongruent stimulation, etc.), this research confirms the potential of digital signal processing and machine learning algorithms to address the challenges of affective sensing in computer users. Future work should include non-visual stress elicitation to avoid any confounding stimulus impact in pupil diameter variations.

Acknowledgments. This work was sponsored by NSF grants CNS-0520811, IIS-0308155, HRD-0317692 and CNS-0426125.

References

1. Picard, R.W.: Affective computing. MIT Press, Cambridge, Mass (1997)
2. Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., Taylor, J.G.: Emotion recognition in human-computer interaction. IEEE Signal Processing Magazine 18, 32–80 (2001)
3. Healey, J.A., Picard, R.W.: Detecting stress during real-world driving tasks using physiological sensors. IEEE Transaction on Intelligent Transportation Systems 6, 156–166 (2005)
4. Martini, F.H., Ober, W.C., Garrison, C.W., Welch, K., Hutchings, R.T.: Fundamentals of Anatomy & Physiology, 5th edn. Prentice-Hall, NJ (2001)
5. Dishman, R.K., Nakamura, Y., Garcia, M.E., Thompson, R.W., Dunn, A.L., Blair, S.N.: Heart rate variability, trait anxiety, and perceived stress among physically fit men and women. Int. Journal of Psychophysiology 37, 121–133 (2000)
6. Beatty, J., Lucero-Wagoner, B.: The Pupillary System. In: Cacioppo, Tassinary, Berntson (eds.) Handbook of Psychophysiology, pp. 142–162. Cambridge Press, Cambridge (2000)
7. Grings, W.W., Dawson, M.E.: Emotions and Bodily Responses A psychophysiological Approach. Academic Press, London (1978)
8. Partala, T., Surakka, V.: Pupil size variation as an indication of affective processing. International Journal of Human-Computer Studies 59, 185–198 (2003)
9. Krogstad, A.L., Elam, M., Karlsson, T., Wallin, B.G.: Arteriovenous anastomoses and the thermoregulatory shift between cutaneous vasoconstrictor and vasodilator reflexes. J. Auton. Nerv. Syst. 53, 215–222 (1995)
10. Stroop, J.R.: Studies of the interference in serial verbal reactions. Journal of Experimental Psychology 18, 643–662 (1935)
11. Renaud, P., Blondin, J.-P.: The stress of Stroop performance: physiological and emotional responses to color-word interference, task pacing, and pacing speed. International Journal of Psychophysiology 27, 87–97 (1997)
12. Barreto, A., Zhai, J.: Physiologic Instrumentation for Real-time Monitoring of Affective State of Computer Users. WSEAS Transactions on Circuits and Systems 3(3), 496–501 (2004)
13. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)

PEYE: Toward a Visual Motion Based Perceptual Interface for Mobile Devices

Gang Hua[1], Ting-Yi Yang[2], and Srinath Vasireddy[2]

[1] Microsoft Live Labs Research, One Microsoft Way, Redmond, WA 98052, USA
[2] Mirosoft Live Labs Engineering, One Microsoft Way, Redmond WA 98052, USA
{ganghua,tingyiy,srinathv}@mirosoft.com

Abstract. We present the architecture and algorithm design of a visual motion based perceptual interface for mobile devices with cameras. In addition to motion vector, we use the term "visual motion" to be any dynamic changes on consecutive image frames. In the lower architectural hierarchy, visual motion events are defined by identifying distinctive motion patterns. In the higher hierarchy, these visual events are used for interacting with user applications. We present an approach to context aware motion vector estimation to better tradeoff between speed and accuracy. It switches among a set of motion estimation algorithms of different speeds and precisions based on system context such as computation load and battery level. For example, when the CPU is heavily loaded or the battery level is low, we switch to a fast but less accurate algorithm, and vice versa. Moreover, to obtain more accurate motion vectors, we propose to adapt the search center of fast block matching methods based on previous motion vectors. Both quantitative evaluation of algorithms and subjective usability study are conducted. It is demonstrated that the proposed approach is very robust yet efficient.

1 Introduction

Video cameras have become a standard setting for all kinds of mobile devices, such as mobile phones, Pocket-PCs and personal digital assistants (PDAs). However, so far their functionalities are mostly limited to taking pictures or small video clips, mainly for entertainment or memorandum purpose. As an alternative to small keyboard, D-pad or touch screens, we can indeed leverage the rich visual information from the cameras for more convenient and non-invasive human machine interaction. This becomes of special interest due to the recent advancement of computer vision based human computer interaction (HCI) techniques [1,2,3] and the ever increasing computational power the mobile devices have.

To summarize some previous works, TinyMotion [4,5] may be one of the first general visual motion sensing systems for camera phones, which estimates the motion vectors by full search block matching (FSBM) [6] on grid samples. Other earlier work include the bar-code readers [7,8], where the phone cameras are used for reading highly regular shaped bar-code sheet to fire commands for the mobile phones. Hannuksela et. al [9] proposed a sparse motion estimation algorithm for vision based HCI on mobile devices, but they did not really come out of any real

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 39–48, 2007.

systems to use that. Some other research works even have investigated using mobile phone cameras for interacting with large displays [10].

Notwithstanding the demonstrated success of the systems discussed above, they have not fully leveraged the richness of the visual information we can leverage. For example, even the most recent TinyMotion [4,5] system has only utilized a fraction of the distinctive motion patterns to trigger the commands. What is worse, systems like bar-code readers [10,7] are obviously of limited use because they must be played with some distinctive visual patterns in a planar surface.

In this paper, we present the architecture design of a visual motion based perceptual interface for mobile devices. Here we use the term “visual motion” to incorporate any dynamic changes of image features. The overall architecture has two layers: at the lower-level of the hierarchy, we define a set of visual motion events, e.g., “left”, “right”, “up”, “down”, “rotate clockwise”, “rotate anticlockwise”, “blurred”, and “darkened”, based on different characteristic motion patterns. These visual events are fired through an integrated application programming interface (API), which we call *portable eye*, or in short PEYE. At the higher hierarchy, the applications can then be manipulated by these visual events. In view of this, we provide more freedom for the application developers to utilize these events for their own purpose.

The visual motion we exploited include motion vector, blur-ness and lightness changes, among which motion vectors still account for the majority of the visual events. Unlike TinyMotion [4,5], where FSBM is adopted for motion estimation, we explore a set of fast block matching methods such as three step search (TSS) [11], four step search (FSS) [12], diamond search (DS) [13], hexagon search (HS) [14], and the adaptive multiple-mode search (AMMS) [15]. To further improve the accuracy of all the above methods, we propose an adaptive search center scheme by pre-matching three points along the previous motion vectors. We then design an online scheme to switch among the different motion estimation algorithms, which makes a compromise between matching quality and computational expenses. For example, when the CPU is heavily loaded or the battery is low, we switch to a faster but less accurate method such as AMMS, DS, or HS, and vice versa to a more accurate but slower algorithm such as TSS and FSS. We call such a scheme *context aware motion estimation.*

Our contributions reside in three folds: firstly, we presented a novel hierarchical architecture design of visual motion based perceptual interface for mobile devices. Secondly, we designed a context aware motion estimation scheme, which better tradeoffs speed and accuracy for motion estimation. Thirdly, to the best of our knowledge, we are the first to utilize visual cues such as blur-ness and dark-ness changes to define visual events for HCI. Our system runs $25-30$ frames per second, depending on the frame rate of the mobile devices.

2 Platform Architecture

The PEYE API is developed using DirectShow under Windows CE. The overall platform architecture is presented in Fig. 1. It can be divided into three

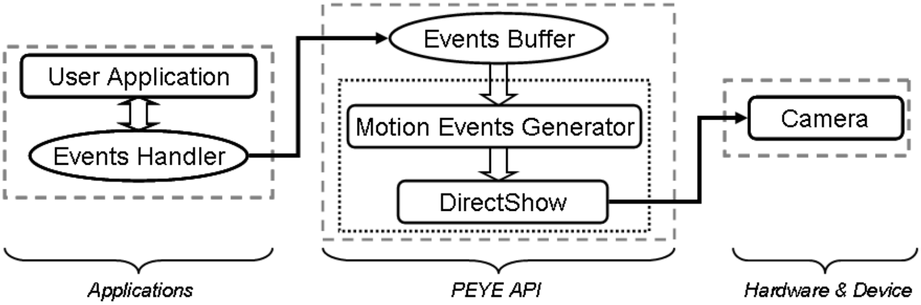


Fig. 1. The overall architecture of the PEYE system

hierarchies: the applications, the PEYE API, and the hardware devices (e.g., the camera). The core is the PEYE API, which analyzes the captured videos, and sends the *visual events* to the user applications. When the user applications receive these events, it responds just as it responds to any other *Windows messages*. We must note that under other mobile operating system, the video interface may need to be changed but the overall architecture is still valid.

The architecture design of our system distinguishes itself from previous ones (e.g., the TinyMotion [4,5]) in the sense that the motion analysis module is clearly separated from the applications. All communications between the user applications and the PEYE API are carried out through the message system of the OS. In other words, the user applications do not need to code any additional interfaces to export the results from the visual motion analysis module. Our hierarchical design also enables the user applications to have more flexibilities to respond to the different visual events.

3 Visual Events

Based on the different visual cues utilized, the visual events we exploited in PEYE fall into three categories: motion gestural, lightness, and blur-ness events.

3.1 Motion Gestural Events

The motion vectors provide the richest source of visual events, such as “left”, “right”, “up”, “down”, “pan”, “tilt”, etc. To detect these motion patterns, we need to estimate the motion vectors firstly.

Context Aware Motion Estimation with Adaptive Search Center. Due to the limited computational resource for PEYE, we can not afford dense optical flow estimation. Thus we partition the image into 4 equal regions, and perform fast block matching for the 4 center blocks of size 16×16, as illustrated in Fig. 2. We use the sum of squared difference (SSD) of pixel intensities as the matching

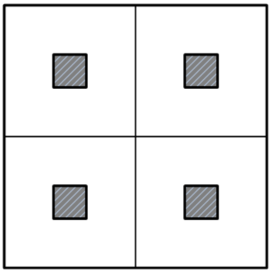


Fig. 2. Each 16 × 16 matching blocks (shaded small rectangles) is centered at one of the four equally sized partitions of the images

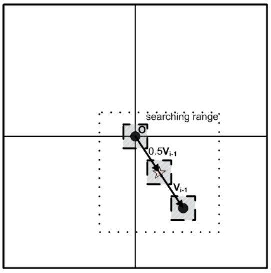


Fig. 3. The adaptive search center: $\boldsymbol{v}_{i-1}$ represents the previous motion vector. Three blocks are checked and the center of the best match is the new search center.

criterion. In our implementation, the square of pixel differences are implemented using a look-up table, which speeds up the matching at least 8 times.

Instead of utilizing a single matching method, we explored a set of fast block matching methods including TSS [11], FSS [12], DS [13], HS [14], and the AMMS [15]. These different fast matching algorithms are all faster than FSBM [6], although the gain in speed is at the expense of the accuracy to different extent. For example, in general HS is the fastest but least accurate one, whilst DS may be among the slowest several but it is more accurate. We refer the readers to the corresponding references for details of the different algorithms.

We improve all these fast block matching methods by introducing an adaptive search center scheme. Denote $\boldsymbol{v}_{i-1}$ as the motion vector estimated from the previous frame, also denote $\mathbf{O}$ as the zero search center, which can be the center of any of the shaded blocks in Fig. 2. We first match the template block at three points $\mathbf{O}$, $\mathbf{O} + \frac{1}{2}\boldsymbol{v}_{i-1}$, and $\mathbf{O} + \boldsymbol{v}_{i-1}$. The one with the smallest SSD will be chosen as the search center to start any of the aforementioned fast block matching methods, as illustrated in Fig. 3. We add the suffix "AC" to denote adaptive search center, e.g., FSSAC stands for FSS with adaptive search center.

Our study reveals that these different methods may perform differently over different motion patterns and/or different scenes. This motivated us to design an online selection scheme to switch among this set of motion estimation algorithms. The followings are general selection conditions with descending priorities:

1. If the battery power of the mobile device is below a certain level, then we always switch to HSAC (i.e., HS with adaptive search center) since it is the fastest one.
2. If the CPU is heavily loaded, we will select only between HSAC and AMMSAC, otherwise select among all algorithms, using Condition 3.
3. Based on the historical matching accuracies of each of the candidate methods, choose the most accurate one. Note the historical matching accuracy of each method is obtained by evaluating its accuracies every K frames and is accumulated over a shifting time window.

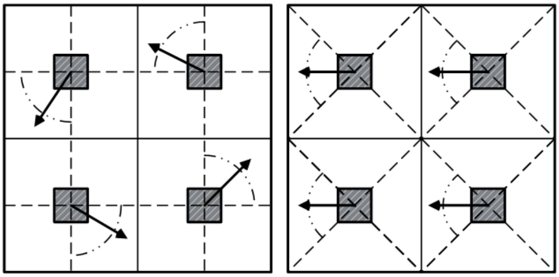


Fig. 4. The motion patterns associated with "rotate clockwise" and "right" events. The arcs indicate the direction ranges the motion vectors should fall into. The events are named according to camera motion, which is the reverse of camera motion. (Left figure: "rotate clockwise"; Right figure: "right")).

The above scheme enables the system to switch to a more accurate but slower matching method when more computation resource is available, and vice versa. We call it as *context aware motion estimation*. The motion vectors for each block are averaged over 5 consecutive frames to reduce the effects of jittering noise. Another engineering issue we should highlight here is that all the methods discussed above work with grey-scale images. In some of the mobile devices, the video stream is in YUV format, in that case the Y component is used directly. When the video stream is of the format RGB, we use bit shift to transform it to a grey scale image [5], i.e., $Y = (R >> 2) + (G >> 1) + (G >> 3) + (B >> 3)$, which approximates to the RGB to Y conversion formula.

Gestural Events. The motion gestural events can then be defined based on the four motion vectors estimated for each frame. For example, if all four motion vectors are going *left*, then the camera motion and thus the gestural event is "right", since we are indeed moving the camera to the *right*. Fig. 4 presents two motion patterns, which defines the motion gestural events "rotate clockwise" and "right". Other motion gestural events are defined in a similar fashion.

3.2 Lightness Events

The overall lightness of the image is also a very useful visual cue to define visual events, i.e., the "darkness" change events. Assuming that the environment lighting is not too dark, the image can only be dark if the camera view is occluded. Since the users can very conveniently cover the camera view by hand, it is natural and non-invasive for them to use it to interact with the mobile applications.

We estimate the darkness of the images by estimating the average pixel intensity I_{ave} over the four matching blocks defined in Fig. 2. If I_{ave} is below a threshold T_I, then it fires the event "darkened". The threshold T_I is set by collecting a set of images using the mobile cameras under different lighting conditions during which we also intentionally use our hand to block the view of the camera from time to time. We fit a Gaussian distribution with mean μ_I

and variance σ_I^2 on the average pixel intensities over all these image frames we collected. The threshold is set to be $\mu_I - 3\sigma_I$.

3.3 Blur-ness Events

There are mainly two types of blurs: the *de-focus* blur and the *motion* blur. De-focus blur happens when the mobile camera is out of focus, and motion blur happens when either the object or the camera is moving fast. In spite of the different causes, the blurred images present common characteristics – the lack of sharp object boundaries. This results in a very simple yet effective method to judge if an image is blurred. Denote $I_x(i,j)$ and $I_y(i,j)$ as the image gradient in x and y directions at pixel location (i,j), respectively. We define

$$I_{Blur} = \sum_{k=1}^{4} \sum_{(i,j)\in B_k} |I_x(i,j)| + |I_y(i,j)| \tag{1}$$

where B_k indicates the 25×25 block centered at the k^{th} block defined in Fig. 2. If I_{Blur} is smaller than a threshold T_{Blur}, the image is regarded as being blurred.

To distinguish between the de-focus blur and the motion blur, we exclude the case where both of them present simultaneously. Notice that for motion blur, the background scene is subject to dramatic change due to the large motion, while for de-focus blur, the frame differences should be subtle since the motion is small. Denote $I_t(i,j)$ as the pixel intensity at time frame t. We then define

$$I_m = \sum_{k=1}^{4} \sum_{(i,j)\in B_k} (I_t(i,j) - I_{t-1}(i,j))^2. \tag{2}$$

If I_m is larger than a threshold T_m and $I_{Blur} > T_{Blur}$, then the "motion blur" event will be alarmed, while if $I_m \leq T_m$ and $I_{Blur} > T_{Blur}$, the "de-focus blur" event will be issued. We reuse the square operation look-up table we built for estimating motion vectors to evaluate Eq. 2. Both T_{Blur} and T_m are determined empirically and are fixed in PEYE.

4 Applications

We have integrated PEYE with a bunch of interesting applications such as key-free mobile web browsing, pen-free sketchy, game playing, and automatic phone pick-up, as visualized in Fig. 5. We present more details as follows.

Key-free Mobile Web Browsing. Not until the recent development of the DeepFish system [16] has Web-browsing on mobile devices been an enjoyable experiences. We have integrated PEYE with the DeepFish browser to use motion gestural events to scroll up/down and zoom in/out the web pages. It provides a much more natural-to-use and non-invasive interaction scheme for the users to interact with the web browser. The first two figures in Fig. 5 show two views of using PEYE for web-browsing, the second is a zoom-in version of the first.

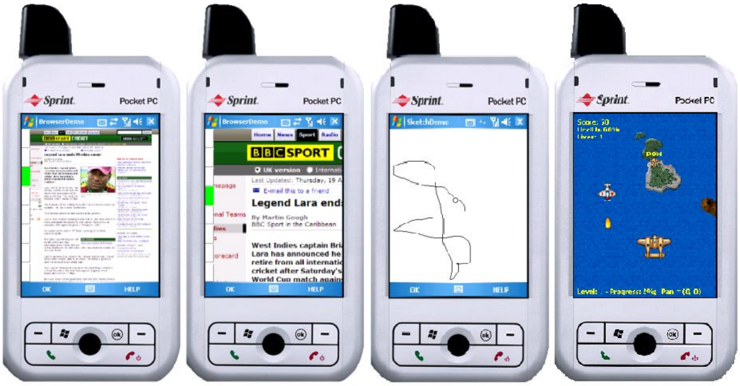


Fig. 5. Sample PEYE applications: a) key-free web browsing (left two figures), b) pen-free sketchy (the third), and c) games (last figure shows the game *1942*)

Pen-free Sketchy. PEYE provides a natural means to place sketches, where the users just need to move the mobile devices to place the sketches. We let the users to push the D-pad once to start or end one sketch. The third figure in Fig. 5 presents a screen shot on how we can draw a complex sketch. To erase the sketches, the users just need to use his hand to wipe off the camera view to trigger a "darkness" event. Possible extensions include recognizing sketches for inputting characters, especially for east Asian languages [5,4].

Mobile Games. When playing games on mobile devices, it is not that convenient to respond quickly using either small key-board or D-pad. We demonstrate that using PEYE, the users can respond very quickly since all he/she need to do is to move the mobile devices in a certain way. To show one example game we have deployed, the last figure in Fig. 5 presents a screen shot of an air-fighter game called *1942*. Certainly, we are also exploring means to let two or more users to play games together, each with his/her own mobile device.

Automatic Phone Pick-up. When the users put the mobile devices on the table, the cameras are usually facing down. If there is an incoming phone call and the users grab the mobile devices, PEYE will automatically detect a darkness transition event. Upon receipt of this event, the application can automatically answer the phone call instead of waiting for the users to push the pick-up button.

5 Evaluation

In this section, we evaluate the performance of the PEYE algorithms, and conduct usability studies for the mobile web browsing experiences.

5.1 Evaluation of Algorithms

In this section we evaluate the performances of the different block matching methods in terms of both speed and accuracy, with or without adaptive search

Table 1. The average number of matching positions for the different methods

	Average number of matching positions (#/frame)									
Methods\Video#	V1	V2	V3	V4	V5	V6	V7	V8	V9	Overall
FSBM	225									225
TSS	25									25
FSS	20.6	21.6	24.6	24.7	24.6	19.6	24.5	20.6	21.7	22.9
DS	18.9	20.4	26.6	25.5	25.5	16.8	24.7	18.6	20.5	22.6
HS	14.2	15.3	15.2	17.6	18.3	13.1	17.6	14.4	15.2	16.0
AMMS	11.4	14.4	25.2	25.3	25.5	8.9	22.9	11.8	14.7	18.9
TSSAC	26.4	25.8	26.9	26.9	26.9	26.3	27.0	26.6	26.7	26.8
FSSAC	20.2	20.3	25.6	24.2	24.8	18.8	24.6	18.9	20.0	22.3
DSAC	17.2	17.2	26.3	23.5	24.9	14.4	24.0	14.6	16.6	20.5
HSAC	13.7	14.4	17.0	16.4	17.0	12.8	16.5	13.3	14.2	15.3
AMMSAC	9.3	10.0	22.6	18.6	20.8	6.8	18.8	7.2	9.4	14.6

Table 2. The average SSD of the matching results for the different methods

	Average SSD /frame									
Methods\Video#	V1	V2	V3	V4	V5	V6	V7	V8	V9	Overall
FSBM	3.24	9.11	14.35	12.11	19.95	9.10	9.51	21.71	28.98	14.23
TSS	2.28	9.86	15.13	12.93	21.33	10.27	10.34	23.94	30.95	15.33
FSS	3.33	10.57	15.50	13.15	21.99	9.57	10.70	23.02	33.40	15.69
DS	3.32	9.95	14.77	12.58	20.84	9.28	10.44	23.18	33.08	15.27
HS	3.38	15.34	14.74	14.02	22.78	10.08	11.25	24.82	36.24	16.50
AMMS	3.39	14.45	14.86	12.51	20.74	9.47	10.57	24.99	35.07	15.79
TSSAC	3.23	9.05	11.59	8.53	14.38	9.37	8.44	21.71	28.13	13.13
FSSAC	3.24	9.15	12.15	8.64	15.60	9.34	8.77	21.98	29.21	13.12
DSAC	3.24	9.12	11.56	8.43	14.29	9.27	8.78	22.06	29.02	12.86
HSAC	3.32	10.50	13.34	11.75	19.43	9.86	10.33	23.66	33.78	15.10
AMMSAC	3.27	9.32	11.88	8.65	14.58	9.36	9.05	22.20	29.68	13.11

centers. The speed is measured by the number of matching locations. The less the number is, the faster the algorithm is. The accuracy is quantified by the SSD. The smaller it is, the better the match is. We use the results of FSBM as a baseline. Note all the matching are confined in a block of size 15 × 15.

Since we can not afford to do all the evaluations on the mobile device. We recorded 9 video sequences, which were taken against different background and/ or with different motions patterns using a Samsung Pocket-PC. Then we evaluated all the algorithms on a PC. We summarize the evaluation results in Table 1 and Table 2. As we can easily observe, except for TSS where only accuracy is improved, the adaptive center scheme does improve all the other algorithms in terms of both speed and accuracy. For example, AMMSAC on average only evaluates 14.6 positions (v.s. 18.9 for AMMS). DSAC achieves the best average matching error as low as 12.86 (v.s. 15.27 for DS). Another observation is that w.r.t. either speed or accuracy, none of the methods is constantly the best. This is the main motivation for us to design the third selection scheme in Sec. 3.1.

5.2 Usability Study

We perform usability study on the application of web-browsing using PEYE. With just some brief how-to-use descriptions, we let the users to browse a web using PEYE for interaction. The users then rank their experience in 5 grades, with 5 being "very satisfactory" and 1 being "very unsatisfactory". We also ask the users to compare their PEYE browsing experiences with that of using D-Pad. It is also ranked in 5 grades, with 5 being "much better" and 1 being "much worse". The users are also categorize to be "novice", "intermediate" or "advanced" based on how long they have been using mobile devices. 15 users participated in our usability study (6 novice, 8 intermediate, and 1 advanced).

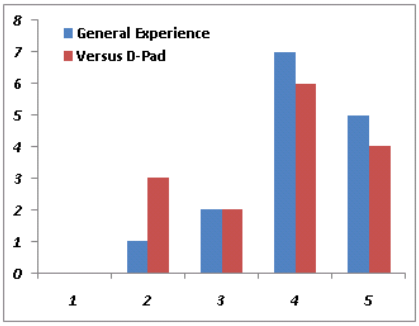


Fig. 6. The blue bar presents the distribution of the experiences of the users using PEYE, and the red bar displays the distribution of the users' experiences compared with that of using D-Pad

The histograms of the users' scores of the two studies are summarized in Fig. 6. As we can observe, 80% users rank their experiences using PEYE to be either *satisfactory* or *very satisfactory* (46.7%+33.3%). For comparison with the users' experience using D-Pad, 66.7% of the users think that it is *much better* or *better* to use PEYE than using D-Pad. On the other hand, three users (i.e., 20%) think that it is more convenient to use D-Pad. We found that two of the three users are in the intermediate level, and the other is in the advanced level. Moreover, the only user who ranked his/her experience with PEYE as unsatisfactory is an intermediate level user. This implies that they may have been biased toward using D-Pad since they have got used to it. We expect the users to prefer PEYE more, once they became more familiar with it.

6 Conclusion and Future Work

In this paper, we have presented PEYE: a novel computer vision based perceptual interface for mobile devices. Although the current implementation is under Windows CE, the algorithms as well as the architectural design can certainly be applied to other embedded OS. Our algorithm evaluation and usability study demonstrate the efficacy of the proposed approach. Future work includes identifying more visual events and developing novel applications based on PEYE.

References

1. Zhang, Z., Wu, Y., Shan, Y., Shafer, S.: Visual panel: Virtual mouse, keyboard and 3D controller with an ordinary piece of paper. In: Proc. ACM Perceptive User Interface Workshop, ACM Press, New York (2001)
2. Wilson, A., Oliver, N.: Multimodal sensing for explicit and implicit interaction. In: Proc. of 11th International Conference on Human Computer Interaction, Las Vegas, NV (July 2005)
3. Wilson, A.: Robust vision-based detection of pinching for one and two-handed gesture input. In: Proc. of Intenational Symposium on User Interface Software and Technology (UIST), Montreux, Switzerland (October 2006)
4. Wang, J., Canny, J.: Tinymotion: Camera phone based interaction methods. In: Proc. alt.chi of ACM CHI, Montreal, Canada (April 2006)
5. Wang, J., Zhai, S., Canny, J.: Camera phone based motion sensing: Interaction techniques, applications and performance study. In: Proc. of Intenational Symposium on User Interface Software and Technology (UIST), Montreux, Switzerland (October 2006)
6. Furht, B., Greenberg, J., Westwater, R.: Motion Estimation Algorithms for Video Compression. In: The International Series in Engineering and Computer Science, Springer, Heidelberg (2004)
7. Rohs, M., Zweifel, P.: A conceptual framework for camera phone-based interaction techniques. In: Gellersen, H.-W., Want, R., Schmidt, A. (eds.) PERVASIVE 2005. LNCS, vol. 3468, Springer, Heidelberg (2005)
8. Rekimoto, J., Ayatsuka, Y.: Cybercode: Designing augmented reality environments with visual tags. In: Proc. of DARE, pp. 1–10 (2001)
9. Jari Hannuksela, P.S., Heikkila, J.: A vision-based approach for controlling user interfaces of mobile devices. In: CVPR 2005. Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, Washington, DC, USA, p. 71. IEEE, Los Alamitos (2005)
10. Ballagas, R., Rohs, M., Sheridan, J.G.: Sweep and point and shoot: phonecam-based interactions for large public displays. In: CHI 2005 extended abstracts on Human factors in computing systems. Conference on Human Factors in Computing Systems archive, Portland, OR, USA, pp. 1200–1203 (2005)
11. Koga, T., Iinuma, K., Hiranoa, A., Iijima, Y., Ishiguro, T.: Motion compensated interframe coding for video conferencing. In: Proc. of IEEE NTC 1981, pp. G.5.3.1–G.5.3.4 (1981)
12. Po, L.M., Ma, W.C.: A novel four-step search algorithm for fast block motion estimation. IEEE Trans. on Circuits and Systems for Video Technology 6(3), 313–317 (1996)
13. Zhu, S., Ma, K.K.: A new diamond search algorithm for fast block-matching motion estimation. IEEE Trans. on Image Processing 9(2), 287–290 (2000)
14. Zhu, C., Lin, X., Chau, L.P.: Hexagon-based search pattern for fast block motion estimation. IEEE Trans. on Circuits and Systems for Video Technology 12(2), 355–394 (2002)
15. Liu, Y., Oraintara, S.: Adaptive multiple-mode search algorithm for fast block-matching motion estimation. In: IEEE Trans. on Circuits and Systems for Video Technology (September 2003)
16. `http://labs.live.com/deepfish/`

Vision-Based Projected Tabletop Interface for Finger Interactions

Peng Song, Stefan Winkler, Syed Omer Gilani, and ZhiYing Zhou

Interactive Multimedia Lab
Department of Electrical and Computer Engineering
National University of Singapore, Singapore 117576
{psong,winkler,omer,elezzy}@nus.edu.sg

Abstract. We designed and implemented a vision-based projected tabletop interface for finger interaction. The system offers a simple and quick setup and economic design. The projection onto the tabletop provides more comfortable and direct viewing for users, and more natural, intuitive yet flexible interaction than classical or tangible interfaces. Homography calibration techniques are used to provide geometrically compensated projections on the tabletop. A robust finger tracking algorithm is proposed to enable accurate and efficient interactions using this interface. Two applications have been implemented based on this interface.

Keywords: Projector-camera systems, projector-based display, interface design, tabletop interface, homography calibration, finger tracking, augmented reality, bare-hand interface.

1 Introduction

As projectors are becoming smaller and more affordable, they are used in more and more applications to replace traditional monitor displays because of their high scalability, larger display size and setup flexibility. Projection displays are no longer limited to traditional entertainment uses such as showing movies, playing video games, etc. They are now also being used for education [1], visualization [2], simulation [3], design [4] and interaction [5] applications.

The input to these applications is traditionally mouse-and-keyboard based, which is unnatural for users and limits their interactions and flexibility. Tangible interfaces have also been used in some projected environments. By holding some physical objects in hand, users feel comfortable manipulating the objects for interaction. Sensetable [6] uses a projected interface for visualization and design. Physical objects with embedded sensors can be held by users for movements to represent corresponding interactions. The Flatland system [7] projects onto a normal whiteboard, and interactions are based on the interpretation of strokes by the stylus on the whiteboard. More recently, Escritoire [8] uses special pens with embedded sensors to enable user interaction between a user and a projected tabletop. These applications are all based on manipulating tangible objects including pens for interactions. However, the flexibility can be greatly improved

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 49–58, 2007.

through using only hands and fingers. Barehand interfaces enjoy higher flexibility and more natural interaction than tangible interfaces. DiamondTouch [9] and SmartSkin [10] both rely on the users' hands for interaction, however, a grid of wired sensors needs to be embedded in the tabletop to enable the barehand interaction. Such special hardware and materials are expensive and not readily accessible to everyone.

As cameras can be used to track the hand and finger movements through computer vision approaches, we propose a vision-based projected tabletop interface with finger tracking. Webcams are very cheap and non-intrusive sensors, therefore the tracking of fingers by a webcam not only provides more flexible, natural and intuitive interaction possibilities, but also offers an economic and practical way of interaction for the user and the computer.

The design of this interface will be presented in section 2, followed by a discussion on homography calibration in section 3. In section 4, a robust finger tracking algorithm will be proposed. Two applications based on this interface, and the finger tracking results will be reported in section 5.

2 Interface Design

The projector and the camera are mounted on tripods with an overlapping view of the tabletop. The projector is projecting from the upper left onto the tabletop, and the camera is observing the projected area from the upper right. The projector and the camera can be randomly put in the above mentioned manner without any calibration/measurement. In this setup, the camera can see the hand and its shadow under the projector illumination during the interaction.

As shown in Figure 1, the camera is overlooking the projected content on the tabletop and the hand as well. The geometric distortion caused by the obliquely positioned projector can be compensated through homography transformation on the image before projection (see section 3). The finger and its shadow can be tracked using the camera input, and tracks of the finger can be projected onto the tabletop in a handwriting recognition system, or used to indicate the direction and location a document is dragged to simulating typical desk operations.

Our system only requires a commodity projector with an off-the-shelf webcam. Users need no other equipment or devices for interaction, hence the system provides direct, natural and intuitive interactions between the user and the computer.

3 Homography Calibration

In our setup, the projector and the camera can be casually set up at oblique angles for minimizing the occlusion by the human hand in the projection area. As a result, images sent to the projector need to be geometrically compensated for the projective distortions, as shown in Figure 2, in order to achieve an undistorted view from the user's perspective.

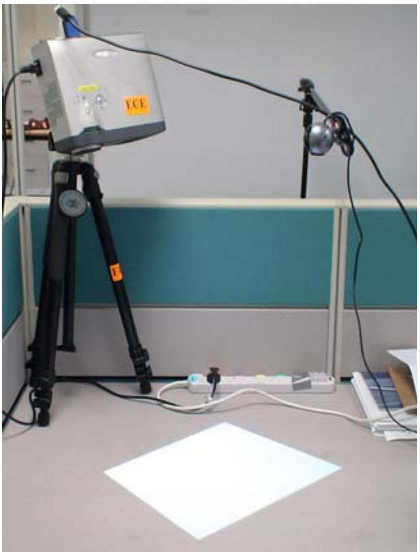


Fig. 1. System setup with the projector and the camera overlooking the tabletop. The projector and the camera can be casually and obliquely positioned, as long as the camera can see the projection area.

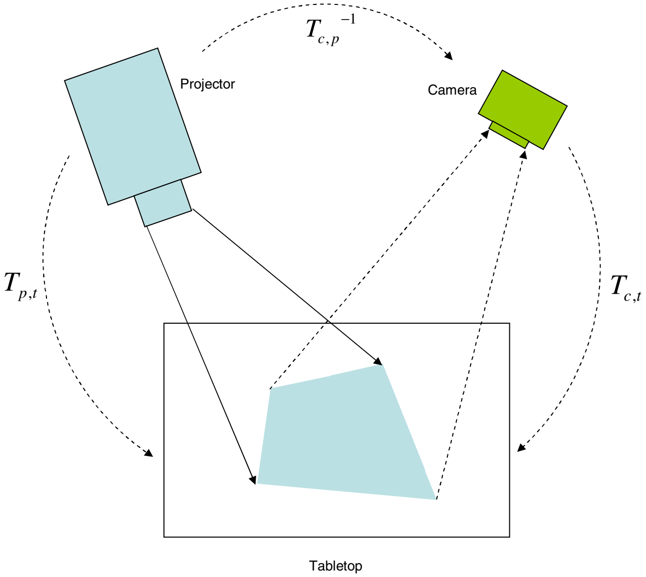


Fig. 2. Homography calibration. Due to misalignment of projector and projection surface (tabletop), the rectangular display appears as a distorted quadrilateral in the center of the tabletop. This can be compensated by pre-warping using the homographies shown.

As the distortion is predominantly the result of perspective projection, pre-warping the image via a homography can be used to correct for the distortion. The homographies between projector, camera, and the tabletop need to be calibrated prior to pre-warping. Sukthankar *et al.* [11, 1] proposed an automatic keystone correction method for the above setup. A simple solution using four feature point correspondences is shown in Figure 2. If a white rectangle is projected onto the tabletop against a high-contrast background, the four corners of the projected quadrilateral in the camera image can be computed. With the coordinates of the corners known in the projector reference frame, the projector-camera homography $\mathbf{T}_{cp}$ can be recovered. The corners of this quadrilateral completely specify the camera-tabletop homography $\mathbf{T}_{ct}$. The homography between the projector and the tabletop $\mathbf{T}_{pt}$ can be recovered from $\mathbf{T}_{pt} = \mathbf{T}_{cp}^{-1}\mathbf{T}_{ct}$. By applying the inverse transform $\mathbf{T}_{pt}^{-1}$ to pre-warp the original image, a corrected image can be displayed on the tabletop.

4 Fingertip Tracking

Because our projected tabletop interface provides finger interaction, the finger of the user needs to be tracked in the camera view such that interactions can be enabled accordingly. There are many vision-based techniques that can track fingers. However, there are many constraints on these methods: methods based on color segmentation [12] need users to wear colored gloves for efficient detection; wavelet-based methods [13] are computationally expensive and non real-time; contour based methods [14] work only on restricted backgrounds; infrared segmentation based methods [15, 16] require expensive infrared cameras; correlation-based methods [17] require an explicit setup stage before the tracking starts; the blob-model based method [18] imposes restrictions on the maximum speed of hand movements. In order to provide robust real-time hand and finger tracking in the presence of rapid hand movements and without the need of initial setup stage, we propose an improved finger tracking based on Hardenberg's fingertip shape detection method [19] with more robustness and the ability to detect the event of a finger touching the surface.

4.1 Fingertip Shape Detection

Hardenberg [19] proposed a finger tracking algorithm using a single camera. With its smart image differencing, the user's hand can be easily distinguished from the static background. Fingertips are then needed to be detected for interaction purposes.

In a difference image, the hand is represented in filled pixels, while the background pixels are all unfilled, as shown in Figure 3(a). If a square box is shown around a fingertip, as in Figure 3(b), the fingertip shape is formed by circles of linked pixels around location (x, y), a long chain of non-filled pixels, and a short chain of filled pixels.

In order to identify the fingertip shape around pixel location (x, y), 3 criteria have to be satisfied:

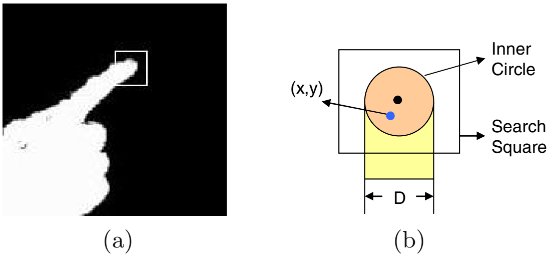


Fig. 3. Fingertip Shape Detection. (a) Difference image shows the fingertip shape of filled pixels on a background of non-filled pixels. (b) The fingertip at position (x, y) can be detected by searching in a square box (see text).

1. In the close neighborhood of position (x, y), there have to be enough filled pixels within the searching square;
2. The number of filled pixels has to be less than that of the non-filled pixels within the searching square;
3. The filled pixels within the searching square have to be connected in a chain.

The width of the chain of filled pixels can be expressed as D and is the diameter of the identified finger. D needs to be preset/adjusted in order to detect fingers of different diameters in the camera view.

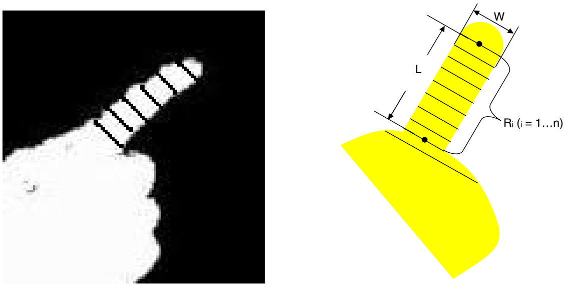
This method works well under normal lighting conditions. However, under the strong projector illumination in our setup, a lot of false fingertip detections will appear because of the failure of smart differencing in such conditions. In order to detect the fingertips more robustly, a finger shape detection method is proposed in the following section.

4.2 Finger Shape Detection

The above detection method may produce false detections on the finger end in connection with one's palm, as shown in Figure 4(a), because of its similar shape. However, these false detections can be eliminated if the shape of the finger is taken into consideration. As a fingertip always has a long chain of filled pixels connected to the palm, and the width of the chain of filled pixels will be greatly changed only at the connection from the finger to the palm, a more robust finger detection algorithm based on the shape of the finger is proposed as follows.

When a fingertip is detected from the method used in section 4.1, record the center of the fingertip, move further along the direction of the chains of filled pixels. As shown in Figure 4(b), the width $W_i \ (i = 1 \cdots n)$ of the i^{th} row of filled pixels $(R_i \ (i = 1 \cdots n))$ orthogonal to the direction of the detected chain of filled pixels is computed. If the width of row W_{i+1} is comparable with W_i, move along the direction of the chain of filled pixels, until the width increase between row W_n and W_{n-1} increases dramatically.

The length of the potential finger L can be derived by computing the distance between the center pixel of row R_1 and that of row R_{n-1}. A finger is confirmed



(a) (b)

Fig. 4. Finger Shape Detection. (a) Difference image showing the finger shape of a long chain of filled pixels connected with the palm's chain of filled pixels. The finger is labeled by rows of black pixels orthogonal to the direction of the chain of filled pixels detected in Hardenberg's method [19]. (b) A finger can be represented by a long chain of filled pixels. Along the finger, the width W of chain of filled pixels orthogonal to the direction of the detected chain of filled pixels will change dramatically at the connection of a finger to its palm.

if L is sufficiently long, since false detected fingertips normally do not have long fingers attached.

Through employing the finger information, this algorithm effectively eliminates the false fingertip detections based on the fingertip shape information.

4.3 Touch-table Detection

Since only a single camera is used for detection, no depth or touch information is available to identify if the finger is touching the tabletop. However, under projector illumination in our setup, prominent finger shadows are observed, which are also detected as fingers by the above algorithm. Therefore, when two detected fingertips (the real finger and its shadow) merge into one, we can assume that the finger touches the tabletop.

In order to keep tracking the finger merged with its shadow while it is moving on the tabletop as shown in Figure 5(b), the diameter D of the finger detection algorithm has to be adjusted. If a finger moving in the air is to be detected, the diameter of the fingertip is set to D_2. Likewise, if a finger touching the tabletop is to be detected, the diameter of the fingertip is set to D_1 where $D_1 > D_2$. The diameters D_2 and D_1 can be pre-determined accordingly. However, in order to automatically switch between these two diameters when a finger is touching and leaving the tabletop, a two-state system is proposed based on the assumption that there is only one finger used in the interaction.

As shown in Figure 6, an initial state is set to 2 fingers tracked with diameter D_2, as the user would first move his hand in the air before touching the tabletop. The state of each frame can be determined using the fingertip diameter D in the previous frame, and then D may be adjusted according to the state diagram.

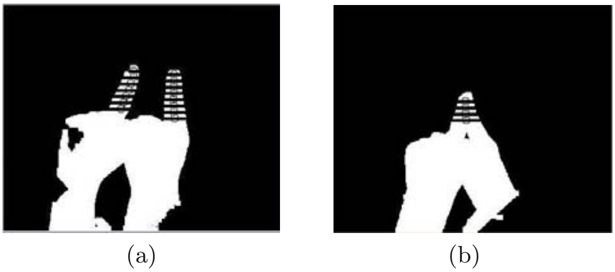


Fig. 5. Touch Table Event Detection. (a) Finger moving in the air; both its shadow and itself are detected as separate fingers. (b) Finger touching the table; it merges with its shadow, resulting in the detection of a single wide finger.

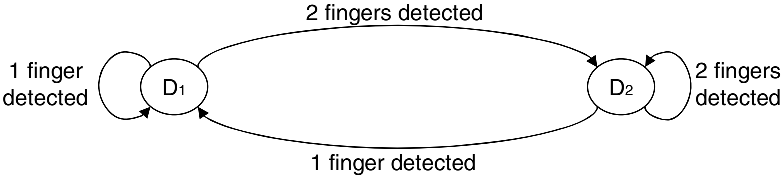


Fig. 6. State diagram for finger diameter switching

5 Applications and Results

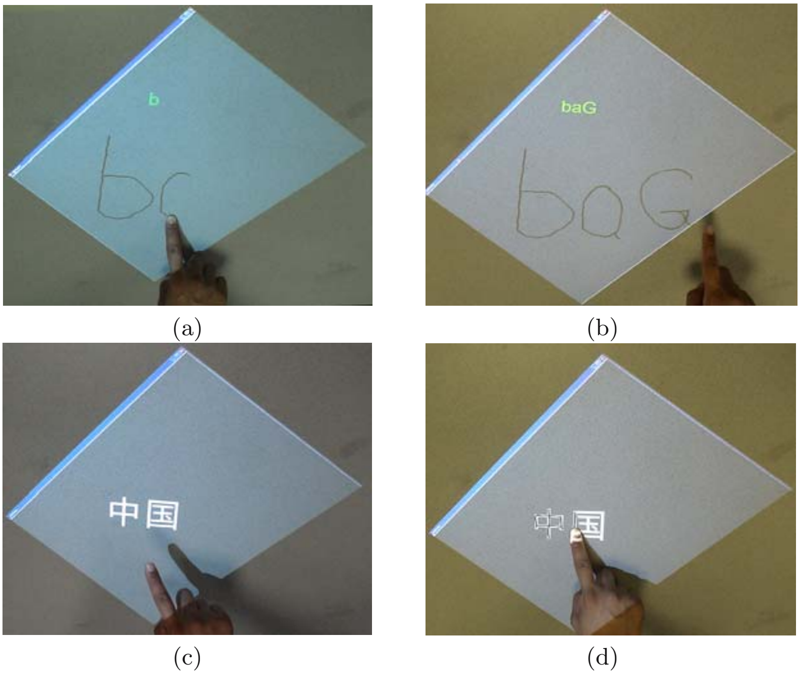
Our system comprises an Optoma EP-739H DLP portable projector, a Logitech webcam and a Dell Optiplex GX620 Desktop PC. Two applications are tested on this interface and shown below.

5.1 Handwriting Recognition and Template Writing

Through tracking the forefinger of a user on the tabletop, the stroke information can be projected onto the tabletop after geometric compensation. Based on Microsoft Windows XP Tablet PC Edition 2005 Recognizer Pack [20], recognized letters can be shown on the upper-right corner in the projection area (shown in Figure 7(a,b)). In addition, to train kids to learn how to write, colored templates can be projected onto the tabletop, and a user can write on the template while his writing is shown in projected strokes (shown in Figure 7(c,d)).

5.2 Desktop Finger Mouse

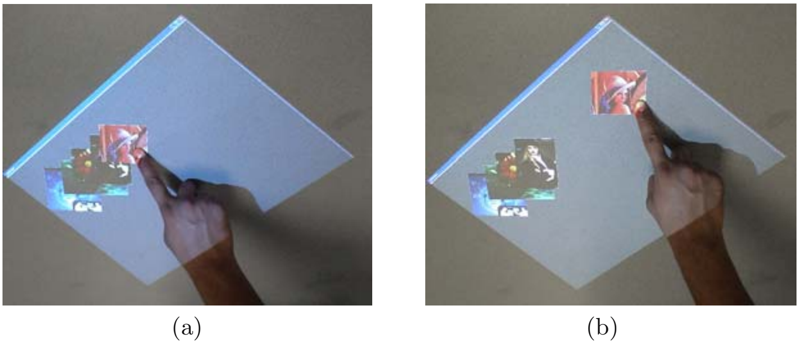
Because the finger of a user can be tracked both when it is moving in the air and when moving on the tabletop, a few mouse functions can be replaced by finger touching the table, moving on the tabletop, etc. Shown in Figure 8 is the action of a user selecting a file (a) and drags it to another location in the projection area (b).



(a) (b)

(c) (d)

Fig. 7. Screenshots of Handwriting Recognition System and Template Writing. A user completing the writing of "baG" using his finger and recognized by our system (green printed words) is shown in (a,b); two Chinese characters are projected in (c) and the user is using his hand to following the strokes to practice Chinese calligraphy (d).



(a) (b)

Fig. 8. Desktop Finger Mouse.A user is using his finger as a mouse to "select" a picture on the left (shown in (a)) and "drag" it to the right part of the display (shown in (b)).

5.3 Fingertip Detection Results

Our improved finger detection algorithm proposed in section 4 is more robust. In table 1, the finger tracking accuracy rates of our method is shown in comparison with Hardenberg's Method [19]. Among 180 sampled frames, the average correct detection percentage of our method (88.9%) is much higher than that of their method (46.1%). Among these frames, the finger can be either correctly detected, or not detected at all, or detected with additional 1 false detection, or detected with additional 2 false detections. The number of frames corresponding to each type of detection is shown in table 1 below.

Table 1. Finger Tracking Accuracy Rates

	Types	180 Frames	Average Accuracy
Hargenberg's method	correct	83	46.1%
	1 false detection	71	
	2 false detections	26	
	no detection	0	
Our method	correct	160	88.9%
	no detection	20	

6 Conclusions and Future Work

We have designed and implemented a vision-based projected tabletop interface for finger interactions. Our interface provides more direct viewing, natural and intuitive interactions than HMD's or monitor-based interfaces. The system setup is simple, easy, quick and economic since no extra special devices are needed. Homography calibration is used to compensate for distortions caused by the oblique projection. A fast finger tracking algorithm is proposed to enable robust tracking of fingers for interaction.

We are now conducting a usability study to evaluate the interface from a user perspective. Furthermore, due to the oblique projection, there are issues such as projection color imbalance and out-of-focus projector blur that need to be investigated.

References

1. Sukthankar, R., Stockton, R., Mullin, M.: Smarter Presentations: Exploiting homography in camera-projector systems. In: Proc. International Conference on Computer Vision, Vancouver, Canada, pp. 247–253 (2001)
2. Chen, H., Wallace, G., Gupta, A., Li, K., Funkhouser, T., Cook, P.: Experiences with scalability of display walls. In: Proc. Immersive Projection Technology Symposium (IPT), Orlando, FL (2002)
3. Czernuszenko, M., Pape, D., Sandin, D., DeFanti, T., Dawe, L., Brown, M.: The ImmersaDesk and InfinityWall projection-based virtual reality displays. Computer Graphics, 46–49 (1997)
4. Buxton, W., Fitzmaurice, G., Balakrishnan, R., Kurtenbach, G.: Large displays in automotive design. IEEE Computer Graphics and Applications 20(4), 68–75 (2000)

5. Song, P., Winkler, S., Tedjokusumo, J.: A Tangible Game Interface Using Projector-Camera Systems. In: HCI 2007. LNCS, vol. 4551, pp. 956–965. Springer, Heidelberg (2007)
6. Patten, J., Ishii, H., Pangaro, G.: Sensetable: A wireless object tracking platform for tangible user interfaces. In: Proc. CHI, Conference on Human Factors in Computing Systems, Seattle, Washington, USA (2001)
7. Mynatt, E.D., Igarashi, T., Edwards, W.K.: Flatland: New dimensions in office whiteboards. In: Proc. CHI 1999, Pittsburgh, PA., USA (1999)
8. Ashdown, M., Robinson, P.: Escritoire: A personal projected display. IEEE Multimedia Magazine 12(1), 34–42 (2005)
9. Leigh, D., Dietz, P.: DiamondTouch characteristics and capabilities. In: UbiComp 2002 Workshop on Collaboration with Interactive Tables and Walls, Göteborg, Sweden (2002)
10. Rekimoto, J.: SmartSkin: An infrastructure for freehand manipulation on interactive surfaces. In: Proc. CHI 2002, Göteborg, Sweden, pp. 113–120 (2002)
11. Sukthankar, R., Stockton, R., Mullin, M.: Automatic keystone correction for camera-assisted presentation interfaces. In: Tan, T., Shi, Y., Gao, W. (eds.) ICMI 2000. LNCS, vol. 1948, pp. 607–614. Springer, Heidelberg (2000)
12. Lien, C., Huang, C.: Model-based articulated hand motion tracking for gesture recognition. Image and Vision Computing 16(2), 121–134 (1998)
13. Triesch, J., Malsburg, C.: Robust classification of hand postures against complex background. In: Proc. International Conference On Automatic Face and Gesture Recognition, Killington (1996)
14. Segen, J.: Gesture VR: Vision-based 3D hand interface for spatial interaction. In: Proc. ACM Multimedia Conference, Bristol, UK, ACM Press, New York (1998)
15. Rehg, J., Kanade, T.: Digiteyes: Vision-based 3D human hand tracking. In: Technical Report CMU-CS-93-220, School of Computer Science, Carnegie Mellon University (1993)
16. Sato, Y., Kobayashi, Y., Koike, H.: Fast tracking of hands and fingertips in infrared images for augmented desk interface. In: Proc. International Conference on Automatic Face and Gesture Recognition, Grenoble, France (2000)
17. Crowley, J., Bérard, F., Coutaz, J.: Finger tracking as an input device for augmented reality. In: Proc. International Conference on Automatic Face and Gesture Recognition, Zürich, Switzerland (1995)
18. Laptev, I., Lindeberg, T.: Tracking of multi-state hand models using particle filtering and a hierarchy of multi-scale image features. In: Technical Report ISRN KTH/NA/P-00/12-SE, The Royal Institute of Technology (KTH), Stockholm, Sweden (2000)
19. Hardenberg, C., Brard, F.: Bare-hand human computer interaction. In: Proc. Perceptual User Interfaces, Orlando, Florida, USA (2001)
20. Microsoft Corporation: Microsoft Windows XP Tablet PC Edition 2005 Recognizer Pack, http://www.microsoft.com/downloads/details.aspx?familyid=080184dd-5e92-4464-b907-10762e9f918b&displaylang=en

A System for Hybrid Vision- and Sound-Based Interaction with Distal and Proximal Targets on Wall-Sized, High-Resolution Tiled Displays

Daniel Stødle, John Markus Bjørndalen, and Otto J. Anshus

Dept. of Computer Science, University of Tromsø, N-9037 Tromsø, Norway
{daniels,jmb,otto}@cs.uit.no

Abstract. When interacting with wall-sized, high-resolution tiled displays, users typically stand or move in front of it rather than sit at fixed locations. Using a mouse to interact can be inconvenient in this context, as it must be carried around and often requires a surface to be used. Even for devices that work in mid-air, accuracy when trying to hit small or distal targets becomes an issue. Ideally, the user should not need devices to interact with applications on the display wall. We have developed a hybrid vision- and sound-based system for device-free interaction with software running on a 7x4 tile 220-inch display wall. The system comprises three components that together enable interaction with both distal and proximal targets: (i) A camera determines the direction in which a user is pointing, allowing distal targets to be selected. The direction is determined using edge detection followed by applying the Hough transform. (ii) Using four microphones, a user double-snapping his fingers is detected and located, before the selected target is moved to the location of the snap. This is implemented using correlation and multilateration. (iii) 16 cameras detect objects (fingers, hands) in front of the display wall. The 1D positions of detected objects are then used to triangulate object positions, enabling touch-free multi-point interaction with proximal content. The system is used on the display wall in three contexts to (i) move and interact with windows from a traditional desktop interface, (ii) interact with a whiteboard-style application, and (iii) play two games.

Keywords: Vision- and sound-based interaction, large displays.

1 Introduction

Wall-sized, high-resolution displays are typically built by tiling displays or projectors in a grid. The displays or projectors are driven by a cluster of computers cooperating to produce a combined image covering the display wall, with existing display walls ranging in resolution from 10 to 100 Megapixels [8,16]. Our display wall combines 7x4 projectors back-projecting onto a non-rigid canvas to create a 220-inch display with a resolution of 7168x3072 pixels.

Display walls invite users to stand and move in front of them. To use input devices like mice and keyboards in this context, users must carry them around.

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 59–68, 2007.

Mice often require flat surfaces to be used, and even for "gyro-mice" and similar devices that don't have this requirement, the accuracy when trying to hit small or distal targets becomes an issue. We believe that ideally, users should not need devices to interact with applications running on the display wall.

We have developed a hybrid vision- and sound-based system that enables device-free interaction with software running on our display wall. The system analyzes and combines input from three main components to enable both distal and proximal interaction. The three components perform the following tasks, respectively: (i) Determine the direction in which a user is pointing his arm, (ii) determine the location at which a user snaps his fingers, and (iii) determine the location of objects (usually fingers or hands) in front of the wall. Figure 1 shows the system in use for playing two games.

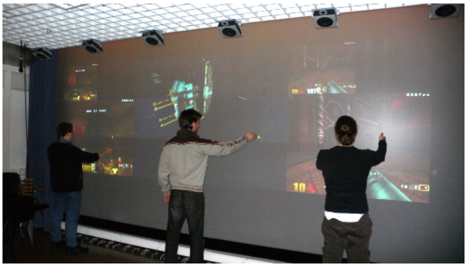


Fig. 1. The system in use for playing Quake 3 Arena and Homeworld

Each component contributes to a different part of the device-free workflow on the display wall. The first component lets a user select a distal target by pointing his arm towards it. A camera situated in the ceiling behind the user captures video. Each frame of the video is run through an edge detector, before the Hough transform is applied to determine the angle and position of the prevalent line segments visible in the video. Combining this with knowledge about what is currently visible on the display wall lets the component determine which target the user is attempting to select.

The second component lets the user bring the selected, distal target closer by double-snapping his fingers. Four microphones arranged in a rectangular fashion near the display wall stream audio to a computer. The computer correlates the audio samples with a template audio clip. When the user's snap is detected from at least three microphones, the snap's position can be determined using multilateration.

The third component lets the user interact with proximal targets. 16 cameras arranged in a row on the floor below the display wall's canvas stream data to 8 computers. When an object is visible from at least three cameras as it intersects a plane parallel to the display wall's canvas, its position can be determined using triangulation. Multiple objects can be detected and positioned simultaneously, enabling touch-free multi-point and multi-user interaction. We refer to it as touch-free as the user does not actually have to touch the display wall in order to interact with it. This is important in our context, as the display wall's canvas is flexible, and thus prone to perturbations when users touch it.

The main contribution of this paper is a hybrid vision- and sound-based system for interacting with both distal and proximal targets. We detail the system's implementation and show its use in three different contexts on a display wall:

(i) A traditional desktop interface, (ii) an experimental whiteboard-style application, and (iii) two games.

2 Related Work

Interacting with distal targets on very large displays is an important problem in HCI, and much previous work exists. In [18], the authors present a system that allows distant freehand pointing for interacting with wall-sized displays. Our system does not support pointing from afar, but lets users select distal targets while standing close to the display wall. Our system detects the angle at which a user points his arm, while the authors of [18] use a commercial system that requires the user to wear markers. There are numerous other techniques for reaching distal targets described in the literature; they include drag-and-pop [1], the Vacuum [2] and the Frisbee [7]. These techniques generally work by temporarily bringing distal targets closer. Our approach is complementary in this regard, as distal targets are moved semi-permanently (that is, until the user decides to move them again). The "tablecloth" is an entirely different approach that lets the user scroll the desktop much as he would scroll a window [13].

In [14], untagged audio is used for interacting with a 3D interface connected to a media player. The system recognizes loud sounds above a dynamic threshold, such as snapping fingers, and in turn uses this to determine the position of the sound. Their work focuses on creating 3D interfaces, by creating and placing virtual "buttons" in space. Rather than creating buttons with fixed locations, we utilize the user's actual position to bring distal targets closer to the user. In [6], the authors propose using continuous vocal sounds for controlling an interface, rather than interpreting spoken commands. They do not attempt to determine the user's physical location, while one key capability of the snap-detecting component of our system is that it allows applications to leverage this information. Using vocal sounds to control the position of a cursor is proposed in [9]. This work could conceivably be used to act on distal targets, and indeed one aspect of moving the cursor often includes moving windows. However, no attempt is made at determining the user's location and using this information to improve interaction.

Sound source localization is much used in the field of robotic navigation. In [17], a system is described whereby 8 microphones are used to reliably locate sounds in a 3D environment using cross-correlation and time-delay. The technique used by the snap-detecting component is similar in that it estimates the time-delay between incoming snaps, but differs in that it uses the located sound for interacting with a user interface, rather than interacting with a robot. Our component only attempts to detect snaps, whereas the system in [17] does not discriminate between different sounds. Rather, it is used to locate all "interesting" sound sources, in order to focus the robot's attention towards the sound source.

Much research has been done on systems for supporting multi-touch and multi-point interaction. The Diamondtouch [3] tabletop is one approach, where the position of touches is detected using electric capacitance. Other technologies include [5], where infrared light is projected into a canvas and internally reflected. The light escapes the canvas at points where the user is touching, which can be detected using a camera. Our system is based on detecting the presence of objects directly using cameras, and does not require the user to actually touch the display wall's canvas. In [10], the author presents a camera-based solution to detecting and positioning objects in front of a whiteboard. They use custom cameras with on-chip processing to perform object recognition, while we take a parallel and commodity-based approach with 16 cheap cameras connected to 8 computers.

3 Design

The hybrid vision- and sound-based system is comprised of three components and a custom event delivery system. The three components are (i) arm-angle, (ii) snap-detect and (iii) object-locator. Each component is responsible for detecting different user actions, with the event system providing communication between the components and end-user applications[1]. Figure 2 illustrates the overall design. The purpose of the arm-angle and snap-detect components is to let the user select and access distal targets, while the object-locator enables the user to interact with proximal targets using single- or multi-point interaction. Figure 4 illustrates an example scenario where each component is used in turn to select a distal target, move it closer to the user and then interact with it.

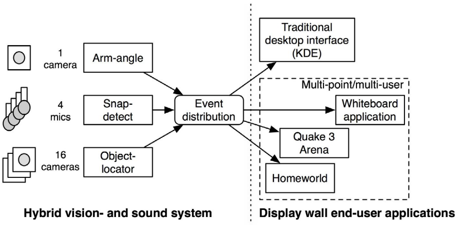


Fig. 2. The system design with three components and event distribution

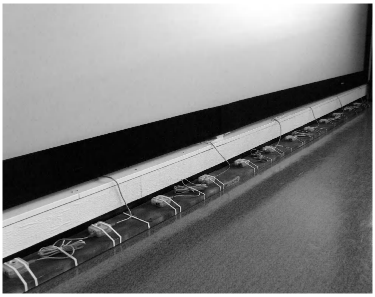


Fig. 3. 16 cameras along the floor enable object detection in front of the display wall

The arm-angle component processes images streamed from a single camera, with the purpose of identifying the angle at which a user is pointing. The camera is mounted in the ceiling, looking at the backs of users interacting with the display wall. The component delivers events to end-user applications, which interpret them according to their needs and current state.

The snap-detect component performs signal processing on audio from four microphones. The microphones are arranged in a rectangle in front of the display wall, with two microphones mounted near the ceiling at op-

[1] The event system is outside the scope of this paper.

posite ends of the canvas, and the other two near the floor. The component's goal is to detect a user snapping his fingers or clapping his hands in at least three of the four audio streams, allowing the signal's origin in 2D space to be determined using multilateration.

Multilateration uses the difference between the time at which snaps are detected by the different microphones to determine possible locations where the user snapped his finger. With a fixed sample rate, it is possible to determine the approximate, relative distance the sound has travelled from the time it was detected by a given microphone until it was detected by another. The resulting points lie on a hyperbolic curve with focus point at the given microphone's position. For many microphones, the intersections of the resulting curves can be used to determine the location of the user in 2D. For each positioned snap, an event containing the snap's 2D location and strength is delivered to end-user applications.

The object-locator component uses 16 cameras to detect objects intersecting a plane parallel to the display wall's canvas, with typical objects being a user's fingers or hands. The cameras are connected pairwise to 8 computers, and mounted in a row along the floor looking at the ceiling (Figure 3). When three or more cameras see the same object, its position can be determined using triangulation. The component can detect several objects simultaneously. If the end-user application supports it, both multi-user and multi-point interaction is possible.

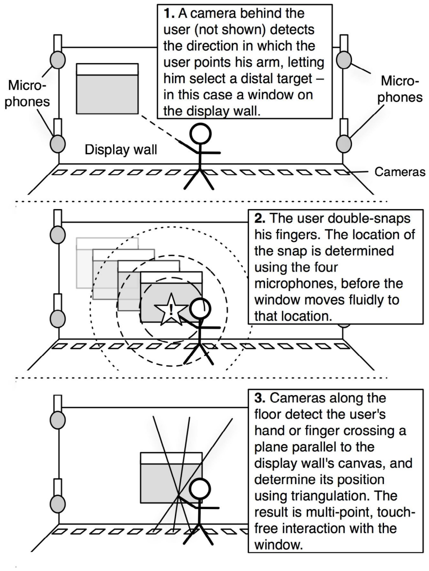

1. A camera behind the user (not shown) detects the direction in which the user points his arm, letting him select a distal target – in this case a window on the display wall.
Micro-phones
Micro-phones
Display wall
Cameras
2. The user double-snaps his fingers. The location of the snap is determined using the four microphones, before the window moves fluidly to that location.
3. Cameras along the floor detect the user's hand or finger crossing a plane parallel to the display wall's canvas, and determine its position using triangulation. The result is multi-point, touch-free interaction with the window.

Fig. 4. The user selects a window, double-snaps to bring it closer, and interacts using the touch-free interface

4 Implementation

The arm-angle component, implemented in C, uses a Canon VC-C4R pan-tilt-zoom camera connected to a framegrabber-card on a computer running Linux. The component captures frames at 8-9 frames per second (FPS). Each frame has a resolution of 720x540 pixels and is scaled down to half-size and converted from color to grayscale. Then the Sobel edge detector is used to locate edges in the image, before the equations of lines appearing in the image are determined using the Hough transform [4]. End-user applications receive events from the arm-angle component, which they use to determine the targets that the user is trying to select. For the traditional desktop interface, this is done by determining the bounding rectangles of currently visible windows, and then intersect each window's bounds with the line indicating the user's pointing direction. Since the Hough-transform typically yields several potential lines for the arm and other straight edges, each line votes for the closest window it intersects with. A red square over the selected window highlights it to the user. The traditional desktop interface on the display wall is created using Xvnc [12,11], with each tile of the display wall showing its respective part of the entire, high-resolution desktop.

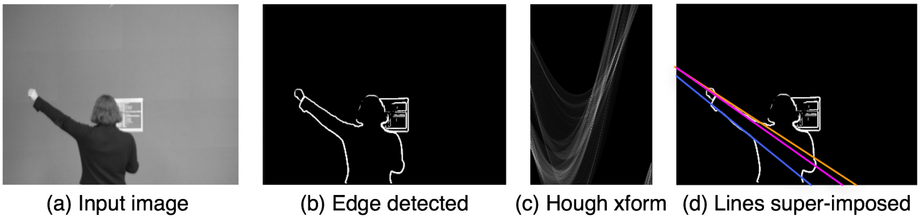

(a) Input image
(b) Edge detected
(c) Hough xform
(d) Lines super-imposed

Fig. 5. The arm-angle image processing steps. (a) The input image, (b) the image after edge detection, (c) output from the Hough-transform, (d) the edge-detected image with lines extracted from the Hough-transform super-imposed.

The snap-detect component uses four microphones connected to a mixer. The mixer feeds the audio to a PowerMac G5, which uses a Hammerfall HDSP 9652 sound card to capture 4-channel audio samples at a rate of 48000 Hz. Samples from each channel are correlated with a template sound of a user snapping his fingers. When the correlation for a given channel exceeds an experimentally determined threshold, a snap is detected in that channel. The snap-detect component records a timestamp (essentially a sample counter) for each channel the snap is detected in. When a snap is detected in at least three channels, the resulting timestamps can be used to determine the user's location using the difference in time between when the snap is detected by the first microphone, and when it is detected by the remaining two or three microphones (multilateration). The location is determined by the intersection of conics created from different pairs of microphones, illustrated in Figure 7. For the desktop interface on the display wall, the selected window is moved to the location of the snap over a period of half

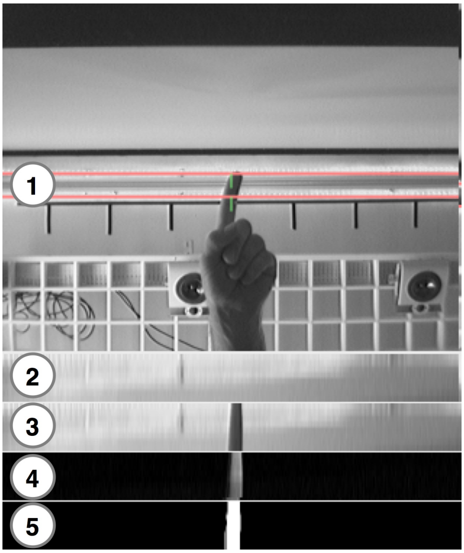


Fig. 6. (1) Image from camera. (2) Current background image. (3) Area of interest from current camera image. (4) The result by subtracting (3) from (2). (5) Result after thresholding the image.

a second. Changes to existing applications are not necessary to allow window movement by snapping fingers.

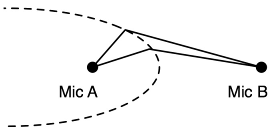


Fig. 7. The possible positions given the time difference between mic A and B are all located on the conic section.

The object-locator component is implemented in Objective-C, and consists of two modules: An image processing module, and a triangulation module. The image processing module runs in parallel on 8 Mac minis. Each Mac mini captures streaming video from two Unibrain Fire-i FireWire cameras in 640x480 grayscale at 30 FPS. Figure 6 illustrates the image processing done for a single camera. When the image processing module on a Mac mini starts up, it sets the first grabbed image to be the background image; the background is subsequently updated dynamically to deal with changing light conditions. For each new image captured, a narrow region of interest is isolated, before the background is subtracted from it. The result is then thresholded, yielding one-dimensional object positions and radii wherever pixel values exceed the dynamic threshold (the radius is set to the number of successive pixels above the threshold divided by two). The one-dimensional positions and radii are gathered by the triangulation module. For each camera, the triangulation module creates line segments that start at the camera's position, and pass through the center of each object detected by that camera. These lines are then intersected with the lines generated for the two cameras to the immediate left and right of the current camera. The resulting intersections are examined, and if two or more intersection points from three different cameras lie sufficiently close, an object is detected, as illustrated in Figure 8. The traditional desktop interface uses the 2D positions reported by the object-locator to control the cursor, and uses the radius to determine if the user wants to click.

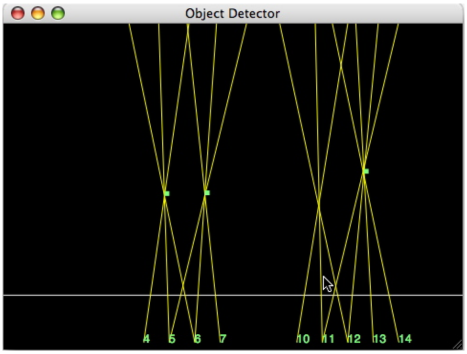


Fig. 8. The object-locator identifying three of four objects and their position

5 Early Deployment

The hybrid vision- and sound-based interface has been deployed in three different contexts: (i) A traditional desktop interface running on the display wall, (ii) a prototype whiteboard-style application supporting multiple users, and (iii) two previously commercial, but now open-source games (Quake 3 Arena and Homeworld). When used with a traditional desktop interface, the system enables one user to select distal windows on the display wall, and move them closer by double-snapping his fingers. When the window is within reach, the user can interact with it using the touch-free interface. The cursor tracks the position of the user's hand or finger, and a click or drag can be invoked by tilting the hand (making it flat), as illustrated in Figure 9. Its use in the whiteboard-application is similar, where the distal objects drawn on screen can be brought closer by pointing at them and then double-snapping. In addition, the whiteboard brings up a tool palette at the user's location when a user single-snaps, allowing new objects to be added. The touch-free interface is used to support multi-user and multi-point interaction, for instance allowing users to resize objects by varying the distance between their hands.

The final context in which the system has been used is to play two games, Quake 3 Arena and Homeworld, further detailed in [15]. In this context, we only utilize the object-locator component, and to some extent the snap-detect component (people not playing the game can make the players fire their weapons in Quake 3 Arena by snapping their fingers, for instance). The touch-free interface provided by the object-locator enables users to play the two games using gestures only, rather than using traditional mouse/keyboard-based input.

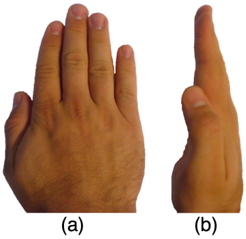


Fig. 9. (a) A flat and (b) vertical hand

6 Discussion

One principle employed throughout the design of the system is the following: Rather than identify what objects are, it is more important to identify where the objects are. The system at present does not distinguish between an arm pointing to give a direction, or a stick being used to do the same; nor does the system distinguish a snap from a clap or snap-like sounds made by mechanical clickers. This principle is also used to realize the touch-free interface - users can interact using their fingers, hands, arms, pens or even by moving their head through the (invisible) plane in front of the display wall.

This principle is not without issues, however, as false positives can occur for all of the components used in the system. For the arm-angle component, one issue is that the content currently on the display wall interferes with the edge-detection and the following Hough-transform to produce results that do not reflect the user's intentions. Another issue is that the resolution offered by the arm-angle component is too coarse to be used for selecting very small targets, such as icons. Techniques like drag-and-pop or the vacuum may be better suited for picking out small targets [1,2]. For the object-locator, the fact that it does not recognize what objects are means that it has no way of distinguishing several users from a single user interacting with several fingers or both hands. Although our ideal is to provide device-free interaction with the display wall for multiple, simultaneous users, the only component of the system that currently supports multiple simultaneous users is the object-locator. We are currently working on ways to support multiple users with the arm-angle and snap-detect components as well. We are also investigating the accuracy of the three components, to better characterize their performance.

7 Conclusion

This paper has presented a system combining techniques from computer vision and signal processing to support device-free interaction with applications running on high-resolution, wall-sized displays. The system consists of three components utilizing in total 17 cameras and 4 microphones. It enables a user to select distal targets by pointing at them, bring the targets closer by double-snapping his fingers and finally interact with them through the use of a touch-free, multipoint interface. The system does not require the user to wear gloves or use other external devices. Instead, the system has been designed so as not to care exactly *what* a detected object is, but rather *where* the object – whatever it may be – is located. This in turn means that the system does not attempt to tell different users apart, for either of the three components. The interface has been deployed in three different contexts on the display wall: (i) One user at a time can interact with a traditional desktop interface, (ii) several users can interact simultaneously with a custom whiteboard-style application, with the caveat that the distal target selector only works for the user positioned near the center of the display wall, and (iii) up to three persons may play the games Quake 3 Arena and Homeworld simultaneously.

Acknowledgements

The authors thank Espen S. Johnsen and Tor-Magne Stien Hagen for their discussions, as well as the technical staff at the CS department at the University of Tromsø. This work has been supported by the Norwegian Research Council, projects No. 159936/V30, SHARE - A Distributed Shared Virtual Desktop for Simple, Scalable and Robust Resource Sharing across Computer, Storage and Display Devices, and No. 155550/420 - Display Wall with Compute Cluster.

References

1. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B., Zierlinger, A.: Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems. In: Proceedings of Interact 2003, pp. 57–64 (2003)
2. Bezerianos, A., Balakrishnan, R.: The vacuum: facilitating the manipulation of distant objects. In: CHI 2005. Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 361–370. ACM Press, New York (2005)
3. Dietz, P., Leigh, D.: DiamondTouch: a multi-user touch technology. In: UIST 2001. Proceedings of the 14th annual ACM symposium on User interface software and technology, pp. 219–226. ACM Press, New York (2001)

4. Duda, R.O., Hart, P.E.: Use of the hough transformation to detect lines and curves in pictures. Commun. ACM 15(1), 11–15 (1972)
5. Han, J.Y.: Low-cost multi-touch sensing through frustrated total internal reflection. In: UIST 2005. Proceedings of the 18th annual ACM symposium on User interface software and technology, pp. 115–118. ACM Press, New York (2005)
6. Igarashi, T., Hughes, J.F.: Voice as sound: using non-verbal voice input for interactive control. In: UIST 2001. Proceedings of the 14th annual ACM symposium on User interface software and technology, pp. 155–156. ACM Press, New York (2001)
7. Khan, A., Fitzmaurice, G., Almeida, D., Burtnyk, N., Kurtenbach, G.: A remote control interface for large displays. In: UIST 2004. Proceedings of the 17th annual ACM symposium on User interface software and technology, pp. 127–136. ACM Press, New York (2004)
8. Li, K., Chen, H., Chen, Y., Clark, D.W., Cook, P., Damianakis, S., Essl, G., Finkelstein, A., Funkhouser, T., Housel, T., Klein, A., Liu, Z., Praun, E., Samanta, R., Shedd, B., Singh, J.P., Tzanetakis, G., Zheng, J.: Building and Using A Scalable Display Wall System. IEEE Comput. Graph. Appl. 20(4), 29–37 (2000)
9. Mihara, Y., Shibayama, E., Takahashi, S.: The migratory cursor: accurate speech-based cursor movement by moving multiple ghost cursors using non-verbal vocalizations. In: Assets 2005. Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility, pp. 76–83. ACM Press, New York (2005)
10. Gerald, D.: A camera-based input device for large interactive displays. IEEE Computer Graphics and Applications 25(4), 52–57 (2005)
11. RealVNC, Ltd. VNC for Unix 4.0. `http://www.realvnc.com/`
12. Richardson, T., Stafford-Fraser, Q., Wood, K.R., Hopper, A.: Virtual Network Computing. IEEE Internet Computing 2(1), 33–38 (1998)
13. Robertson, G., Czerwinski, M., Baudisch, P., Meyers, B., Robbins, D., Smith, G., Tan, D.: The large-display user experience. IEEE Comput. Graph. Appl. 25(4), 44–51 (2005)
14. Scott, J., Dragovic, B.: Audio Location: Accurate Low-Cost Location Sensing. In: Gellersen, H.-W., Want, R., Schmidt, A. (eds.) PERVASIVE 2005. LNCS, vol. 3468, pp. 1–18. Springer, Heidelberg (2005)
15. Stødle, D., Hagen, T.-M.S., Bjørndalen, J.M., Anshus, O.J.: Gesture-based, touch-free multi-user gaming on wall-sized, high-resolution tiled displays. In: Proceedings of the 4th Intl. Symposium on Pervasive Gaming Applications, PerGames 2007, pp. 75–83 (June 2007)
16. Stolk, B., Wielinga, P.: Building a 100 Mpixel graphics device for the OptIPuter. Future Gener. Comput. Syst. 22(8), 972–975 (2006)
17. Valin, J.-M., Michaud, F., Rouat, J., Letourneau, D.: Robust sound source localization using a microphone array on a mobile robot. In: Proceedings of Interation Conference on Intelligent Robots and Systems (IROS), vol. 2, pp. 1228–1233 (October 2003)
18. Vogel, D., Balakrishnan, R.: Distant freehand pointing and clicking on very large, high resolution displays. In: UIST 2005. Proceedings of the 18th annual ACM symposium on User interface software and technology, pp. 33–42. ACM Press, New York (2005)

Real-Time Automatic Kinematic Model Building for Optical Motion Capture Using a Markov Random Field

Stjepan Rajko and Gang Qian

Arts, Media and Engineering Program, Arizona State University, Tempe AZ 85287, USA
srajko@asu.edu, gang.qian@asu.edu

Abstract. We present a completely autonomous algorithm for the real-time creation of a moving subject's kinematic model from optical motion capture data and with no a priori information. Our approach solves marker tracking, the building of the kinematic model, and the tracking of the body simultaneously. The novelty lies in doing so through a unifying Markov random field framework, which allows the kinematic model to be built incrementally and in real-time. We validate the potential of this method through experiments in which the system is able to accurately track the movement of the human body without an a priori model, as well as through experiments on synthetic data.

1 Introduction

Body movement data is used for many applications, such as rehabilitation, kinesiology, sports training, animation, and interactive arts. With the recent research efforts on the analysis of human gesture and movement, motion capture systems are also promising to become an important part of human-computer interaction systems.

However, when using marker-based optical motion capture, there are obstacles to creating a seamless interactive environment. The placement of markers on the user is intrusive, and usually requires time and care to be placed properly. Furthermore, the system requires a training phase in which it builds a model describing the marker properties and their placement on a particular user. Figure 1 shows a model that has been manually entered into a Motion Analysis Corporation optical motion capture system.

To make movement-based human-computer interaction more streamlined, the research community has made improvements both in automatizing parts of the marker-based capture process, and on improving markerless motion capture methods. In this paper, we contribute to the automatization of marker-based motion capture through a method which eliminates the need for a detailed marker set model and subject calibration, and instead provides the configuration of the body purely by observing marker movement. The method is based on Markov random fields, and allows the model building, marker tracking, joint position inference, and body tracking to be achieved simultaneously, automatically, and in real-time. Additionally, the general probabilistic framework we propose may be extended to markerless motion capture systems in the future.

The paper is organized as follows. In Section 2, we discuss some of the related work, then give a theoretical presentation of our framework in Section 3. The results of its implementation on some real-world and synthetic data are presented in Section 4, and conclusions in Section 5.

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 69–78, 2007.

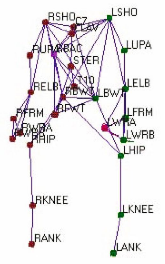


Fig. 1. A manually entered model, with the markers named according to body placement. Pairs of markers expected to remain at a relatively constant distance have been manually connected.

2 Previous Work

The progression from collected raw point data to a high-level understanding of the movement is divided into different stages [1]. *Tracking* refers to making correspondences between observed points across frames. The final goal is to obtain the paths of markers throughout the trial. *Labeling* corresponds to assigning labels to each of the observed paths. In a system that uses pre-existing models of marker configurations, the goal is to transfer the available marker labels (such as “Left Shoulder”, or “Right Knee”) from the model to the observed data. Otherwise, the goal would be to build a model (*modeling*) from the observed data, and maintain it as the subject leaves and re-enters the space. Finally, *pose estimation* produces a high-level representation of the motion data, such as the joint angles of the skeletal structure.

There are a number of methods which produce a model of the body, but they require tracked feature / marker data to be provided. Recently, Yan and Pollefeys [2] presented a method which computes the kinematic chain of an articulated body from 2D trajectories of feature points. Silaghi et al. [3] also give a method which accomplishes automatic model building when given tracked motion capture data as input. In addition to building a model of the skeleton, they present the motion in terms of the skeleton. In some methods, tracking is incorporated in the model building, such as the approach taken by Ringer and Lasenby [4], but the produced model is not used to improve the tracking. However, using an a priori model to help with the tracking process has been considered, for example by Zakotnik et al. [5] and Karaulova et al. [6]. Finally, the isolated problem of marker tracking or feature tracking has been studied extensively in both 2D and 3D domains. A good review of some methods is provided by Veenman el al. [7]. Shafique and Shah [8] also give a graph based multi-frame approach.

Recently, Hornung et al. [9], in conjunction with a very nice explanation of the problem and goals which are very much in line with ours, proposed a method which ties all of the above subproblems together and gives promising results. Rajko and Qian [10] also presented a heuristic approach that unifies marker tracking and some model building. Our proposed method differs in that it brings together all of the elements of the problem into a unifying probabilistic framework. This not only provides a more compact solution, but also allows probabilistic inference of the underlying (hidden) ground truth data which can give more accurate results.

3 The MRF-Based Framework

Our framework captures the behavior of the body under observation using a series of *models* that specify this behavior at different levels of abstraction. At the highest level, we model the body as a collection of *rigid objects* that are connected by *joints*. At the lower level, we model individual *markers*, their corresponding *observations* provided by the motion capture system, as well as pairs of markers that appear to be connected by a *rigid link*.

The body models can be specified in a little more detail as follows:

- The *observation model* defines the relationship between a marker's true position and its observation provided by the motion capture system.
- The *single point model* defines the behavior of a single marker and assumes the motion of the marker is relatively smooth, i.e. its velocity does not change much from frame to frame.
- The *rigid link model* specifies that that the distance between markers that lie on the same rigid object should remain relatively constant.
- The *rigid object model* specifies the behavior of all markers on a rigid object in terms of the behavior of the object and the relative placement of the markers on the object.

The observations of the markers are the only observable element of the body - the existence of all other higher level models is inferred by monitoring the behavior of these observations. As the higher level models are constructed, we can use them to provide a more accurate state of the body, as well as perform more accurate marker tracking. Correspondingly, the set of models in the system is not static. It may expand as the system becomes aware of higher level models as they are inferred, or new markers/bodies enter the scene, or shrink as the models cease to be accurate or leave the scene.

To combine the various models, we will make use of a Markov random field graph, in which the various models will exhibit themselves through the nodes and edges of the graph. The following section will introduce some assumptions and the notation we use in the remainder of the paper. Due to the page limit, we will present only a part of the model in detail.

3.1 Assumptions and Notation

We assume to have available a motion capture system capable of providing 3D observations of markers attached to the objects of interest. The system provides instantaneous observation information at a constant frame rate, and for simplicity we assume that this information is provided at times $t = 1, 2, \ldots$ which we also refer to as frames. In our notation, we will consistently use (t) in the superscript to indicate the time.

We denote *the set of observations* at time t by $Y^{(t)} = \{y_1^{(t)}, y_2^{(t)}, \ldots, y_{|Y^{(t)}|}^{(t)}\}$. Each $y_i^{(t)}$ is a three dimensional point given in the motion capture reference coordinate system. Please note that the observations are noisy, i.e. the true position of the markers is hidden. Also, markers can be occluded (not producing an observation), and we sometimes observe “ghost markers” (observations that do not correspond to a marker).

Since objects of interest can leave and enter the motion capture volume, the set of markers observed by the system at each time point is not necessarily always the same. We therefore define a marker index set M that encompasses all markers that are a part of the system at any time of the capture trial, and the entire marker set as $X = \{X_m | m \in M\}$. We can then define *the set of markers* in the system at time $t = 1, 2, \ldots$, denoted by $X^{(t)} = \{X_m | X_m \text{ is in the system at time } t\}$. Due to occluded markers or ghost markers, it is possible that $\left|Y^{(t)}\right| \neq \left|X^{(t)}\right|$.

We define the state of a marker m at time t, $X_m^{(t)} = (x_m^{(t)}, \dot{x}_m^{(t)}, \sigma_m^{(t)})$, by its true position $x_m^{(t)}$, velocity $\dot{x}_m^{(t)}$, and uncertainty in the movement characterized by $\sigma_m^{(t)}$. We discretize the velocity of the marker in the sense that

$$\dot{x}_m^{(t')} = \int_{t=t'-1}^{t'} \frac{dx_m^{(t)}}{dt} dt = x_m^{(t')} - x_m^{(t'-1)} \tag{1}$$

The markers are related to the observations by an (unobservable) function $f^{(t)} : X^{(t)} \to Y^{(t)} \cup \{\zeta\}$, where $f^{(t)}(X_m^{(t)}) = y_i^{(t)}$ whenever observation $y_i^{(t)}$ corresponds to marker $X_m^{(t)}$, and $f^{(t)}(X_m^{(t)}) = \zeta$ whenever marker $X_m^{(t)}$ is occluded.

The *set of rigid objects* is denoted by as $R^{(t)}$. We will not define the rigid objects in detail, but only mention that a rigid object is modeled by its position and orientation, as well as (discretized) translational and rotational velocity.

3.2 The Markov Random Field Graph

The models themselves are embedded in a *Markov random field* (MRF). An MRF can be modeled by a graph over a set of random variables in which edges represent interdependencies between random variables. These interdependencies can be expressed in terms of functions (*potentials*) over cliques of the graph, which are then combined to formulate the joint probability of the random variables.

In our case, the random variables in the MRF represent the following entities:

- *The set of observations* $Y^{(t)}$ for all t.
- *The set of markers* $X^{(t)}$ for all t.
- *The set of rigid objects* $R^{(t)}$ for all t.

Each of these is modeled as a node in the MRF graph. The edges in the graph represent different interdependences. We will define these nodes and edges in such a way that the only potentials defined in the system exist over cliques of size 2 as described below. We will denote the set of all two-node cliques by C, and the set of all two-node cliques for a particular frame as $C^{(t)}$. $c \in C^{(t)}$ if and only if at least one node in c is from frame t, and both nodes in c come from either frame t or frame $t-1$. We will write all potentials as $V_c^{(t)}$, where $c \in C^{(t)}$ (see below).

An edge exists between a marker and its corresponding observation. Let p_ζ be the probability that a marker is occluded and $p_\gamma = 1 - p_\zeta$ the probability that a marker is visible. We characterize the noise of the observations through a probabilistic function as follows:

$$V^{(t)}_{X_m^{(t)}, f(X_m^{(t)})} = \begin{cases} p_\gamma \mathcal{N}(x_m^{(t)}, \sigma_o; f(X_m^{(t)})) & \text{if } f(X_m^{(t)}) \in Y^{(t)} \\ p_\zeta & \text{if } f(X_m^{(t)}) = \zeta \end{cases} \tag{2}$$

Similarly, an edge exists between a marker at time $t-1$ ($X_m^{(t-1)}$) and time t ($X_m^{(t)}$) to capture the probabilistic relationship between a marker's position at two adjacent frames. The corresponding potential is defined as

$$V^{(t)}_{X_m^{(t-1)},X_m^{(t)}} = \mathcal{N}(\dot{x}_m^{(t-1)} + x_m^{(t-1)}, \sigma_{pm}^{(t-1)}; x_m^{(t)}) \tag{3}$$

If two markers lie on the same rigid object (e.g., rigid body part), we say that the two markers are connected by a rigid link, and employ the *rigid link* model which gives constraints on the distance between two markers. The rigidity of the link dictates that the distance between such a pair of markers should stay relatively constant:

$$V^{(t)}_{X_m^{(t)},X_n^{(t)}} = \mathcal{N}(d_{mn}^{(t)}, \sigma_{mn}^{(t)}; x_m^{(t)} - x_n^{(t)}), \tag{4}$$

where $d_{mn}^{(t)}$ is the expected length of the link between markers $X_m^{(t)}$ and $X_n^{(t)}$ at time t and $\sigma_{mn}^{(t)}$ its standard deviation.

Please note that in general, parameters such as $d_{mn}^{(t)}$ and $\sigma_{mn}^{(t)}$ should not vary much with time. However, it is possible that markers change their relative position on the body during the capture (for example, a marker placed at the end of a sleeve of a tight shirt will permanently come closer to the elbow if the arms reach overhead and cause the sleeve to ride up the arm). For this reason, we allow the properties of the link to vary with time.

Additional potentials, which we will not explain in detail, are those modeling the change of the rigid object state from one frame to the next (similar to the single point model, but for rigid bodies), and those modeling the position of a marker that is a part of a rigid body (similar to the rigid link model, but relating the configuration of the rigid body and the inferred body placement of the marker with the actual marker position).

Figure 2 shows a diagram of two time slices of the MRF graph. Each time slice contains nodes for that particular time (the markers, rigid objects and observations). Edges between the nodes either connect nodes of the same time slice, or nodes from adjacent time slices.

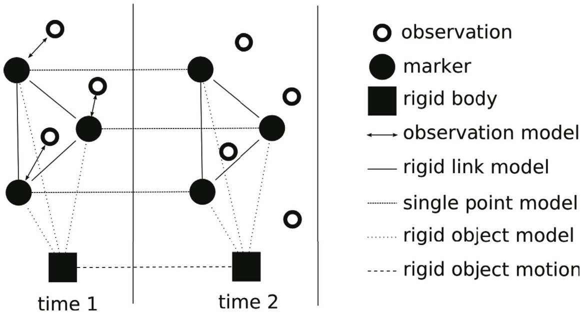


Fig. 2. A diagram of two time slices of the MRF graph. Markers at time 2 have not yet been assigned to observations.

3.3 Constructing the MRF Graph

At each time t, the marker states $X^{(t)}$, and the rigid object states $R^{(t)}$ comprise the state of the system $S^{(t)} = \{X^{(t)}, R^{(t)}\}$. Our goal will be to construct the most likely estimate of the system to have produced the observations at each frame, i.e. we are trying to find $S^{(1)}, S^{(2)}, \ldots$ s.t. $p(Y^{(1)}, Y^{(2)}, \ldots, S^{(1)}, S^{(1)}, \ldots) = P(Y, S)$ is maximized.

The probability $P(Y, S)$ can be broken down as follows:

$$P(Y,S) \propto \prod_{c \in C} V_c = \prod_{t} \prod_{c \in C^{(t)}} V_c^{(t)} = \prod_{t} P^{(t)} \tag{5}$$

where $P^{(t)} = \prod_{c \in C^{(t)}} V^{(t)}$. We divide the problem of maximizing $P(Y, S)$ by maximizing $P^{(t)}$ for each t as much as possible.

You can see an outline of the entire algorithm in Figure 3.

```
for each frame $t$ do
    Create a new marker from each observation $y_i^{(t)}$
    Create a marker $X_m^{(t)}$ for each marker $X_m^{(t-1)}$
    Compute $f$
    Erase new markers that are unassigned, and markers that have been occluded too long
    for frames $t$, $t-1$, and $t-2$ do
        Update MRF parameters from newly observed values
        Update MRF random variables using updated parameters
    end for
    Watch for new models (rigid links, rigid objects, joint positions...)
end for
```

Fig. 3. Marker to point assignment algorithm

To initialize the MRF graph, we start with the marker observations provided for frame 1, $Y^{(1)}$. For each observation $y_i^{(1)}$ we create a marker model $X_m^{(1)}$ with $m = i$, and set $x_m^{(t)} = y_i^{(t)}$ and $\dot{x}_m^{(t)} = (0, 0, 0)$. For each incoming frame of observations $Y^{(t)}$, we similarly create new marker models for each observation. We also extend each marker in $X^{(t-1)}$ to $X^{(t)}$ by initializing $x_m^{(t)}$ to the most likely position given the single point model, and computing $\dot{x}_m^{(t)}$ correspondingly.

The first task for a given frame is to assign markers to observations, i.e. find the mapping $f^{(t)}$. We start with $f^{(t)}(X_m^{(t)})$ undefined for all $X_m^{(t)}$, and then assign $f^{(t)}(X_m^{(t)})$ to an appropriate $y_i^{(t)}$ or ζ for each $X_m^{(t)}$ in turn. We do so using a greedy approach based on confidence, such as the one used in [10]. Essentially, for each marker m and each observation i, the algorithm maximizes the value of the potentials associated with the marker, with the conjecture that $f^{(t)}(X_m^{(t)}) = y_i^{(t)}$. Once an assignment of a marker to an observation (or occlusion) has been determined, we call that marker *assigned*, and until then it we call it *unassigned*. Similarly, a rigid object is considered assigned if and only if at least three of the markers associated with the rigid object have been assigned. A two node clique c is assigned if both of its nodes are assigned.

We now define the potential of a marker $X_m^{(t)}$ as

$$P^{(t)}(X_m^{(t)}) = \prod_{c \in C^{(t)}, X_m^{(t)} \in c} V_c^{(t)}. \tag{6}$$

Additionally, we define the potential of $X_m^{(t)}$ relative to an observation $y_i^{(t)}$, which conjectures that $f^{(t)}(X_m^{(t)}) = y_m^{(t)}$, and then limits the potential $P^{(t)}(X_m^{(t)})$ to only cliques that have been assigned:

$$P_{y_i^{(t)}}^{(t)}(X_m^{(t)}) = \prod_{c \in C^{(t)}, c \text{ assigned}, X_m^{(t)} \in c} V_c^{(t)}. \tag{7}$$

Determining f then goes as follows. For each marker/observation pair, we compute $A_{mi} = \max P_{y_i^{(t)}}^{(t)}(X_m^{(t)})$ using a gradient ascent method on $P_{y_i^{(t)}}^{(t)}(X_m^{(t)})$ by manipulating $x_m^{(t)}$. In other words, we find the most likely location of marker $X_m^{(t)}$ given the potentials with other assigned markers or rigid objects, as well as the observation model relative to $y_i^{(t)}$.

This defines a matrix $A = \{A_{mi}\}$. At this point, we choose an assignment that can be made with a degree of confidence and unambiguity. Define $p_{maxi(m)}$ to be the best and $p_{maxi'(m)}$ the second best match for $X_m^{(t)}$ according to the values in A. I.e., $maxi(m) = \max \arg_j A_{mj}$, and $maxi'(m) = \max \arg_{j \neq maxi(m)} A_{mj}$.

A confident assignment would be one where the potential of the best match is large, as is the ratio with the second best match. Hence, for each marker $X_m^{(t)} \in X^{(t)}$ we define the confidence factor $conf(X_m^{(t)}) = A_{m,maxi(i)} \times \frac{A_{m,maxi(i)}}{A_{m,maxi'(m)}}$. The confidence factor indicates how confidently we can make a correspondence between $X_m^{(t)}$ and $y_{maxi(m)}$.

We choose a marker $X_m^{(t)}$ for which $conf(X_m^{(t)})$ is maximal, and set $f(X_m^{(t)}) = y_{maxi(m)}$. Since a marker has been assigned, we update A for all unassigned markers that are affected by the newly assigned marker. Finally, all affected confidence factors are recalculated, and we can now make the the next correspondence for the unassigned marked with the maximal confidence factor.

The process is repeated until all markers are assigned. Markers that have been created in the current frame (using an observation) that are marked as occluded (because an older marker was assigned to that same observation) are immediately erased. Markers that appear to have left the system (by being occluded for too long) are also erased.

At this point, for the last 3 frames we update the estimates of various model parameters (lengths of rigid links, local coordinates of markers on rigid objects, etc.), and maximize the potentials dependent on the random variables in the system (positions of markers, positions and orientations of rigid bodies, etc.), which improves the tracking results with minimal latency. We do so by adapting the expectation-maximization (EM) algorithm to the real-time setting, on which we will not go into detail.

Finally, after the adapted EM, we watch for existence of new models in the system, such as rigid links and rigid objects. In most cases we do so by randomly choosing candidates to observe (e.g., a pair of markers that have not been examined before), and inspect their history in the MRF graph to detect whether their behavior satisfies a particular model (e.g., their distance remains relatively constant).

4 Experimental Results

We will first present the results of running the algorithm on two motion capture trials to demonstrate the general model-building capabilities of the algorithm. Figure 4 shows the model built automatically from a 20 second full-body capture sequence, in which the subject was asked to move using all of their joints. You will notice that most of the rigid links (connections) that had to be entered manually for the model in Figure 1 are automatically found. The one exception is a wrist marker on the left arm, which remained unconnected because it was particularly noisy.

Also, the proposed algorithm successfully partitioned the markers into rigid objects corresponding to the two shoulders, two lower arms, the pelvic area, and the two legs. The few errors concern markers that are placed close to a rigid object but are not a part of it. For example, the right upper arm marker was assigned to the right shoulder, and the left upper arm marker was assigned to the left lower arm. Similarly, even though the markers on the legs were actually placed on both the upper leg and the lower leg, the algorithm assigned them to the same rigid object because the range of motion sequence did not involve much bending at the knees.

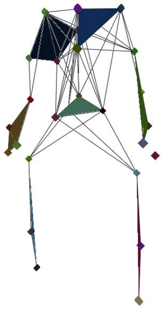


Fig. 4. Automatically built model using the proposed algorithm. The model was built using the same movement sequence as the manually built model in Figure 1. Spheres represent markers, lines represent inferred rigid links, and solid shapes represent inferred rigid bodies.

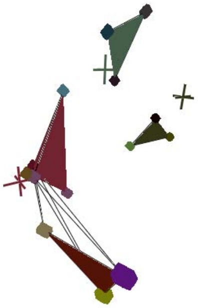


Fig. 5. Model built during an arm reaching movement. The solid shapes represent the lower arm, upper arm, right shoulder, and lower back. The 3D crosses represent the automatically estimated positions of joints

The automatically built model from an arm reaching movement is shown in Figure 5. Here we also show the joint locations that were estimated by the system (the elbow, the shoulder, and a joint that was found between the shoulder and the lower back and estimated to be somewhere in the torso). Although the joints are not yet modeled in the MRF graph, we implemented a method that infers their position.

Second, we use synthetic data to present impact of the automatically built model on the accuracy of marker tracking. We present three synthetic scenarios, each consisting of 1000 frames. In the *single marker scenario*, a single marker is following a trajectory

Table 1. Root Mean Square Error (in mm) of the observation data, as well as the inferred marker position with 0, 1, or 2 frames of latency. The algorithm shows a moderate improvement of tracking accuracy over raw observations, and maintains an acceptable error even when the marker becomes occluded.

scenario	*observation*	latency 0	latency 1	latency 2
single marker	17.57	16.24	13.31	13.18
four markers - unoccluded	17.03	16.52	14.94	14.96
three markers - unoccluded	16.95	16.43	14.80	14.81
four markers - occluded	N/A	19.70	18.61	18.54
three markers - occluded	N/A	50.51	49.50	48.31

given by $[t + \sin(t/100), 0, 0]$, i.e. it is traveling in a steady direction while accelerating and decelerating as given by the sin function. In the *four marker scenario*, four markers are traveling at a constant distance from each other, individually following the same trajectory as the marker in the single point scenario. A marker roughly between the other three becomes occluded between frames 600 and 900. In the *three point scenario*, three co-linear markers are traveling at a constant distance from each other (following the same individual trajectory described above). The middle marker becomes occluded between frames 600 and 900. In all scenarios, the ground truth data was masked by Gaussian noise with a standard deviation of roughly 17mm (very noisy for marker data).

For a marker of interest (in the multiple marker scenarios, it is the marker that becomes occluded), we compare the ground-truth location with the location provided by the algorithm with 0, 1, and 2 frames of latency. Note that each frame of latency involves another pass of the adapted EM algorithm on the position of the marker. The results are given in Table 1. We also give a graph of the error for the entire three marker trial in Figure 6. The results show the positive impact of EM algorithm passes on tracking results, and indicate very satisfactory tracking results even in the presence of highly noisy and occluded data.

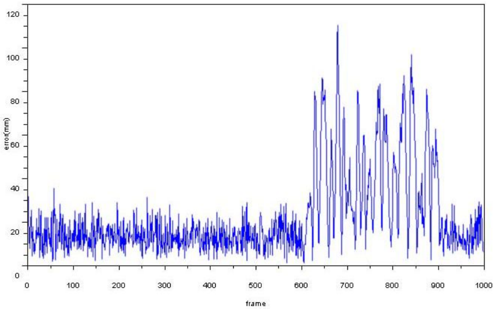


Fig. 6. Tracking error in the three marker scenario. Even though the algorithm has barely enough information to infer the position of the occluded marker, it manages to track it successfully.

5 Summary and Future Work

We have presented a novel Markov Random Field framework which unifies the tracking, labeling, and pose estimation steps of motion capture in a optical marker-based system, and demonstrated its ability to reconstruct and track the kinematic structure of a moving body with no a priori model. Our experimental results show that this approach can lead to an accurate model of a human body, joint position estimation, as well as improved marker tracking. This could eliminate requirements for manual model entry, which can potentially make the process of using motion capture systems easier and less tedious.

In our future work, we are interested in expanding the MRF framework to model joint movement explicitly, further enhancing the tracking of markers and body parts. We would also like to incorporate human body part identification, so that after the algorithm partitions the body parts it can correctly label them (lower arm, shoulder, etc.).

However, perhaps the most interesting application of this method lies in its extension to markerless tracking. As the MRF-based algorithm is not inherently tied to the 3D observations provided by a marker-based system, it can be augmented to instead work with regular cameras (e.g., with two dimensional point feature locations, or contour-based observations). Eventually, we hope that this novel method can be used to provide accurate, seamless, and non-intrusive motion capture for human-computer interaction.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No.0403428. and Grant No. 0504647. We would also like to thank the anonymous reviewers of both the HCI workshop and CVPR for their valuable suggestions.

References

1. Moeslund, T.B., Granum, E.: A survey of computer vision-based human motion capture. Comput. Vis. Image Underst. 81(3), 231–268 (2001)
2. Yan, J., Pollefeys, M.: Automatic kinematic chain building from feature trajectories of articulated objects. In: IEEE CVPR, IEEE Computer Society Press, Los Alamitos (2006)
3. Silaghi, M.C., Plankers, R., Boulic, R., Fua, P., Thalmann, D.: Local and global skeleton fitting techniques for optical motion capture. IFIP CapTech (1998)
4. Ringer, M., Lasenby, J.: A procedure for automatically estimating model parameters in optical motion capture. BMVC (2002)
5. Zakotnik, J., Matheson, T., Durr, V.: A posture optimization algorithm for model-based motion capture of movement sequences. Journal of Neuroscience Methods (2004)
6. Karaulova, I.A., Hall, P.M., Marshall, A.D.: A hierarchical model of dynamics for tracking people with a single video camera. In: BMVC, Bristol, England (2000)
7. Veenman, C.J., Reinders, M.J.T., Backer, E.: Resolving motion correspondence for densely moving points. IEEE Transactions on Pattern Analysis and Machine Intelligence (2001)
8. Shafique, K., Shah, M.: A non-iterative greedy algorithm for multi-frame point correspondence. In: Int. Conf. Computer Vision, Nice, France, pp. 110–115 (2003)
9. Hornung, A., Sar-Dessai, S., Kobbelt, L.: Self-calibrating optical motion tracking for articulated bodies. In: IEEE Virtual Reality Conference (2005)
10. Rajko, S., Qian, G.: Autonomous real-time model building for optical motion capture. In: IEEE ICIP, pp. 1284–1287. IEEE Computer Society Press, Los Alamitos (2005)

Interactive Feedback for Video Tracking Using a Hybrid Maximum Likelihood Similarity Measure

Ard Oerlemans and Bart Thomee

LIACS Media Lab, Leiden University, The Netherlands
{aoerlema,bthomee}@liacs.nl

Abstract. In this article, we present an object tracking system which allows interactive user feedback to improve the accuracy of the tracking process in real-time video. In addition, we describe the hybrid maximum likelihood similarity, which integrates traditional metrics with the maximum likelihood estimated metric. The hybrid similarity measure is used to improve the dynamic relevance feedback process between the human user and the objects detected by our system.

Keywords: Visual similarity, video analysis, motion detection, object tracking, relevance feedback, hybrid maximum likelihood similarity.

1 Introduction

Human Computer Interaction (HCI) will require the computer to have similar sensory capabilities as humans including face [6,11,12,13] and body [1,2,7,12,14-17] understanding. This article presents an interactive video tracking system which includes real-time user feedback in both motion detection and object tracking [14]. The feedback from the user is applied in real-time, so the change in the tracking results is immediately visible.

Tracking and identifying moving objects in images from a stationary camera, such as people walking by or cars driving through a scene, has gained much attention in the recent years. It can be a very useful technique for human-computer interaction, the next-generation games, and for surveillance applications [14].

We developed an object tracking system that can analyze live input streams from any video input device and which is able to output the locations, unique identifiers and pictorial information of the moving objects in the scene. All components are plug-ins, so in theory any method for object segmentation, tracking and user feedback can be inserted.

Object detection and identification however is a topic that has its unique set of problems that still are not fully addressed. Multiple object tracking is complicated by the fact that objects can touch or interact with each other, can occlude another, even leave the scene and come back again. And the usual problems with single object tracking, like illumination changes, partial occlusion or object deformation, still apply as well.

To get improved object tracking results, we investigate methods to include user feedback for detecting moving regions and to ignore or focus on specific tracked

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 79–87, 2007.

objects. Our object tracking framework includes color-based relevance feedback [3] functionality at both the segmentation and tracking level.

However, we have seen from earlier experiments [4], that matching the user's feedback to new input from the segmentation and tracking algorithms using standard visual similarity metrics performs poorly in some situations. Similarity metrics that adjust to the true visual similarity are needed. Especially for the constantly slightly changing object appearances, differences in color-feature values do not always have the same visual difference, so we are investigating new similarity metrics that are applied to the object tracking framework, but are applicable in general visual similarity matching as well.

2 Related Work

There has been significant research on motion tracking - an extensive review has been written by Yilmaz et. al [14], that gives a clear overview of object tracking and the challenges and limitations that current object tracking systems face. Notable scientific meetings on object tracking and related research include the Performance and Evaluation of Tracking and Surveillance (PETS) and Video Surveillance & Sensor Networks (VSSN).

Relevance feedback [3] is an interactive method that has been successfully introduced into text and image queries. It is an interactive query process where a computers internal representation of the object that a user is interested in is continually updated by providing the system with information about the relevancy of the query outcome.

3 Maximum Likelihood Based Visual Similarity

We begin by reviewing the theory of maximum likelihood theory. Given two images *X* and *Y* with feature vectors *x* and *y*, the probability of these two being similar, is the product of the probabilities of the similarity of each element of the vector.

$$Sim(x,y) = \prod_{i=1}^{n} P_{sim}(x_i, y_i) \tag{1}$$

These individual probabilities $P_{sim}(x_i, y_i)$ are directly linked to the distribution of the feature value co-occurrences and are often modelled by a chosen probability density function.

The origin of the widely used L_2 distance metric is the assumption that the differences between feature vector elements are normally distributed [4], with the same parameters for each element. This results in the following similarity metric:

$$Sim(x,y) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} e^{\left(-\frac{(x_i - y_i)^2}{2\sigma^2}\right)} \tag{2}$$

If one wants to find the most similar image to an image *X*, we loop through all images *Y* to find the image with the highest similarity value resulting from (2).

However, since σ and μ are both constants in this formula, we can simplify the maximization problem using:

$$Sim(x,y) = \prod_{i=1}^{n} e^{-(x_i - y_i)^2} \tag{3}$$

Applying *ln(ab)=ln(a)+ln(b)* to this function yields:

$$Sim(x,y) = \sum_{i}^{n} \ln\left(e^{-(x_i - y_i)^2}\right) \tag{4}$$

Which in turn can be simplified to

$$Sim(x,y) = \sum_{i}^{n} -(x_i - y_i)^2 \tag{5}$$

Finally, if we convert the maximization problem to a minimization problem, we can use:

$$Sim(x,y) = \sum_{i}^{n} (x_i - y_i)^2 \tag{6}$$

This is exactly the sum of squared distances. So now we can conclude that if the differences of all feature values have the same normal distribution, using L_2 as a distance metric is justified.

For example in motion detection in video, a representative distribution and the best fit Gaussian is shown in Figure 1.

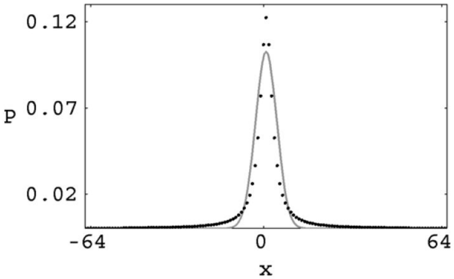


Fig. 1. The best fit Gaussian to the true distribution in video tracking

3.1 Hybrid Maximum Likelihood Similarity (HMLS)

Previous research [4] has shown that analysing the distribution $P(x_i- y_i)$ of feature differences for adjusting the similarity metric results in better retrieval results. The resulting similarity metric was called the maximum likelihood metric. In this research we are directing our focus at the distribution of $P(x_i,y_i)$.

Recall that a similarity measure is a histogram of feature value co-occurrences. If given enough training samples, the normalized histogram will give a representation of the true joint probability of the occurrence of each feature value pair for a certain class of images. The similarity of two images, given the class C of one of them, can then be calculated by

$$Sim(x, y) = \prod_{i=1}^{n} H_C(x_i, y_i) \tag{7}$$

Where H_C is the probability of the two feature values occurring for similar images, determined by the histogram for class C. To convert this into a minimization problem and to get more numerical stability in the calculations (by converting the product to a sum), we get:

$$Sim(x, y) = -\sum_{i}^{n} \log(H_C(x_i, y_i)) \tag{8}$$

3.2 The Maximum Likelihood Training Problem

In general it is difficult to find sufficient training examples to arrive at a statistically representative model of the underlying probability density function. This fundamental training problem motivates the Hybrid Maximum Likelihood Similarity Measure described next.

3.3 Hybrid Maximum Likelihood Similarity

In practice, the L_2 distance measure is typically a rough but not perfect fit to the underlying similarity measure. Therefore, we propose the hybrid maximum likelihood similarity (HMLS) measure which interpolates between the L_2 distance and the maximum likelihood distance to both obtain a better similarity measure and address the training problem from Section 3.2. At both the pixel and object-level feedback, the general algorithm for using the hybrid maximum likelihood similarity measure for a color feature is:

```
For each feature vector element x:
 Initialize a histogram Hx to Hx[i][j] = (i-j)*(i-j)
 Normalize Hx so that the sum of all H[i][j] is 1

When calculating the similarity between to elements at
position x with values i and j:
- Use Hx[i][j]

After feedback from the user:
- Create a new histogram Htemp[i][j] for each feature
  vector element
- Fill Htemp with the feature values from the examples
  from the user
- Normalize Htemp
- Set Hx = w * Hx + (1 – w) * Htemp
```

In this algorithm, *i* and *j* range over the possible values of the feature vector element, in our case [0...255]. The last step in the algorithm generates a new version of the histogram that will converge to the true similarity distribution if enough training samples are given.

4 Relevance Feedback in Object Tracking

Figure 2 gives an overview of our object tracking system and the location of the relevance feedback module in it.

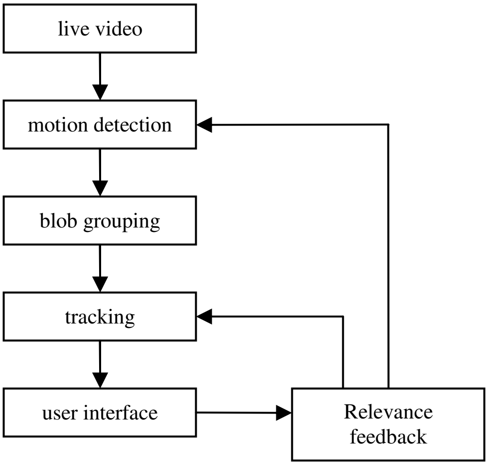


Fig. 2. The components of the object tracking system with relevance feedback

For our first experiments, we decided to use color-based relevance feedback, so we have used a color-based motion detection method developed by Horprasert, Harwood and Davis [2]. The main idea of their method is to decompose the difference between the current color information and the modeled background for every pixel into a chromaticity (color) component and a brightness component.

One assumption that the authors of this motion detection method made, was that the lighting would stay roughly constant. In real world applications however, the light can change gradually. Thus, we implemented an adaptive version of their model to compensate for dynamic real world lighting. Small parts of the background model are continuously updated with new data, to compensate for these gradual changes in lighting. Another effect of this adaptation is that deposited objects can be added to the background model if they do not move for a given period of time.

We use bounding boxes as a method for object tracking. Objects are considered either simple or compound. Simple objects are objects that can be tracked by the straightforward version of the object tracker, in which every blob corresponds to no more than one object. In the case of object interactions, or overlapping objects, there is ambiguity as to which blob belongs to which object. We define compound objects as virtual objects which consist of two or more objects which are currently interacting in some fashion. The object tracker will track these objects just as simple objects, but will also track the actual simple objects which are involved.

4.2 Pixel-Level Feedback

Our object tracking system continuously compares pixel values to the modeled background to decide whether a pixel should be considered as part of a moving object. The relevance feedback component can change this decision by learning from user input. An example is given in Figure 3. In this case, the user indicates that pixels that look like the selected part of the image (the brick wall) should never be considered to be an object, even if the background model indicates that it should, which could happen in case of fast lighting changes.

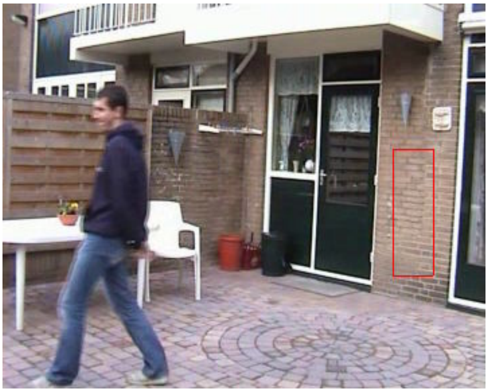


Fig. 3. User feedback for the object segmentation algorithm: selecting a negative example

The user can supply feedback while the tracking system is running. The selected positive and negative examples are immediately included in the decision process, so the effect on the object tracking results is instantly visible.

The HMLS is trained using all pairs of pixels in the area that the user has selected.

4.3 Object-Level Feedback

Below is an example of using feedback for the tracking algorithm. Figure 4 shows a frame from a sequence in which a person is leaving an object behind. In Figure 5, the user selects a positive example for the object tracking algorithm. In this case, objects with similar color will always remain marked as foreground and they will not be added to the background model, which is the normal behaviour for adaptive object tracking algorithms. Figure 6 shows the object being classified as a foreground object based on the user feedback.

Negative examples for the object tracking algorithm are useful for marking objects that are not interesting to the user. The tracking algorithm will ignore objects that are similar to the examples supplied by the user.

The HMLS is trained using information on the tracked object from each frame in which it is still tracked.

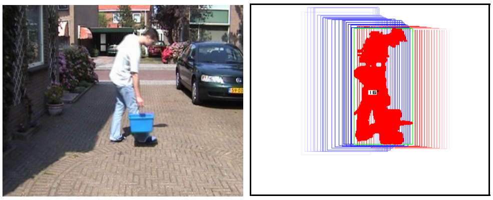


Fig. 4. One frame from a sequence where someone leaves an object behind, together with the object tracking results

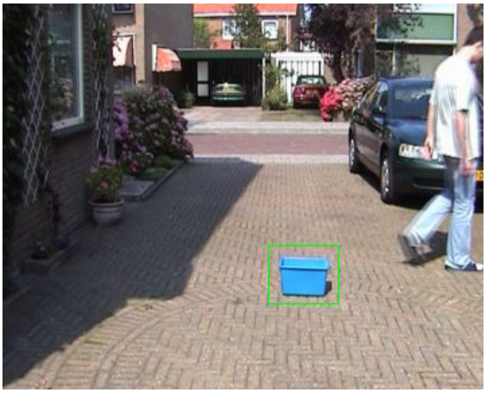


Fig. 5. The user selects a positive example for the object tracking algorithm

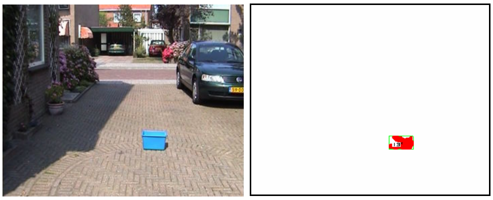


Fig. 6. The object tracking using the positive example. The object will not be added to the background model and stays visible as a tracked object.

5 Conclusions and Future Work

Based on user surveys, our interactive video tracking system has shown that including relevance feedback in the motion detection and object tracking process is intuitive and promising.

The strong point of the HMLS is that it gives the benefits of maximum likelihood similarity estimation while also addressing the limited training set problem.

In future work, we are interested in treating the temporal space as a wavelet based texture [8], learning optimal features [5], and performing more extensive quantitative evaluation including comparing different similarity measures.

Acknowledgments

This work was made possible by Leiden University, University of Amsterdam, University of Illinois at Urbana-Champaign, the Dutch National Science Foundation (NWO), and the BSIK/BRICKS research funding programs. We would also like to thank Dr. Michael Lew and Dr. Erwin Bakker for advice and discussions.

References

[1] Lazarevic-McManus, N., Renno, J., Jones, G.A.: Performance evaluation in visual surveillance using the F-measure. In: Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks, pp. 45–52 (2006)

[2] Horprasert, T., Harwood, D., Davis, L.S.: A statistical approach for real-time robust background subtraction and shadow Detection. In: IEEE Frame-Rate Applications Workshop, Kerkyra, Greece, IEEE, Los Alamitos (1999)

[3] Rui, Y., Huang, T.S.: Relevance feedback techniques in image retrieval. In: Lew, M.S. (ed.) Chapter in Principles of Visual Information Retrieval, pp. 219–258. Springer, Heidelberg (2001)

[4] Sebe, N., Lew, M.S., Huijsmans, D.P.: Toward Improved Ranking Metrics. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(10), 1132–1141 (2000)

[5] Lew, M.S., Huang, T.S., Wong, K.: Learning and Feature Selection in Stereo Matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 16(9), 869–881 (1994)

[6] Sebe, N., Lew, M.S., Cohen, I., Sun, Y., Gevers, T., Huang, T.S.: Authentic Facial Expression Analysis. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition(FG), Seoul, Korea, pp. 517–522 (May 2004)

[7] Lew, M.S., Sebe, N., Djeraba, C., Jain, R.: Content-based Multimedia Information Retrieval: State-of-the-art and Challenges. ACM Transactions on Multimedia Computing, Communication, and Applications 2(1), 1–19 (2006)

[8] Sebe, N., Lew, M.S.: Wavelet Based Texture Classification. In: Proceedings of the 15th International Conference on Pattern Recognition(ICPR), Barcelona, Spain, vol. III, pp. 959–962 (2000)

[9] Lew, M.S.: Information theoretic view-based and modular face detection. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition(FG), Killington, VT, pp. 198–203 (1996)

[10] Lew, M.S., Huijsmans, N.: Information Theory and Face Detection. In: Proceedings of the International Conference on Pattern Recognition(ICPR), Vienna, Austria, August 25-30, pp. 601–605 (1996)

[11] Cohen, I., Sebe, N., Garg, A., Lew, M.S., Huang, T.S.: Facial expression recognition from video sequences. In: Proceedings of the IEEE International Conference Multimedia and Expo (ICME), Lausanne, Switzerland, vol. I, pp. 641–644 (2002)

[12] Sebe, N., Lew, M.S., Zhou, X., Huang, T.S., Bakker, E.: The State of the Art in Image and Video Retrieval. In: Bakker, E.M., Lew, M.S., Huang, T.S., Sebe, N., Zhou, X.S. (eds.) CIVR 2003. LNCS, vol. 2728, pp. 1–8. Springer, Heidelberg (2003)

[13] Lew, M.S.: Next Generation Web Searches for Visual Content. IEEE Computer, 46–53 (2000)

[14] Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys 38(4), article no. 13 (2006)

[15] Cucchiara, R.: Multimedia surveillance systems. In: Proceedings of the third ACM international workshop on Video surveillance & sensor networks, pp. 3–10. ACM, New York (2005)

[16] Siddiqui, M., Medioni, G.: Robust real-time upper body limb detection and tracking. In: Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks, pp. 53–60 (2006)

[17] Nam, W., Han, J.: Motion-based background modeling for foreground segmentation. In: Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks, pp. 35–44 (2006)

Large Lexicon Detection of Sign Language

Helen Cooper and Richard Bowden

CVSSP, SEPS, University of Surrey, Guildford, UK
{H.M.Cooper,R.Bowden}@Surrey.ac.uk

Abstract. This paper presents an approach to large lexicon sign recognition that does not require tracking. This overcomes the issues of how to accurately track the hands through self occlusion in unconstrained video, instead opting to take a detection strategy, where patterns of motion are identified. It is demonstrated that detection can be achieved with only minor loss of accuracy compared to a perfectly tracked sequence using coloured gloves. The approach uses two levels of classification. In the first, a set of viseme classifiers detects the presence of sub-Sign units of activity. The second level then assembles visemes into word level Sign using Markov chains. The system is able to cope with a large lexicon and is more expandable than traditional word level approaches. Using as few as 5 training examples the proposed system has classification rates as high as 74.3% on a randomly selected 164 sign vocabulary performing at a comparable level to other tracking based systems.

1 Introduction

The objective of this research is to produce a non-tracking/detection based system for recognising sign language. Sign Language, being as complex as any spoken language, has many thousands of signs each differing from the next by minor changes in hand motion, shape or position. Its grammar includes the modification of signs to indicate an adverb modifying a verb and the concept of placement where objects or people are given a spatial position and then referred to later. This, coupled with the intra-signer differences make true Sign Language Recognition (SLR) an intricate challenge.

Most state of the art approaches track the hands and then classify the path that they take. This causes difficulties as the hands move quickly in sign (introducing blur and interlacing effects), have high degrees of freedom (and therefore vary in appearance) and often occlude each other. In addition, tracking often employs skin tone which means that the face and hands can be easily confused and the clothing worn by the signer must be of a contrasting colour and have sleeves which cover the arms. All of these issues are limiting factors to the success of tracking approaches. Furthermore, by combining the output of tracking with further sign classification, there are two systems which can fail, reducing overall performance. To date, relatively little work has been done on using detection for gestures or actions [1][2][3][4] and it has been limited to extremely small lexicons of around 5-10 classes. To the authors knowledge, no work to-date has

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 88–97, 2007.

addressed the scalability needed for a detection approach to tackle large lexicon recognition in sign. To allow direct comparison of our work with a tracking based approach, the dataset of Kadir et al[5] is used. The proposed detection approach can tackle large lexicon sign recognition with only a small drop in performance when compared to perfectly tracked data.

2 Background

Many of the solutions to SLR that have achieved large lexicon recognition use data gloves to acquire an accurate 3D position and trajectory of the hands [6] which, while facilitating a large vocabulary are cumbersome to the user. The majority of vision approaches are tracking based solutions with relatively small lexicons. Staner and Pentland [7] used colour to segment the hands for ease of tracking and reported classification results on a 40 sign lexicon. More recently, scalability has been addressed by turning to sign linguistics to aid classification. Vogler and Metaxas' [8] initial work operated on a lexicon of 53 signs but later reported a scalable solution using parallel HMMs on both hand shape and motion to recognise a 22 sign lexicon. Kadir et al [5] took this further by combining head, hand and torso position as well as hand shape to create a system that could be trained on five or fewer examples on a large lexicon of 164 signs. It is this work that we will make a direct comparison with as the dataset is available and allows our detection approach to be compared with the results of tracking.

Detection/non-tracking based approaches have recently begun to emerge, Zahedi et al [1] apply skin segmentation combined with 5 types of differencing to each frame in a sequence which are then down sampled to get features. Wong and Cippola [2] use PCA on motion gradient images of a sequence to obtain their features. Blank et al used space-time correlation to identify activity [3] while YanKe [4] employed boosted volumetric features in space-time to detect behaviour. All of these approaches are designed for gesture or behaviour recognition and typically only address a small number of gestures (<10). It is not obvious how these approaches could be extended to larger lexicons in a scalable way.

3 Methodology

Sign language can be broken down into visemes in much the same way that speech can be broken down into phonemes. These visemes can be separated into 5 main categories [9] based on hand; shape(s) (*dez*), placement (*tab*), movement (*sig*), orientation(s) (*ori*) and arrangement (*ha*). This work concentrates on the *tab*, *sig* and *ha* visemes shown in table 1.

Figure 1 shows an overview of the approach. Signs are recognised by a two stage process. In the second stage a high level classifier bank made up of 1st order Markov chains recognises the temporal order of visemes as they are produced. The visemes are *detected* by three different types of viseme level classifiers.

For *tab* visemes there needs to be correlation between where the motion is happening and where the person is; to this end spatial grid features centred

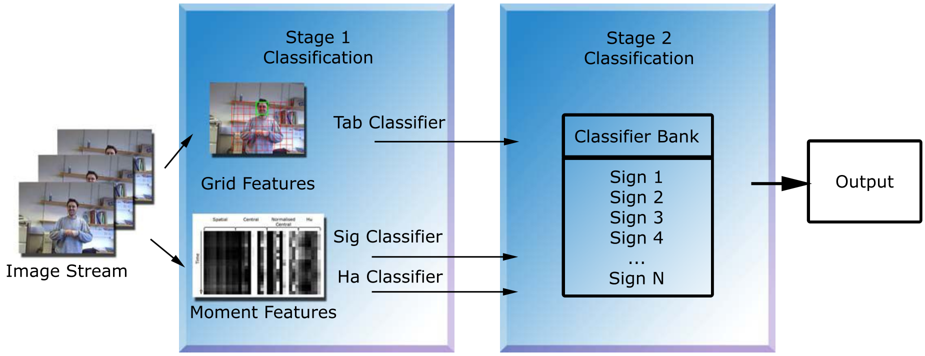


Fig. 1. Block diagram showing a high level overview of the stages of classification

around the face of the signer are used. For *sig* visemes we are interested in what type of motion is occurring, often regardless of its position, orientation or size, this is approached by extracting moment features and using local binary pattern (LBP) and additive classifiers based on their changes over time. *Ha* visemes look at where the hands are in relation to each other so these are only relevant for bi-manual signs, this is done using the same moment features as for *sig* but this time over a single frame since there is no temporal context required.

All of these viseme level classifiers are learnt using boosting which provides a way of building a strong classifier that performs well through a simple selection process. An iterative algorithm, boosting first selects the best weak classifier from a set compiled of all available features (each with an optimum response threshold). It then applies a weighting to each training example. Reducing the weighting of examples classified in the last pass and increasing the weighting of those not classified boosts the importance of examples which prove challenging to classify. This encourages the next iteration to concentrate more on the challenging examples with the heaviest weightings. More specifically in this paper AdaBoost [10] is used.

The next section discusses in detail the approaches to spatial and moment based feature extraction along with the classifiers applied to them. Section 5 then discusses how signs are recognised from the detected viseme sequences and section 6 provides comparative results. Finally conclusions are drawn.

4 Stage 1 - Viseme Detection

4.1 Skin Segmentation

In order to perform viseme detection the video is first preprocessed to find candidate hand regions. This is done by first finding the face of the user using the Viola Jones Face detector [10] included in the OpenCV library [11]. From this face region a Gaussian skin colour model is learnt using the process outlined

Table 1. The viseme level classifiers that are built

Tab	***Sig* (Both Hands)**	***Sig* (Right Hand Only)**	***Ha***
Upper Face	Apart	Circle (Type 4)	Left Higher
Nose	Together	Up	Right Higher
Ear	Together (Bend Wrist)	Up & Left	Side by Side
Eyes	Circle (Type 1)	Up & Away	Interlinking
Whole Face	Circle (Type 2)	left	Contacting
Cheek	Alt Circle (Type 3)	Left & Down	Right Near
Mouth & Lips	Up	Right	Left Near
Lower Face/Chin	Right	Right & Away	
Under Chin	Wiggle	Wiggle	
Front of Shoulders	Alt Away & Towards	Palm Down	
On Right Shoulder	Up & Down	Away & Towards	
Chest	Alt Up & Down	Away	
Right of Chest	Tap	Away & Down	
Left of Chest	Down	Spiral Away	
Upper Arm		Towards	
Lower Arm		Towards & Up	
Neutral Space		Down	
		Down & Away	
		Away & Towards(Twist Wrist)	
		Tap	
		Side to Side	

in [12]. Then the background is modelled using a normalised histogram (PDF). A threshold applied to the likelihood ratio of *face* to *background* for each pixel gives a binary, skin segmented frame. Morphological opening is used to clean up any noise and the result is shown in figure 2. Although this provides candidate hand regions it also segments the face, however, as this is consistent across both negative and positive training examples the viseme detectors will ignore its presence. Likewise, any noise in the segmented image can also be ignored as it will be inconsistent across positive training examples.

4.2 *Tab* - Spatial Features

In order that the motion can be localised in relation to the signer, a grid is applied to the image dependant upon the position and scale of the face detection. Each rectangle is a quarter of the face size and the grid is 10 rectangles wide by 8 deep, as shown in figure 3 (a). The skin segmented frame is then quantised into this grid and a rectangle is considered to be firing if over 50% of its pixels are made up of skin. For each of the *tab* visemes a classifier can then be built via boosting to show which rectangles fire for that particular viseme, examples of these classifiers are shown in figure 3 (b).

4.3 *Sig* and *Ha* - Moment Feature Vectors

There are several different types of moments which can be calculated over a segmented image, each of them displaying different properties. Four of the basic

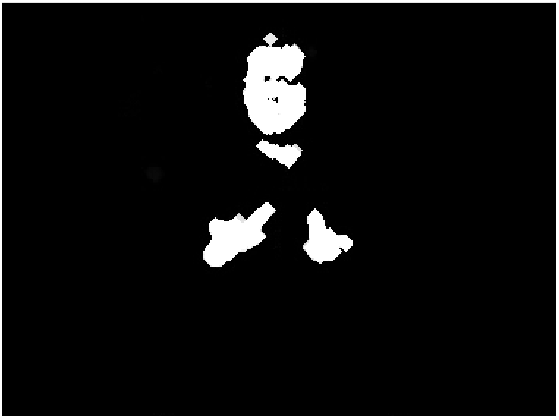


Fig. 2. Skin segmented frame showing hands and face

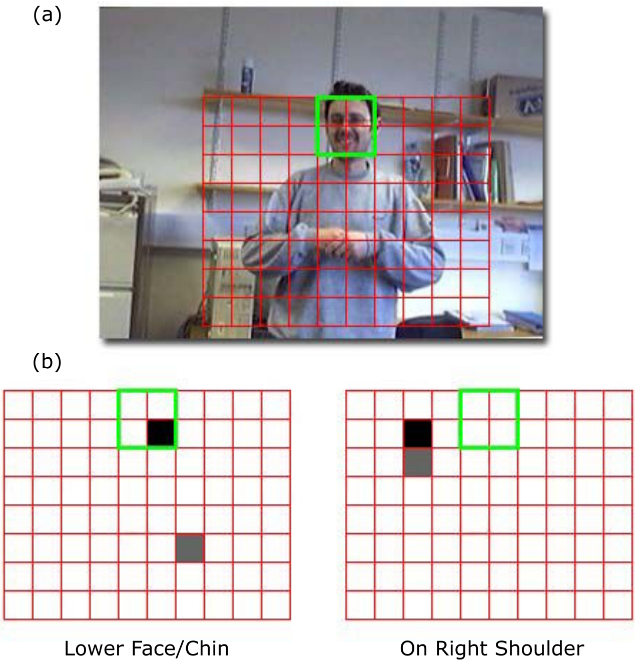


Fig. 3. (a) An example of the grid produced from the face dimensions. and (b) Grid features chosen by boosting for two of the 17 *tab* visemes. The thick central box shows the face location and the first and second chosen features are shown in black and grey respectively.

types were chosen to form a feature vector: spatial, central, normalised central and the Hu set of invariant moments. The central moments are invariant to position, the normalised central moments invariant to size and position and the Hu moments offer rotational and skew invariance. Taking up to the 3rd order from each of these types gives a vector of 37 different parameters with a wide range of different properties. Since spatial moments are not invariant to translation and scale there needs to be a common point of origin and similar scale across examples. To this end, the spatial moments are treated in a similar way to the spatial features in 4.2 by centring and scaling about the face of the signer. For training *Ha*, this vector is used to boost a set of thresholds for individual moments, but for *Sig*, temporal information needs to be included. So the video clips are described by a stack of these vectors, like a series of 2D arrays, as shown in figure 4 (a) and temporal features employed (see next section).

4.4 *Sig* - Local Binary Patterns and Additive Classifiers

Boosting chooses from two different types of classifiers which act upon the 2D feature array; local binary patterns (LBPs) and additive classifiers. LBPs work on the gradient of a feature over time, they vary in size from 2 bits to 5 bits and there are therefore 60 different classifier patterns ($2^2 + 2^3 + 2^4 + 2^5$). We run the LBPs parallel with the time axis so that they are always operating on one type of value. In essence, the LBPs encode whether a moment is increasing or decreasing with time. For an LBP to return a 1 every gradient must match its corresponding value in the patten, 1 for an increase or 0 for a decrease or no change as can be seen in figure 4 (b).

The additive classifiers sum the values across a single moment type for a given number of frames, they can be as small as a single value or as large as the maximum classifier size allowed (tests were run of classifiers up to 26 frames long). They therefore contain information about the magnitude of values across a given viseme which complements the LBPs gradient data.

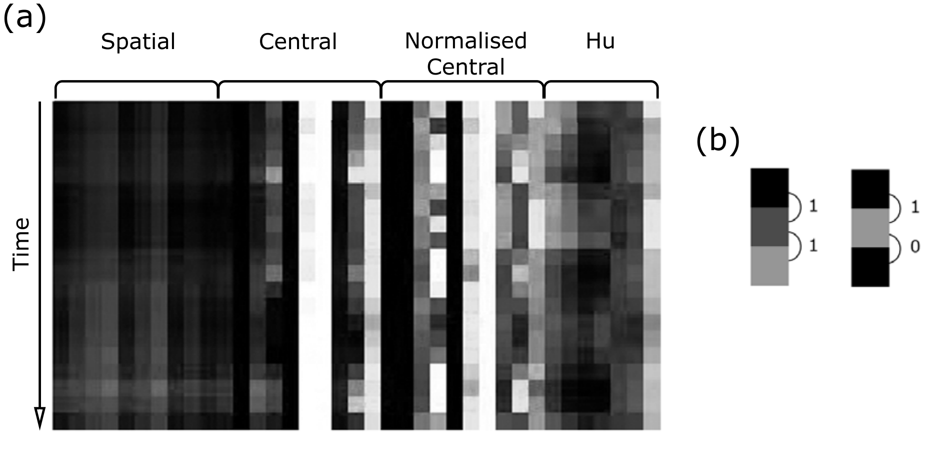


Fig. 4. (a) A pictorial description of moment vectors (normalised along each moment type for a selection of examples). (b) Local Binary Patterns, an increase in gradient is depicted by a 1 and a decrease or no change by a 0.

5 Stage II - Word Level Learning

The boosted viseme classifiers are combined to create a binary feature vector which is fed into a second stage classifier similar to that used in Kadir et al's work [5]. In order to represent the temporal transitions which are indicative of a sign, a 1st order assumption is made and a Markov chain is constructed for each word in the lexicon. An ergodic model is used and a Look Up Table (LUT) used to maintain as little of the chain as is required. Code entries not contained within the LUT are assigned a nominal probability. This is done to avoid otherwise correct chains being assigned zero probabilities. The result is a sparse state transition matrix, $P_w(s_t|s_{t-1})$, for each word w giving a classification bank of Markov chains.

During classification, the model bank is applied to incoming data in a similar fashion to HMMs. The objective is to calculate the chain which best describes the incoming data i.e. has the highest probability that it produced the observation sequence s. Symbols are found in the symbol LUT using an L1 distance on the binary vectors. The probability of a model matching the observation sequence is calculated as $P(w|s) = \upsilon \prod_{t=1}^{l} P_w(s_t|s_{t-1})$, where l is the length of the word in the test sequence and υ is the prior probability of a chain starting in any one of its states, as in [5] this is set to $\upsilon = 1$.

6 Experimental results

6.1 Data Set

The data set used is the same 164 sign set as used by Kadir et al [5] and therefore a direct comparison can be made between their tracking based system and this detection based approach. The data set consists of 1640 examples (10 of each sign). Signs were chosen randomly rather than picking specific signs which are known to be easy to separate. The viseme classifiers are built using only data from the first 4 of the 10 repetitions of each sign and the word level classifier is then trained on up to 5 examples (including the 4 previously seen) leaving 5 completely unseen examples for testing purposes. Furthermore, only visemes from the first 91 signs are used in the viseme detector learning.

6.2 Stage 1 Classification Results

Since the time taken for a viseme differs between *Sig* types, several different length classifiers were boosted starting at 6 frames long, increasing in steps of 2 and finishing at 26 frames long. Training classification results were then found for each viseme and the best length chosen to create a final set of classifiers of various lengths as shown in table 2. As can be seen there is a large disparity between optimum classifier lengths for different visemes, while short motions like the *wiggle* visemes result in short classifiers, only 6 or 10 frames, others like *together* and *spiral away* benefit from a longer classifier.

A breakdown of viseme classifiers combined with the second stage classifier is shown in table 3. Unfortunately this data is not available for comparison in the Kadir et al paper [5]. As can be seen, each of the first stage classifiers can achieve around 30% accuracy when combined individually with the second stage classifier. This is relatively poor performance as it is not possible to distinguish 164 signs on something as simple as hands moving appart, however, through their combination impressive results can be achieved as will be seen in the next section.

Table 2. Classifier lengths for a given viseme

***Sig* (Both Hands)**	**length**	***Sig* (Right Hand Only)**	**length**
Apart	10	Circle (Type 4)	24
Together	26	Up	8
Together (Bend Wrist)	20	Up & Left	20
Circle (Type 1)	18	Up & Away	18
Circle (Type 2)	6	left	26
Alt - Circle (Type 3)	6	Left & Down	20
Up	12	Right	6
Right	22	Right & Away	12
Wiggle	6	Wiggle	10
Alt - Away & Towards	14	Palm Down	6
Up & Down	26	Away & Towards	24
Alt - Up & Down	22	Away	14
Tap	16	Away & Down	22
Down	14	Spiral Away	22
		Towards	14
		Towards & Up	12
		Down	26
		Down & Away	24
		Towards & Away (Twist Wrist)	18
		Tap	8
		Side to Side	6

Table 3. Classification performance using *Ha Tab Sig* classifiers individually with Stage 2 Classification trained on 5 examples

Stage 1 Classifier	***Ha***	***Tab***	***Sig***
Mean	33.2	31.7	29.4
Minimum	31.6	30.7	28.2
Maximum	35.0	32.2	30.5
Std. Deviation	0.9	0.4	0.6

6.3 Stage 2 Classification Results

Tests were performed on the 5 unseen examples of each of the 164 signs using a random selection of training 1 to 5 examples. The results from these runs are shown in table 4 along with the results from Kadir et al [5]. As can be seen, the detection based method is only 6.6% less accurate than the tracking used in their paper for 5 training examples.

Since the grid used for *tab* classification can produce a binary feature vector of 80 values it was tried in place of the 17 *tab* classifiers (see table 5), while it offered a minor increase when training on 5 examples it was less able to generalise with fewer training examples and consistently performed worse. In addition, this increases the size of the combined viseme vector to 122 in place of 59, more than doubling it which drastically increases the possible states and transitions in the second stage classifier.

Table 4. Classification performance compared with Kadir et al [5] trained on 5 examples using *Ha, Tab, Sig* classifiers together with Stage 2 Classification

No. Training Examples	**1**	**2**	**3**	**4**	**5**	**Kadir et al [5]**
Mean	35.5	50.2	58.6	64.6	**72.6**	**79.2**
Minimum	35.1	49.5	57.6	63.2	**68.7**	**76.1**
Maximum	35.7	50.7	59.1	65.6	**74.3**	**82.4**
Std. Deviation	0.2	0.4	0.4	0.7	**1.5**	**2.1**

Table 5. Classification performance trained on 5 examples using *Ha, Sig* classifiers and using the vector output from the grid in place of the trained *tab* classifiers together with Stage 2 Classification

No. Training Examples	**1**	**2**	**3**	**4**	**5**
Mean	31.7	44.0	54.7	63.7	**74.3**
Minimum	31.0	42.1	53.3	61.8	**69.8**
Maximum	32.2	44.8	55.5	64.6	**77.2**
Std. Deviation	0.3	0.8	0.7	0.8	**2.2**

7 Conclusions

This paper has shown that near equivalence with tracking can be achieved using solely detection in sign language recognition. This has also been done over a large lexicon database with few training examples. It demonstrates the power of combining viseme level classifiers to create word level classifiers in order to reduce the complexity as the vocabulary of the system increases. Kadir et al [5] noted a 10% increase when a *dez* classifier was included so a logical extention of this work would be to include a non-tracking based classifier for hand shapes/orientations which should afford a similar boost to the stated results.

References

1. Zahedi, M., Keysers, D., Ney, H.: Appearance-based recognition of words in american sign language. In: Second Iberian Conference in Pattern Recognition and Image Analysis, vol. 1, pp. 511–519 (June 2005)
2. Wong, S.F., Cipolla, R.: Real-time interpretation of hand motions using a sparse bayesian classifier on motion gradient orientation images. In: Proceedings of the British Machine Vision Conference, Oxford, UK, vol. 1, pp. 379–388 (September 2005)
3. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. In: IEEE International Conference on Computer Vision(ICCV), Beijing, China (October 2005)
4. Ke, Y., Sukthankar, R., Hebert, M.: Efficient Visual Event Detection Using Volumetric Features. In: International Conference on Computer Vision (2005)
5. Kadir, T., Bowden, R., Ong, E.J., Zisserman, A.: Minimal training, large lexicon, unconstrained sign language recognition. In: Proceedings of the British Machine Vision Conference, vol. 2, pp. 939–948 (2004)
6. Fang, G., Gao, W., Ma, J.: Signer-independent sign language recognition based on sofm/hmm. In: RATFG-RTS 2001. Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, Washington, DC, USA, p. 90. IEEE Computer Society, Los Alamitos (2001)
7. Starner, T., Pentland, A.: Real-time american sign language recognition from video using hidden markov models. In: ISCV 1995. Proceedings of the International Symposium on Computer Vision, Washington, DC, USA, p. 265 (1995)
8. Vogler, C., Metaxas, D.N.: Handshapes and movements: Multiple-channel american sign language recognition. In: Camurri, A., Volpe, G. (eds.) GW 2003. LNCS (LNAI), vol. 2915, pp. 299–308. Springer, Heidelberg (2004)
9. British-Deaf-Association: Dictionary of British Sign Language/English. Faber and Faber (1992)
10. Viola, P., Jones, M.: Robust Real-time Object Detection. Second International Workshop on Statistical and Computational Theories Of Vision Modelling, Learning, Computing, and Sampling (2001)
11. OpenCV-User-Group: OpenCV Library (2007), `http://opencvlibrary.sourceforge.net`
12. Micilotta, A.: Bowden, R.: View-based location and tracking of body parts for visual interaction. In: British Machine Vision Conference (BMVC), Kingston, UK (September 2004)

Nonparametric Modelling and Tracking with *Active*-GNG

Anastassia Angelopoulou[1], Alexandra Psarrou[1], Gaurav Gupta[1],
and José García-Rodríguez[2]

[1] Harrow School of Computer Science, University of Westminster,
Harrow HA1 3TP, United Kingdom
{agelopa,psarroa,g.gupta1}@wmin.ac.uk
[2] Department of Computer Technology and Computation, University of Alicante,
Apdo. 99. 03080 Alicante, Spain
jgarcia@dtic.ua.es

Abstract. In this paper we address the correspondence problem, with its application to nonrigid tracking and unsupervised modelling, as a nonparametric, *active*-linking topology learning problem. Unlike existing soft competitive learning methods, *Active* Growing Neural Gas (*A*-GNG) has both global and local properties which allows part of the network to reconfigure while tracking. In addition, *A*-GNG uses a number of features (e.g. topographic product, local grey-level and map transformation) so that the topological relations are preserved and nodes correspondences are retained between tracked configurations. Experimental results in a sequence of hand gestures and artificial data have shown the superiority of our proposed method over the original GNG.

1 Introduction

Accurate nonrigid shape modelling and tracking is a challenging problem with applications in human-computer interaction, motion capture, and scene understanding. Recent approaches to non-rigid modelling and tracking typically involve the collection of training examples which require the segmentation and alignment of an observation sequence, which is an ill-conditioned task due to measurement noise and human variation in the observation. As a result, segmentation typically involves manual intervention and hand labelling of image sequences, a time consuming and labor intensive process that is only feasible for a limited set of gestures [8,5]. Moreover, most of the common tracking schemes require a good representation of the posterior distribution so that low-degree parametric models can be applied to the observation [3]. This has motivated many researchers to consider nonparametric representations, including particle filters and nonparametric belief propagation [11,10].

In this paper the nonrigid tracking and unsupervised model generation, is addressed as a nonparametric topology learning problem [7,1]. The contour of the object is described by adding an *active* step to the GNG network which allows the model to re-deform locally, and update its position. The *Active*-GNG takes

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 98–107, 2007.

into consideration not only the geometrical position of the nodes, but also the underlined local feature structure of the image, and the distance vector between the modal image and any successive images. To measure the quality of our model we use the topographic product. These features (e.g. topographic product, local grey-level and distance vector) of *Active*-GNG allow us to automatically model and track in an unsupervised manner $2D$ hand gestures in a sequence of k frames.

The remaining of the paper is organised as follows. Section 2 introduces the *Active* step of the GNG algorithm for the local searching and tracking of the nodes. Section 3 introduces the topographic product, our objective function to quantify the neighbourhood preservation. A set of experimental results are presented in Section 4, before we conclude in Section 5.

2 *Active*-GNG

When using shape or feature information or combination of the two to track nonrigid objects in video sequences, the most effective models are either 'snakes' introduced by Kass *et al.* [9] or Point Distribution Models (PDMs), Active Shape Models (ASMs) or Active Appearance Models (AAMs) introduced by Cootes and Taylor [4]. In the case of snakes, the deformation of the model to an unseen image is achieved by means of energy minimisation. The snake converges when all the forces achieve an equilibrium state. This dynamic behaviour of the model to minimise its energy function makes the snake *active*. In PDMs, the deformation of the model to an unseen image is specific since *a priori* knowledge such as expected size, shape and appearance is encoded in the model from a training set of correctly annotated images. The ASMs and AAMs have proven to be very powerful tools for interpreting new images. However, as with the snakes the deformation of the model adheres only to global shape transformations.

Since we want the network to converge either globally or locally, we introduce here a nonparametric approach to modelling the objects which makes it ideally suited for learning in dynamic environments. Our model is a modification to the GNG network introduced by Fritzke [6], called *Active* Growing Neural Gas (A-GNG) that has the characteristics of a snake, no *a priori* knowledge of the domain and global properties, but is extended in three ways:

1. The correspondence of the nodes is performed locally, so the model redeforms only where differences in the input space between successive images exist (Figure 1). Therefore, the *active* step is performed locally in contrast to the global properties applied to the image by the snake.
2. The mean vector of the map and of any successive image is calculated and the nodes update their position based on this mean difference. By doing this the map first updates its position into the successive image and then examines a region of the image around each node to determine a better displacement of the node.
3. In order to improve efficiency, we restrict the nodes to their corresponding place by adding a second dimension to the network with information about the local feature structure of the image (Figure 2).

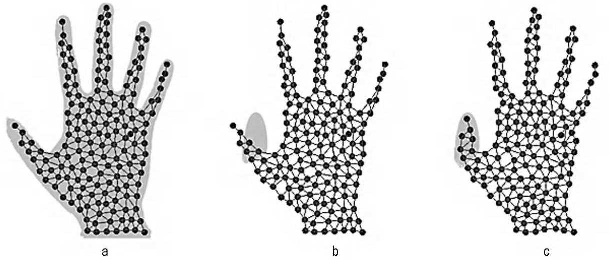


Fig. 1. Example of $2D$ local adaptation of the network. Signals are generated only to the new input distribution, Image (b), and the winner node and its direct topological neighbours update their positions, Image (c).

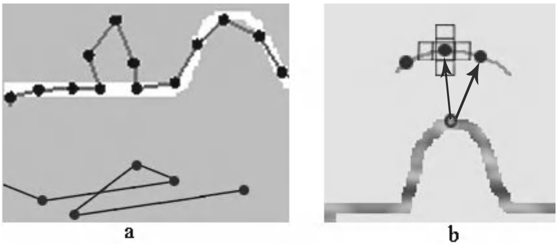


Fig. 2. The upper part of Image a shows the convergence of the algorithm to a local minimum. The top node with its direct neighbours can never be winners. The lower part of Image a shows the fold-over that will occur after a number of iterations. Not only point correspondences are lost but also topology relations are violated. To overcome this problem for each node we compute a $2k+1$ dimensional feature vector which encapsulates grey-level information. Thus, the node with the best feature vector times distance measure will be the winner node. Image b shows the feature vector $2k+1$ added to each node.

Figure 1 and 2 show the local adaptation of the network and the best matching node denoted by the distance and the underline feature structure.

As any self-organising network the A-GNG consists of:

- A set N of cluster centres known as nodes. Each node $c \in N$ has its associated reference vector $y_c \in \mathbb{R}$. The reference vectors can be regarded as positions in the input space of their corresponding nodes. Given N reference vectors $\{\boldsymbol{y}_c\}_{c=1}^{N} \subseteq \mathbb{R}$ drawn from the random vector $\boldsymbol{W}$, we want to find a mapping $G : \mathbb{R} \longrightarrow \mathbb{R}^{\wp}$ and its inverse $F : \mathbb{R}^{\wp} \longrightarrow \mathbb{R}$ such that $\forall c = 1, ..., N$,

$$f(\boldsymbol{x}) = E_{\boldsymbol{W}|g(\boldsymbol{W})}\{\boldsymbol{W}|g(\boldsymbol{W}) = \boldsymbol{x}\}, \forall x \in N \subseteq \mathbb{R}^{\wp} \tag{1}$$

$$g(\boldsymbol{W}) = \arg \min_{\mu \in \{\boldsymbol{x}_c\}_{c=1}^{N}} \|\boldsymbol{W} - f(\mu)\| \tag{2}$$

where $\{\boldsymbol{x}_c\}_{c=1}^{N} \subseteq \mathbb{R}^{\wp}$ are the reduced reference vectors drawn from the random vector $\boldsymbol{W}$, E is the distance operator and $g(\boldsymbol{W})$ is the projection operator. Equations (1) and (2) show that while the forward mapping G is approximated as a projection operator, the reverse mapping F is nonparametric and depends on the unknown latent variable $\boldsymbol{x}$. In order to compute $f(x)$ the A-GNG algorithm evaluates (1) and (2) in an iterative manner. $\wp$ denotes the dimensionality of the latent space. In this work, current experiments include topologies of a line which is the contour of the object ($\wp = 1$) and triangular grid which is the topology preserving graph ($\wp = 2$).

- A set A of edges (connections) between pair of nodes. These connections are not weighted and its purpose is to define the topological structure. The edges are determined using the competitive hebbian learning method. The updating rule of the algorithm is expressed as:

$$\Delta\boldsymbol{x}_{s_1} = \epsilon_x(w_i - \boldsymbol{x}_{s_1}), \Delta\boldsymbol{x}_i = \epsilon_n(w_i - \boldsymbol{x}_i) \ (\forall i \in N_{s_1}) \tag{3}$$

where ϵ_x and ϵ_n represent the constant learning rates for the winner node $\boldsymbol{x}_{s_1}$ and its topological neighbours $\boldsymbol{x}_i$. N_{s_1} is the set of direct topological neighbours of s_1. An *edge aging scheme* is used to remove connections that are invalid due to the activation of the node during the adaptation process. Thus, the network topology is modified by removing edges not being refreshed by a time interval α_{max} and subsequently by removing the nodes connected to these edges.

The main steps of the A-GNG algorithm are as follows:

1. Start with a modal image and run the original GNG algorithm.
2. For every node sample k neighbourhood pixels. Thus, we have $2k+1$ grey-level values which can be put in a vector $\boldsymbol{g}_i$.

$$\boldsymbol{g_i} = [g_1, \ldots.., g_{2k+1}]^T \tag{4}$$

The total shape then is given as:

$$S_m = \boldsymbol{g}_i^T * \boldsymbol{x} \tag{5}$$

where $\boldsymbol{x}$ is a $2n$ $\{x_i, y_i\}$ node vector and $\boldsymbol{g_i}$ is a $2k+1$ local feature vector.

3. Given N number of nodes calculate the mean node $\overline{\boldsymbol{x_c}}$ of the modal image, where

$$\overline{\boldsymbol{x_c}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x_i} \tag{6}$$

4. Calculate the image difference between the modal image and any other successive image. Let A and B be two sets of elements in $\mathbb{R}$ representing the modal and the successive image respectively. The Minkowski subtraction of B from A is defined as:

$$A - B = \bigcap_{b \in B} A_b \tag{7}$$

where the elements b are the pixel coordinates of the successive image.

5. Let $C = A - B$ be the new input distribution of the network.
6. Randomly generate input signals w_i to C and calculate as in step 3 the mean signal $\overline{w_i}$.
7. Calculate the distance vector of the two means and swift the nodes towards C. For each successive image we calculate its deviation from the mean, $\boldsymbol{dx_i}$ where
$$\boldsymbol{dx_i} = \boldsymbol{x_i} - \overline{\boldsymbol{x_c}} \tag{8}$$
8. Randomly generate input signals w_i to C and find the winner node x_{s_1} and its direct topological neighbours x_i.
9. Update the position of the nodes by moving them towards the current signal by the weighted factors ϵ_x, and ϵ_n same as in GNG.
10. Remove the used signal w_i from the input distribution.
11. Repeat iterations $3 - 10$ until the system converges.

The parameters used in all simulations are: $W = 3000$, $N = 150$, $\epsilon_x = 0.1$, $\epsilon_n = 0.005$, $\alpha_{max} = 125$, $k = 5$.

3 Topographic Product

In order to establish correspondence of nodes between successive frames or object instances we use a topology preservation measurement whose attributes derive from nonlinear dynamics. The topographic product P introduced by Bauer and Pawelzik [2] is our objective function which quantifies the neighbourhood preservation of the map by computing the Euclidean distance between neighbouring nodes, in both the input and the latent space. A mapping preserves neighbourhood relations if and only if nearby points in the input space remain close in the latent space. In other words, there is no violation to the topology of the network.

The neighbourhood relationship between each pair of nodes in the latent space $\wp$ and its associative reference vectors in the input space d is given by:

$$P_1(c,k) = [\prod_{l=1}^{k} \frac{d^{\wp}(c, n_l^{\wp}(c))}{d^{\wp}(c, n_l^{d}(c))}]^{1/l} \tag{9}$$

$$P_2(c,k) = [\prod_{l=1}^{k} \frac{d^{d}(\boldsymbol{x}_c, \boldsymbol{x}_{n_l^{\wp}(c)})}{d^{d}(\boldsymbol{x}_c, \boldsymbol{x}_{n_l^{d}(c)})}]^{1/l} \tag{10}$$

where c is a node, $\boldsymbol{x}_c$ is its reference vector, n_l^d is the l-th closest neighbour to c in the input space d according to a distance d^d and $n_l^{\wp}$ is the l-th nearest node to c in the latent space $\wp$ according to a distance $d^{\wp}$. Combining (10) and (11) a measure of the topological relationship between the node c and its k closest nodes is obtained:

$$P_3(c,k) = [\prod_{l=1}^{k} \frac{d^{d}(\boldsymbol{x}_c, \boldsymbol{x}_{n_l^{\wp}(c)})}{d^{d}(\boldsymbol{x}_c, \boldsymbol{x}_{n_l^{d}(c)})} \cdot \frac{d^{\wp}(c, n_l^{\wp}(c))}{d^{\wp}(c, n_l^{d}(c))}]^{1/2k} \tag{11}$$

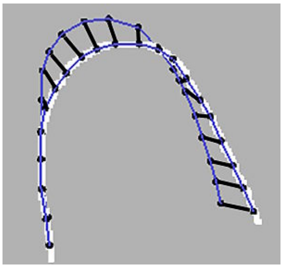


Fig. 3. Neighbourhood relations are perfectly preserved since nearby points in the input space remain close to the nearby nodes in the latent space. The mapping is indicated by the lines.

To extend this measure to all the nodes of the network and all the possible neighbourhood orders, the topographic product P is defined as:

$$P = \frac{1}{N(N-1)} \sum_{c=1}^{N} \sum_{k=1}^{N-1} \log(P_3(c,k)) \tag{12}$$

Figure 3 shows an example of a well preserved line topology mapping between two successive frames, where the network has grown sufficiently to reflect the dimensionality of the input distribution. As the input distribution moves the topological relations are updated and correct correspondences are established.

A violation of the topology occurs in Figure 4(a) since the distance relations of the data points do not correlate with that of the reference vectors in the network. Figure 4(b) shows the ideal correlation if correct correspondences have been previously established. The problem with the topographic product in cases like in Figure 4(a) is its limitation to take into account the structure of the input

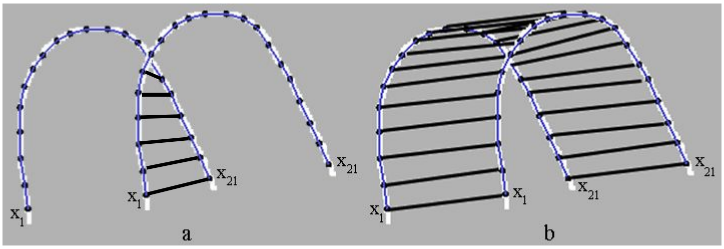


Fig. 4. A set of nodes with their reference vectors $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, up to $\boldsymbol{x}_{21}$. As the input distribution moves and the network re-adapts, the distance relations between the data points and the reference vectors are violated (Image a). In the new adaptation the nearest neighbour of $\boldsymbol{x}_1$ with its topological neighbours is not $\boldsymbol{x}_1$ but $\boldsymbol{x}_{21}$. Image b shows correct correspondences if topological information such as closest Voronoi regions and not only metric information has been used.

distribution since the map of the data points and the reference vectors is one-to-one. In order to overcome this problem where neighbourhood relations are based only on distance measures and not on topological relations, e.g. common borders of Voronoi cells, in every iteration step we update the position of the map towards the image according to the mean vector.

4 Experiments

We demonstrate the performance of our system on a sequence of hand gestures. In order to address the limitations of the existing GNG, and how these have been improved with the *A*-GNG, we use a combination of artificial and real data sets. Figure 5 (*a*) shows the initial *A*-GNG position. The contour of the first image was extracted using the original GNG and the adaptation of the network at every 10th frame is done with the *A*-GNG. Images (*b*) to (*i*) show the tracking of the nodes to a sequence of 90 frames. Our tracker is able to follow the hand gesture and update the topology of the network every 4 iterations. Figure 6 indicates another tracking example to a sequence of 45 frames. Figure 6(*a*) shows the initial position of *A*-GNG. Images (*b*) and (*c*) show the adaptation after 1 iteration to a very subtle movement. Images (*d*), (*f*) and (*h*) show the updated position of the network to a more jumping distribution and how the network re-adapts again after 1 iteration. Figure 7 is an example of local adaptation of the network between a bump model and a square and how correspondences are improved using the *A*-GNG compared to original GNG. Image (*a*) and (*b*) show the map

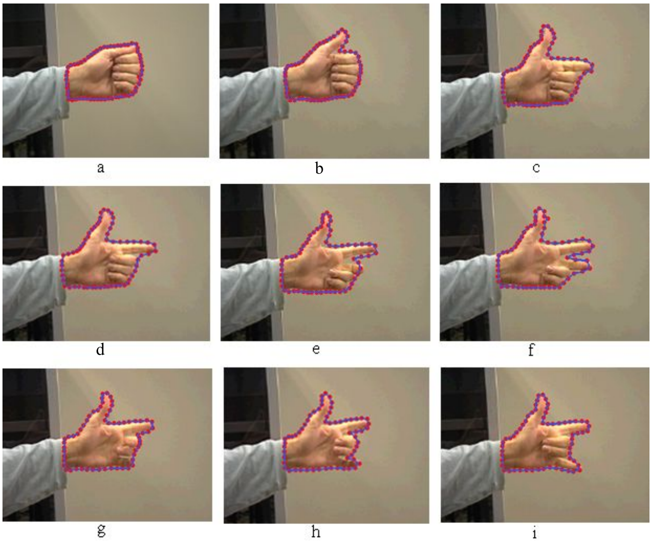


Fig. 5. Tracking a gesture. The images correspond from left to right and from top to bottom to every 10th frame of a 90 frame sequence. In each image the red points indicate the extracted nodes and their adaptation.

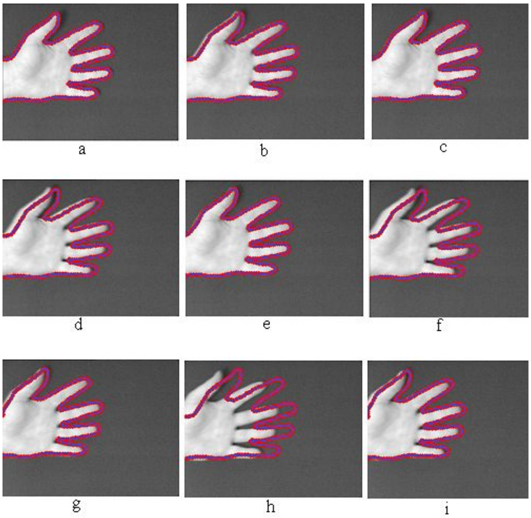


Fig. 6. Tracking a moving hand. The images correspond from left to right and from top to bottom to every 5th frame of a 45 frame sequence.

of the bump model and its superposition to the new image. Image (*c*) shows the mapping of the GNG based only on distance measures. The network fails to converge since the top nodes can never be winners. The network converges to a local minimum and after a number of iterations a fold-over to the network will occur. Image (*d*) and (*e*) show how the convergence is improved by calculating the mean vector of the map and the new image, and then updating the position of the original map according to this difference. The correspondence is improved but still it will take a number of iterations before the top nodes converge unless feature information is incorporated in every node. Figure 8 indicates how feature information can add efficiency to the convergence. Image (*a*) and (*b*) show the map and the movement of the finger. Image (*c*) shows the GNG adaptation and

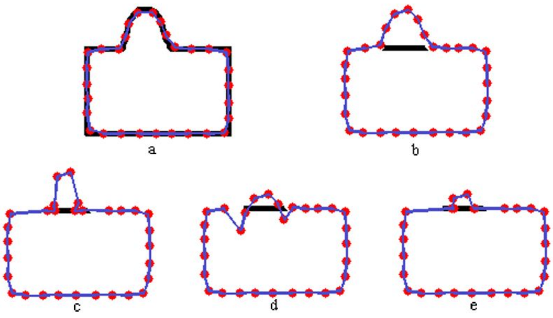


Fig. 7. Local convergence using the *A*-GNG. *c* shows the map adaptation using the GNG algorithm. *d* and *e* show the adaptation using the *A*-GNG algorithm.

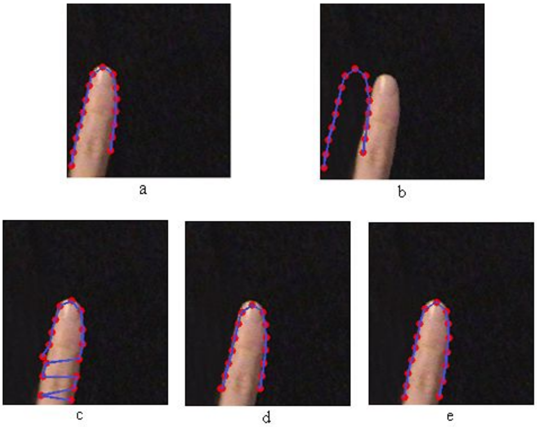


Fig. 8. Convergence with and without the *active* steps of the GNG algorithm

the violation of the map based only on distance measures while Image (d) and (e) show the correct correspondences based on the mean and the feature information added to the network. Table 1 shows the topographic product between input and latent space for both the bump and the finger model, and between any successive frames using the GNG and the A-GNG. The topographic product $P \approx 0$ indicates an approximate match while $P < 0$ and $P > 0$ correspond to a too high and a too low match. The first row indicates a match between the input space and the latent space for both the finger and the bump model. The mapping is preserved since nearby points in the input space remain close in the latent space by computing the Euclidean distance between neighbouring nodes. The second and third rows of table 1 show that A-GNG outperforms GNG and correct correspondences are established only when the map is close enough to the new input distribution.

Table 1. Measuring neighbourhood preservation by calculating the map difference between neighbouring nodes, in both the input and the latent space

Topographic Product	finger model	bump model
original map	0.049377	0.016662
A-GNG	0.043116	0.036559
GNG	-0.303199	-0.540280

5 Conclusions

We have introduced a new nonrigid tracking and unsupervised modelling approach based on a model similar to snake, but with both global and local properties of the image domain. This nonparametric learning makes no assumption about the global structure of the hand gestures thus, no background modelling nor training sets are required. Due to the number of features A-GNG uses, the topological relations are preserved and nodes correspondences are retained between tracked configurations. The proposed approach is robust to object transformations, and can prevent fold-overs of the network. The model is learned automatically by tracking the nodes and evaluating their position over a sequence of k frames. The algorithm is computationally inexpensive, it can handle multiple open/closed boundaries, and it can easily be extend to $3D$. Our method suffers from some limitations that we are trying to overcome in our future work:

- Our current system uses only absolute grey-level values, because of that if the feature match is below a particular threshold no updates to the position of the nodes is performed, so the network stays inactive.
- A clean edge map is required to serve as the distribution for the algorithm.

Currently we are extending our model by approximating a probability measure on the nodes based on previous and current position, colour information and neighbourhood relations.

References

1. Angelopoulou, A., García, J., Psarrou, A.: Learning $2d$ hand shapes using the topology preservation model gng. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 313–324. Springer, Heidelberg (2006)
2. Bauer, H.U., Pawelzik, K.R.: Quantifying the neighbourhood preservation of self-organizing feature maps. IEEE Transactions on Neural Networks 3(4), 570–579 (1992)
3. Baumberg, A., Hogg, D.: Learning flexible models from image sequences. In: Eklundh, J.-O. (ed.) ECCV 1994. LNCS, vol. 800, pp. 299–308. Springer, Heidelberg (1994)
4. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models - their training and application. Comp. Vision Image Underst. 61(1), 38–59 (1995)
5. Davies, H.R., Twining, J.C., Cootes, F.T., Waterton, C.J., Taylor, J.C.: A minimum description length approach to statistical shape modeling. IEEE Transaction on Medical Imaging 21(5), 525–537 (2002)
6. Fritzke, B.: A growing neural gas network learns topologies. In: NIPS 1994. Advances in Neural Information Processing Systems 7, pp. 625–632 (1995)
7. Furao, S., Hasegawa, O.: An incremental network for on-line unsupervised classification and topology learning. The Journal of Neural Networks 19(1), 90–106 (2005)
8. Hill, A., Taylor, C., Brett, A.: A framework for automatic landmark identification using a new method of nonrigid correspondence. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(3), 241–251 (2000)
9. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. In: Proc. of the 1^{st} Internationl Conference on Computer Vision, pp. 259–268. IEEE Computer Society Press, Los Alamitos (1987)
10. MacCormick, J., Isard, M.: Partitioned sampling, articulated objects, and interface quality hand tracking. In: Vernon, D. (ed.) ECCV 2000. LNCS, vol. 1842, pp. 3–19. Springer, Heidelberg (2000)
11. Sudderth, E.B., Mandel, M.I., Freeman, W.T., Willsky, A.S.: Distributed occlusion reasoning for tracking with nonparametric belief propagation. Advances in Neural Information Processing Systems 17, 1369–1376 (2005)

Multiple Cue Integrated Action Detection

Sang-Hack Jung, Yanlin Guo, Harpreet Sawhney, and Rakesh Kumar

Sarnoff Corporation
201 Washington Road, Princeton, NJ 08543, USA
{sjung,yguo,hsawhney,rkumar}@sarnoff.com

Abstract. We present an action recognition scheme that integrates multiple modality of cues that include shape, motion and depth to recognize human gesture in the video sequences. In the proposed approach we extend classification framework that is commonly used in 2D object recognition to 3D spatio-temporal space for recognizing actions. Specifically, a boosting-based classifier is used that learns spatio-temporal features specific to target actions where features are obtained from temporal patterns of shape contour, optical flow and depth changes occuring at local body parts. The individual features exhibit different strength and sensitivity depending on many factors that include action, underlying body parts and background. In the current method, the multiple cues of different modalities are combined optimally by fisher linear discriminant to form a strong feature that preserve strength of individual cues. In the experiment, we apply the integrated action classifier on a set of target actions and evaluate its performance by comparing with single cue-based cases and present qualitative analysis of performance gain.

1 Introduction

Automatic human action recognition plays an increasingly important role in visual surveillance and human-computer interaction systems. Robust recognition of human actions under general environmental conditions ,however, remains a challenging problem due to various factors such as human body articulation, large variations in appearance, viewing geometry, individual action execution, occlusion, dynamic background clutter, and multiple action confusers. Most approaches to action recognition are based on a single visual modality and they are most often restricted to particular environmental conditions.

A large body of works in literature on action recognition [1,2,3,4] exploit motion features such as optical flow or spatio-temporal discontinuities in the 3D image volume, primarily due to the notion that activities are naturally induced by object motions. However, motion-based features are relatively coarse level features and may lack sufficient resolutions for describing detailed action differences for reliable recognition [3]. Motion feature-based detectors can also be fragile under dynamic background motions. Image-based visual cues such as body contours or SIFT & HOG features [5] on the other hand can provide more resilience to dynamic background clutters due to insensitivity to motions. Depth-based features are available in stereo systems where relative distances of underlying body parts can be obtained. The temporal evolution of depth feature can

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 108–117, 2007.

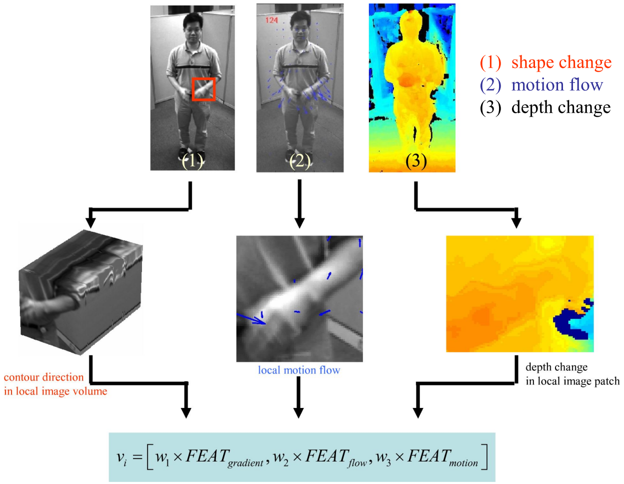

(1) shape change
(2) motion flow
(3) depth change
(1)
(2)
(3)
124
contour direction in local image volume
local motion flow
depth change in local image patch
$v_i = \left[w_1 \times FEAT_{gradient}, w_2 \times FEAT_{flow}, w_3 \times FEAT_{motion} \right]$

Fig. 1. Multiple cues based on shape, motion and depth are illustrated. Individual cues are extracted from common spatio-temporal tubes and optimally merged in the Adaboost training process.

capture protrusion of body parts, for example. The sensitivity and strength of these cues vary depending on many factors including local actions, underlying body parts and background clutters. For example, for nodding motions, motion-based features covering head regions can best capture these actions as the action does not induce significant contour changes or depth variations. However, for action of wiping head, motion-based feature cannot distinguish it from that of turning head because both generate similar motion flows, while image-based contour feature can distinguish two actions by capturing state of face being occluded by hand. To increase robustness of action recognition in dynamically changing environments, and to deal with a wide variety of environmental conditions, we propose to optimally combine the multiple complementary modality of cues in an integrated classification framework. Specifically, we extend classification framework that is commonly used in 2D object recognition to spatio-temporal space for recognizing actions. As shown in Fig. 1, we utilize multiple cue-based features that include (1) shape changes from histogram of gradient orientation (HOG), (2) local motion from optical flow and (3) depth elevation changes with respect to body plane, each of which is sampled from local image regions. A boosting-based classifier is used that learns spatio-temporal features specific to target actions where features are obtained from temporal patterns of shape contour, optical flow and depth changes occurring at local body parts. Modality combination is embedded in each iteration of feature selection process using a Fisher Linear Discriminant (FLD) that minimizes the training errors formulated in a conventional Adaboost framework. As a consequence, a strong classifier that preserves strength of individual cue is produced. We apply the integrated action classifier on a set of target actions and evaluate its performance by comparing with single cue-based cases and present qualitative analysis of performance gain. Our integrated action classifier shows prominent performance improvement over single cue based method.

2 Related Work

Computer vision researchers have attempted to integrate multiple cues in different contexts [6,7,8,9,10], and a majority of works focus on gesture recognition and body tracking. Shet at. al. [6] exploited an exemplar based technique that combines two different forms of exemplars, i.e., shape exemplars and motion exemplars in a unified probabilistic framework. Sidenbladh [7] uses edge, ridge, and motion cues for tracking and reconstruction of articulated human motion from a monocular camera in a probabilistic framework. The three cues are considered independent, and the likelihood with multiple cues is achieved by by summing the log likelihoods from the different cues. Giebel et. al [8] presents a Bayesian framework for multi-cue pedestrian tracking from a moving vehicle. The proposed spatio-temporal object representation involves a set of distinct linear subspace models or Dynamic Point Distribution Models (DPDMs), which is enhanced by texture information using intensity histograms. Direct 3D measurement is provided by a stereo system. State propagation is achieved by a particle filter which combines the three cues shape, texture and depth, in its observation density function. Paletta and Paar [9] demonstrate performance improvement using multi-cue information integrated within a probabilistic framework. Spengler and Schiele [11] discussed multi-cue tracking of faces. The approach is based on the principles of self-organization of the integration mechanism and self-adaptation of the cue models during tracking. Experiments show that the robustness of simple models is leveraged significantly by sensor and model integration. In the weighting aspect using multiple visual cues, Shan et. al. [12] uses a measurement vector consists of three independent edge-based measures and their associated robust measures computed from a pair of aligned vehicle edge maps. The weight of each match measure in the final decision is determined by an unsupervised learning process.

3 Overview

Our approach is largely inspired by the pioneering feature selection and object classification work by Viola and Jones [13]. Our work extends the rectangle features used by Viola and Jones into the spatio-temporal domain for video analysis. Action detection requires searching through 3D image volume, both spatially to localize region of interest containing target objects and temporally to identify video segments containing target actions. To address the computational complexity induced by the dimensionality of the problem, we split the problem into spatial and temopral domain and formulate action as pose changes over time, each of which is handled by two-level Adaboost-based classifiers. The first level classifier is used to obtain spatial image regions that can capture part-pose space, i.e., various pose and state of local body parts. The temporal evolution of such regions are then trained by second level classifier specific for each target action. Multiple cues are handled in the same framework by sampling common spatio-temopral tubes and each cue is integrated optimally through training procedure.

In the following sections, we first describe two-level classifiers to introduce action detection approach. Next, we describe multiple-cue integration in the proposed classification framework.

4 Two-Level Action Classifier

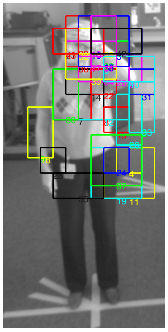


Fig. 2. Location of weak learners from spatial classifier for an action of wiping forehead

For clarity of discussion, we limit the discussion for feature to an image-based shape cue. The spatial classifier follows essentially the same framework of the typical 2D classifiers that are commonly used in object recognition such as pedestrian or car detectors [2,14]. In these approaches, a set of image regions defined by rectangular boxes are trained using image descriptors such as HOG or Haar features to obtain discriminative collection of kernels sets, called weak learners. However, in the current method, the focus of it is slightly different. The goal of this level is to identify weak learners that can capture body parts at various poses and state (such as occlusion of the face by hands) so that the obtained set of weak learners can capture overall body pose information. To obtain part-pose sensitive weak learners, we refine the weak learners from typical pedestrian detectors as follows. First, we build a general classifier for pedestrian detector from training samples where positive examples contain people at various poses and appearnaces and negative ones from non-people images. Given the resulting set of weak learners, as shown in figure 2 for example, next we refine its kernel directions by training on a new set of dataset where positive set is composed of people at various poses and negative sets of people of stand-straight poses. This procedure refines each weak learners so that its distinctiveness becomes tuned to differentation between two sets, i.e., part-pose differences. More details can be found in [15]. The two sets of weak learners from this procedures can serve both spatial target detector and pose-specific weak learners.

Temporal Second-level Action Classifier. The second level classifier aims at recognizing actions by exploiting temporal patterns in the set of refined, pose-specific weak learners over time. In this layer, spatio-temporal tubes of various temporal intervals and locations are evaluated to find action specific kernels, where spatial dimension of which are defined by the first level classifier as described before. In the training process, we construct positive action sets that consist of video segments containing target actions and negative sets are obtained by scanning video sequences that do not contain taregt actions. Temporal size of each training video segments is normalized to a constant and such temporal variations are handled at the detection stage by applying temporal normalization in the same manner. Adaboost is used to obtain discriminative set of 3D tubes that best separate two sets of training data. An example of training results is shown in figure 3 that plots selected spatial locations and its temporal extensions. It can be observed that for this particular action of wiping forehead with left hand trained kernels are strongly correlated with distinctive image regions that undergo target motions as well as covering other discriminative regions of negative actions.

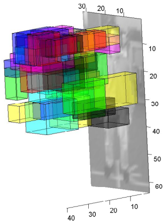


Fig. 3. Spatio-temporal tubes from 2nd level classifier that evolves temporally on weak learners in 2.

5 Multiple Cue Integration

In this section we extend our discussion to multiple cues and describe how they can be integrated in the proposed framework. Shape, motion and depth-based features are combined in this method where each cue is encoded as temporal pattern inside 3D tubes that is trained on the second level classifier. First, shape descriptor is based on temporal HOG patterns which can capture local contour changes in the spatio-tmporal tubes. Motion-based feature is extracted from optical flows sampled inside the tube in the same manner. This generates strong directional response from local parts motion. Note the difference between two features. Shape-based feature explicitily encodes contour changes and less sensitive to actual motion, thus it may encode temporal presence/absence/occlusion of certain directional contours. On the other hand, motion-based feature shows strong sensitivity to direction motion, without regard to underlying structure. The combination of shape and motion cues can explicitly encode both directional motion induced by specific contour structure. The third cue is obtained from depth, where depth elevation changes inside local region is computed with respect to body plane. This feature can capture motions of protrusion such as arms stretched out front.

Assuming that they are sampled in a tube of size $(w \times h \times t)$, they can be represented by

$$\begin{array}{l} \text{(1)shape\_tube :} TB_{shape}\,(n_{dir} \times t) \\ \text{(2)motion\_tube :} TB_{motion}\,(2 \times t) \\ \text{(3)depth\_tube :} TB_{depth}\,(1 \times t) \end{array}$$

where n_{dir} is dimension of HOG feature[1]. Figure 4 shows an example of a spatio-temporal tube sampled with shape, motion and depth features, respectively.

Multiple Cue Integration in Classification Framework. The multiple cues are integrated in training procedure of the temporal, second-level classifier. In this process, two issues need to be addressed: (1) how to combine multiple modality of features, and (2) how to optimally select such feature set for action recognition. Optimal combination of such different features is required so that combined feature can preserve strength of each individual cue. Each cue differs in strength and sensitivity based on many factors including body parts, motion patterns, background motions and so on. For example, construting a composite feature from directly concatenating three such cues will produce a non-optimal feature that is either biased by the strongest cue or perturbed by individual noise that results in overall weaker features compared with individual cues. Because of these variations, the different cues need to be combined considering individual property, i.e., relative strengths and variances across diverse data, which can be accomplished inside the classification training process. In doing so, we adopt linear combination using Fisher linear discriminant as follows. Given three different cues, $\{TB_{shape}, TB_{motion}, TB_{depth}\}$, from a common spatio-temporal tube, first, we learn optimal kernels separately for each cue. This can be obtained by following the typical Adaboost training process for obtaining weak learners based on single features.

[1] n_{dir} = 8 in the current approach.

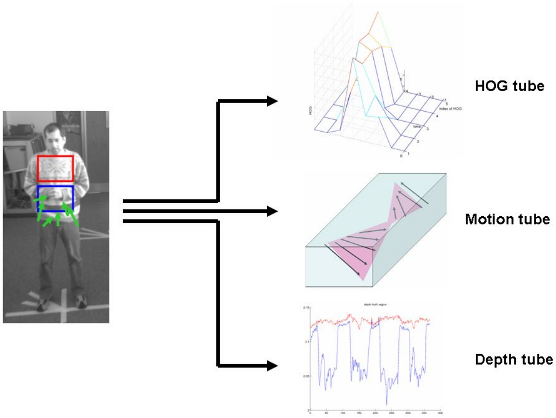
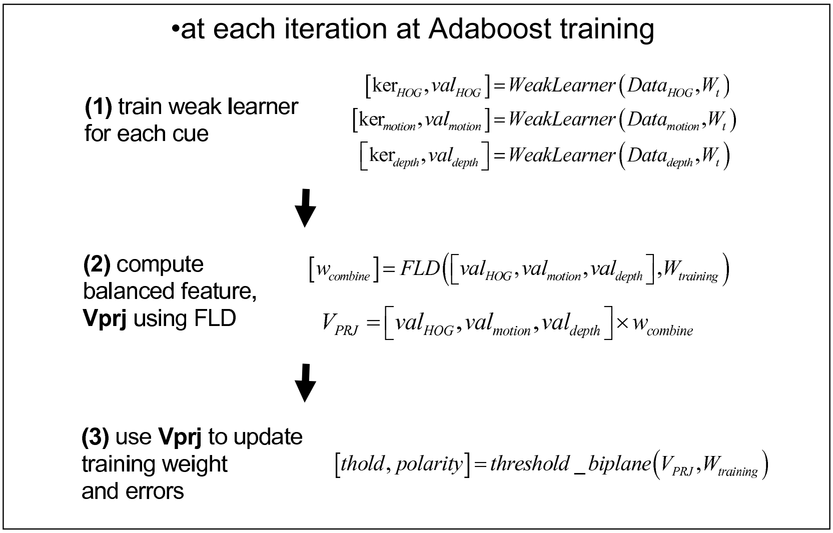


Fig. 4. Spatio-temporal tubes of shape, motion and depth cues sampled under an lifting arms motion. For depth cue, red and blue denotes depth changes w.r.t body plane where blue is shown to capture protrusion from lifted hand during motion.



•at each iteration at Adaboost training

(1) train weak learner for each cue

$$[\ker_{HOG}, val_{HOG}] = WeakLearner(Data_{HOG}, W_t)$$
$$[\ker_{motion}, val_{motion}] = WeakLearner(Data_{motion}, W_t)$$
$$[\ker_{depth}, val_{depth}] = WeakLearner(Data_{depth}, W_t)$$

(2) compute balanced feature, **Vprj** using FLD

$$[w_{combine}] = FLD([val_{HOG}, val_{motion}, val_{depth}], W_{training})$$
$$V_{PRJ} = [val_{HOG}, val_{motion}, val_{depth}] \times w_{combine}$$

(3) use **Vprj** to update training weight and errors

$$[thold, polarity] = threshold\_biplane(V_{PRJ}, W_{training})$$

Fig. 5. Diagram of multiple-cue integration using FLD

$$\begin{aligned}[kernel\,(i)] &= WeakLearner\left(\{D_{data}, P_{polarity}, W_{weight}\}\,; feature\right) \\ &= \underset{v,th}{\arg\min}\left(\sum_i W\,(i)\left(v^T D\,(i) \times P\,(i) < th \times P\,(i)\right)\right)\end{aligned}$$

where each kernel consists of kernel vector and threshold, $ker_{HOG}\,(v\,(1 \times n_{dir} \times t)\,, th)$, for example.

Next, we compute the projection of the tube examples used in the training with learned kernel. Let us assume that N $(= N_{pos} + N_{neg})$ training examples are used. From each cue, we can obtain Nx1 projection data, V_{prj}, from the learned kernel. Then we apply *FLD* to the weighted projection pattern and labels to obtain 3x1 coefficients, where Nx1 weight is computed in the Adaboost learning process. Note that the optimal combination of different cues are obtained from *FLD*[2], which guarantees that the training error of the combined feature becomes smaller than that from each individual feature. Optimal selection of such composite feature is achieved by Adaboost by integrating feature selection seamlessly in the Adaboost framework. In the Adaboost training process, for each 3D tube candidate, the above process is iterated and the best composite feature is selected that maximally separates training examples. A diagram of this process is shown in figure 5.

6 Experimental Results

This section describes experimental results of the proposed action detection method. Three target actions are used: (1) fidgeting hands (SF), (2) turning head (STL), and (3) wiping forehead (SWR). All target actions start from stand-straight pose (SS). For example, SWR describes a motion where a person from standing straight pose starts to lift a right hand and wipe forehead. STL similarly describes a motion that turns a left head and stay, to avoid eye contact, for example. Action dataset is obtained by capturing video sequences of 12 people performing target actions as well as other motions multiple times. Around in total 50 action segments are obtained for each target action. To compensate for the relative small number of positive dataset, multiple copies of positive examples are used where each copy is perturbed with random temporal noise that generates different motion durations up to 20 percent of its temporal length. For training the spatial classifier, 2000 positive image sets containing people and 4000 negative sets are used resulting in 184 weak learners. These weak learners are commonly used across different actions to generate boosting pattern maps. To measure performance, datasets from 6 subjects are used with bootstrapping and evaluation is made with all available datasets including datasets from untrained 6 subjects.

First, we perform training for target action using three cues individually by applying a single cue-based classification framework. Figure 6 shows a resulting set of weak learners for each action based on each individual cue (IMG,FLOW and DEPTH). Note that the spatial location of them differ across the different cues. For example, weak learners from motion cues are placed concentrated mostly on image regions undergoing positive motions. On the other hand, depth cue-based classifier returns weak learners positioned equally covering areas of both positive and negative motions. The performance of each classifier on target actions are measured by ROC curves as shown in figure 7. We can observe that the performance varies depending on actions. For example, for the action of fidgeting hands, motion-based one (FLOW) outperforms shape and depth-based classifiers (IMG, DEPTH). On the other hand, IMG performs strongest for the action of turning head. This can be partly explained by the fact that motion-based

[2] Note that here *FLD* can be replaced with other light-weight classifiers such as Support vector machine, for example.

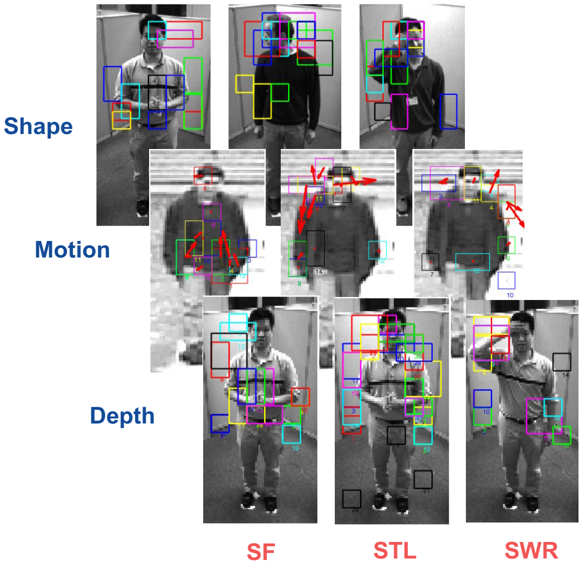
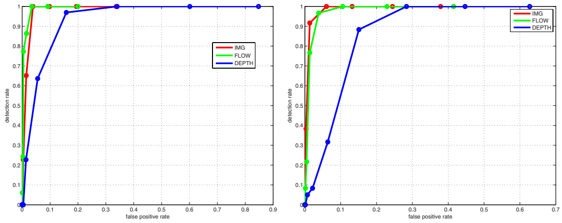


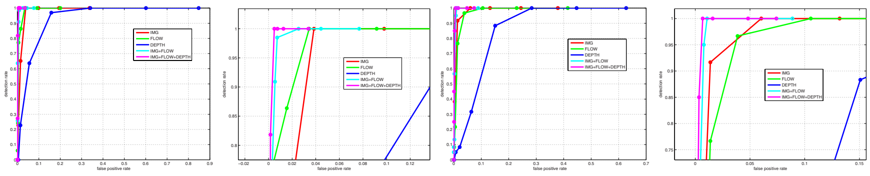
Fig. 6. Figure of weak learners obtained for three actions of fidgeting hands (SF), turning head (STL) and wiping forehead (SWR), trained using shape, motion and depth cues, respectively



(a) (b)

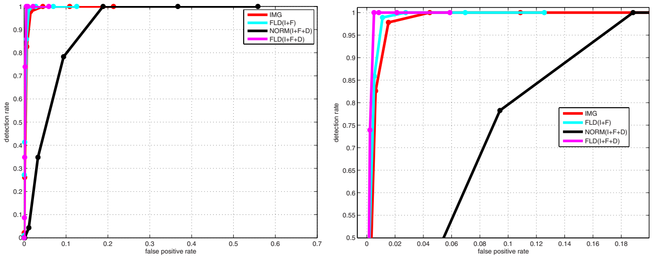
Fig. 7. ROC curves for two actions of fidgeting hands (SF) and turning head (STR) evaluated using three classifiers trained with shape, motion and depth features, respectively

feature may not provide sufficient resolution for resolving head-motion from similar motion such as wiping in the feature space, while shape feature can tell such situations apart by capturing pose/state changes such as occlusion on face v.s. turned head, for example. The performance of individual classifier is compared with that of multi-cue integrated classifier. Figure 8 shows four plots of two actions where shape+motion and shape+motion+depth is shown in cyan and magenta, respectively. In all experiments, multiple cue classifier excels in performance other single cue-based classifiers. It can be observed that multiple-cue classifier can produce performance gain of as large as 10 % as compared with single cue cases.



(a) (b) (c) (d)

Fig. 8. ROC curves for two actions of fidgeting hands (SF) and turning head (STR) evaluated using three classifiers trained with shape (red), motion (green) and depth (blue) features, individually and combination of shape+motion (cyan) and shape+motion+depth (magenta). (b) and (d) are local zoomed view of (a) and (c). The performance gain from integrated classifier can be as large as 10 % increase in detection rate compared with single cue classifier.



(a) (b)

Fig. 9. ROC curves for an action of wiping forehead (SWR) trained with shape (red), shape+motion combined by FLD (cyan), shape+motion+depth combined by normalization (black) and optimal combination of shape+motion+depth by FLD (magenta). It is shown that optimal combination (magenta) excels both individual and normalized composite feature (black). Normalized composite feature appears worse than single feature.

Next, we compare the performance of multi-cue classifier with normalized composite feature based method. In the latter method, features are combined by concatenating normalized individual features and followed by same Adaboost framework for feature selection. In figure 9, normalized composite feature performs even worse than both single shape-based one and shape+motion by FLD by a large margin. This illustrates that non-optimal feature combination from direct composition produces weak feature sets perturbed by individual feature noise.

7 Conclusion

This paper describes a method of action detection by integrating multiple cues of shape, motion and depth information and presents an analysis of it by comparing with cases of single cue-based methods. The proposed scheme of optimally combining multiple features of different modalities are shown to provide significant performance improvement, which is experimentally validated through action detection.

References

1. Shechtman, E., Irani, M.: Space-time behavior based correlation. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog., Washington, DC, USA, pp. 405–412. IEEE Computer Society Press, Los Alamitos (2005)
2. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. International Conference on Computer Vision 02, 734 (2003)
3. Ke, Y., Sukthankar, R., Hebert, M.: Efficient visual event detection using volumetric features. In: International Conference on Computer Vision, Washington, DC, USA, pp. 166–173. IEEE Computer Society Press, Los Alamitos (2005)
4. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: A local svm approach. In: International Conference on Pattern Recognition, Washington, DC, USA, pp. 32–36. IEEE Computer Society Press, Los Alamitos (2004)
5. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog., Washington, DC, USA, pp. 886–893. IEEE Computer Society Press, Los Alamitos (2005)
6. Shet, V., Prasad, V., Elgammal, A., Yacoob, Y., Davis, L.: Multi-cue exemplar-based nonparametric model for gesture recognition. In: Indian Conference on Computer Vision, Graphics and Image Processing (2004)
7. Sidenbladh, H.: Probabilistic Tracking and Reconstruction of 3D Human Motion in Monocular Video Sequences. PhD Thesis TRITA-NA-0114, Dept. of Numerical Analysis and Computer Science, KTH, Sweden (2001) ISBN 91-7283-169-3
8. Giebel, J., Gavrila, D., Schnorr, C.: A bayesian framework for multi-cue 3d object tracking. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 241–252. Springer, Heidelberg (2004)
9. Paletta, L., Paar, G.: Bayesian decision fusion for dynamic multi-cue object detection. In: Indian Conference on Computer Vision, Graphics and Image Processing (2002)
10. Birchfield, S.: Elliptical head tracking using intensity gradients and color histograms. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog. (1998)
11. Spengler, M., Schiele, B.: Towards robust multi-cue integration for visual tracking. IEEE Trans. Pattern Anal. Machine Intell. 13(9), 891–906 (1991)
12. Shan, Y., Sawhney, H., Kumar, R.: Unsupervised learning of discriminative edge measures for vehicle matching between non-overlapping cameras. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog. (2005)
13. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog. (2001)
14. Shan, Y., Han, F., Sawhney, H., Kumar, R.: Learning exemplar-based categorization for the detection of multi-view multi-pose objects. In: Proc. IEEE Conf. on Comp. Vision and Patt. Recog., Washington, DC, USA, pp. 1431–1438. IEEE Computer Society Press, Los Alamitos (2006)
15. Jung, S.H., Shan, Y., Sawhney, H., Aggarwal, M.: Action detection using approximated spatio-temporal adaboost. In: Technical Report, Sarnoff Corporation (2007)

Combined Support Vector Machines and Hidden Markov Models for Modeling Facial Action Temporal Dynamics

M.F. Valstar and M. Pantic

Department of Computing, Imperial College London
180 Queen's gate, London SW7 2AZ, England
{Michel.Valstar, M.Pantic}@imperial.ac.uk

Abstract. The analysis of facial expression temporal dynamics is of great importance for many real-world applications. Being able to automatically analyse facial muscle actions (Action Units, AUs) in terms of recognising their neutral, onset, apex and offset phases would greatly benefit application areas as diverse as medicine, gaming and security. The base system in this paper uses Support Vector Machines (SVMs) and a set of simple geometrical features derived from automatically detected and tracked facial feature point data to segment a facial action into its temporal phases. We propose here two methods to improve on this base system in terms of classification accuracy. The first technique describes the original time-independent set of features over a period of time using polynomial parametrisation. The second technique replaces the SVM with a hybrid SVM/Hidden Markov Model (HMM) classifier to model time in the classifier. Our results show that both techniques contribute to an improved classification accuracy. Modeling the temporal dynamics by the hybrid SVM-HMM classifier attained a statistically significant increase of recall and precision by 4.5% and 7.0%, respectively.

1 Introduction

A system capable of analysing facial actions would have many applications in a wide range of disciplines. For instance, in medicine, it could be used to continuously monitor a patients pain level or anxiety, in gaming a virtual avatar could be directed to mimic the user's facial expressions and in security the analysis of facial expressions could be used to assert a person's credibility.

The method proposed in this paper is based on the analysis of atomic facial actions called Action Units (AUs), which are defined by the Facial Action Coding System (FACS) [1]. FACS is the best known and the most commonly used system developed for human observers to objectively describe facial activity in terms of visually observable facial muscle actions (AUs). FACS defines 9 upper face AUs and 18 lower face AUs. There are 5 additional AUs that occur irregularly in interpersonal communication and are therefore often not mentioned in the literature [2].

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 118–127, 2007.

Previous work on automatic AU detection from videos includes automatic detection of 16 AUs from face image sequences using lip tracking, template matching and neural networks [3], detecting 20 AUs occurring alone or in combination by using temporal templates generated from input face video [4] and detection of 18 AUs using wavelets, AdaBoost and Support Vector Machines [5]. For an overview of the work done on AU and emotion detection from still images or face video the reader is referred to [6,7].

Many of the aforementioned applications of automatic facial expression analysis require the explicit analysis of the temporal dynamics of AU activation. The body of research in cognitive sciences which suggests that the temporal dynamics of human facial behaviour (e.g. the timing and duration of facial actions) are a critical factor for interpretation of the observed behaviour, is large and growing [8,9,10]. Facial expression temporal dynamics are essential for categorisation of complex psychological states such as various types of pain and mood [11]. They are also the key parameter in differentiation between posed and spontaneous facial expressions [12,13,10]. For instance, it has been shown that spontaneous smiles, in contrast to posed smiles (such as a polite smile), are slow in onset, can have multiple AU12 apices (multiple peaks in the mouth corner movement), and are accompanied by other facial actions that appear either simultaneously with AU12 or follow AU12 within 1s [8]. Recently the automatic facial expression recognition community has published work that emphasises the importance of facial expression temporal dynamics for deception detection [14].

In light of the aforementioned, it is striking that very few research groups focus on the analysis of temporal dynamics. Almost all existing facial expression recognition systems are binary classification systems that are only capable of recognising the presence of a facial action, regardless whether that action has just begun and is getting stronger, is at its peak or is returning to its neutral state. Also, while many systems are in essence capable to compute the total duration of a facial action based on this activation detection, none do this explicitly [7]. What's more, this total activation duration information alone would be insufficient for the complex tasks described above.

A facial action, in our case an AU activation, can be in any one of four possible phases: (i) the onset phase, where the muscles are contracting and the appearance of the face changes as the facial action grows stronger, (ii) the apex phase, where the facial action is at its peak, (iii) the offset phase, where the muscles are relaxing and the face returns to its neutral appearance and (iv) the neutral phase, where there are no signs of this particular facial action. Often the order of these phases is neutral-onset-apex-offset-neutral, but other combinations such as multiple-apex facial actions are possible as well.

Only recently we have proposed the first system that is capable to explicitly model the temporal dynamics of facial actions, in terms of the phases neutral, onset, apex or offset [4]. In that work we proposed to classify each frame of a video into one of the four temporal phases using features computed from tracked facial points and a multiclass Support Vector Machine (SVM). This phase detection was used to define a set of high-level parameters, such as the duration of each phase or the number of apices in the video. Using this high-level representation discrimination between posed and spontaneous brow actions [14] was possible.

However, the multiclass-SVM classification strategy adopted in our earlier work does not incorporate any model of time. The dynamics are completely modeled by mid-level parameters such as the current speed of a facial point. We believe that the expressive power of the mid-level parameters proposed in [4] can be improved upon to better describe the characteristics of AU temporal dynamics.

The method we propose is fully automatic, operating on videos of subjects recorded from a near-frontal view. It uses data from 20 tracked facial points to analyse facial expressions. We propose two ways to improve the system presented earlier [4]. First we define a set of mid-level parameters that better encodes the dynamics of facial expressions. Instead of defining the features for each frame, we describe the evolution of the feature values over a period in time using polynomial parametrisation. This way, we capture how a feature related to a specific facial action behaves dynamically instead of observing its value at a single moment.

The second improvement we propose is to explicitly incorporate a sense of time in the classification procedure. We do so by combining a Hidden Markov Model (HMM) with a SVM. While the evolution of a facial action in time can be efficiently represented using HMMs, the distinction between the temporal phases at a single point in time is usually made using Gaussian mixture models, which do not offer a high discrimination. SVMs on the other hand are large-margin classifiers known for their excellent discrimination performance in binary decision problems but they do not incorporate a model of time. We will show that the combination of the high-margin SVMs with the excellent temporal modeling properties of HMMs increases the recognition performance significantly.

2 Automatic Feature Extraction

The features used in our study are extracted by a fully automatic method consisting of, consecutively, face detection, facial point detection, facial point tracking and the calculation of geometry based features. These features are used to train and test the classifier combination described in section 3. We will now describe each subsystem in some detail.

To detect the face in a scene we make use of a real-time face detection scheme proposed in [15], which represents an adapted version of the original Viola-Jones face detector [16]. The Viola-Jones face detector consists of a cascade of classifiers trained by AdaBoost. Each classifier employs integral image filters, which remind of Haar Basis functions and can be computed very fast at any location and scale. This is essential to the speed of the detector. For each stage in the cascade, a subset of features is chosen using a feature selection procedure based on AdaBoost.

The adapted version of the Viola-Jones face detector that we employ uses GentleBoost instead of AdaBoost. GentleBoost has been shown to be more accurate and converges faster than AdaBoost [17]. At each feature selection step

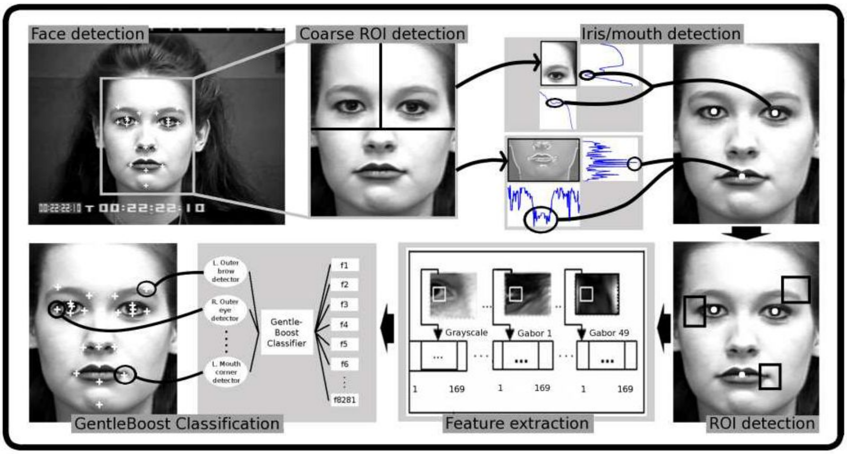


Fig. 1. Outline of the facial point detection system

(i.e., for every feature selected by AdaBoost), the proposed algorithm refines the feature originally proposed by AdaBoost. The algorithm creates a new set of filters generated by placing the original filter and slightly modified versions of the filter at a two pixels distance in each direction.

The method that we use for fully automatic detection of 20 facial feature points plus the irises and the centre of the mouth in a face image, uses Gabor-feature-based boosted classifiers as proposed in [18]. The method, outlined in Fig. 1, assumes that the input image is a face region, such as the output of the face detection algorithm explained above. In this face region, the irises and the medial point of the mouth are detected first. A combination of heuristic techniques based on the analysis of the vertical and horizontal histograms of the upper and the lower half of the face-region image achieves this. Based on these three points and anthropomorphic rules, the input face region is divided into 20 regions of interest (ROIs), each corresponding to a facial point to be detected.

For each pixel in the ROI, a feature vector is computed that consists of the grey values of the 13x13 patch surrounding the pixel and the responses to 48 Gabor filters (8 orientations and 6 spatial frequencies, 2:12 pixels/cycle at 1/2 octave steps). This feature vector is used to learn the pertinent point's patch template and, in the testing stage, to predict whether the current point represents a certain facial point or not.

To capture all facial motion, we track the detected points through all frames of the input video. The algorithm we used to track these facial points is Particle Filtering with Factorised Likelihoods (PFFL) [19]. We used the observation model proposed in [20], which is both insensitive to variations in lighting and able to cope with small deformations in the template. This polymorphic aspect is necessary as many areas around facial points change their appearance when a facial action occurs (e.g. the mouth corner in a smile). The facial point tracking scheme results for every image sequence with n frames in a set of points P with dimensions $20 * 2 * n$.

For all points $\boldsymbol{p_i} \in P$, where $i = [1 : 20]$ denotes the facial point, we compute first two features for every frame j to encode the y and the x coordinate deviation of a point relative to their position in the first frame of an input image sequence:

$$f_1 \left(\boldsymbol{p_i}, t\right) = p_{i,y,t} - p_{i,y,1} \tag{1}$$

$$f_2 \left(\boldsymbol{p_i}, t\right) = p_{i,x,t} - p_{i,x,1} \tag{2}$$

For all pairs of points $\boldsymbol{p_i}$, $\boldsymbol{p_j}$,$i \neq j$ we compute in each frame three features:

$$f_3(\boldsymbol{p_i}, \boldsymbol{p_j}, t) = \|p_{i,t} - p_{j,t}\| \tag{3}$$

$$f_4(\boldsymbol{p_i}, \boldsymbol{p_j}, t) = f_3(\boldsymbol{p_i}, \boldsymbol{p_j}, t) - \|p_{i,1} - p_{j,1}\| \tag{4}$$

$$f_5(\boldsymbol{p_i}, \boldsymbol{p_j}, t) = \arctan\left(\frac{\|p_{i,y,t} - p_{j,y,t}\|}{\|p_{i,x,t} - p_{j,x,t}\|}\right) \tag{5}$$

These features correspond to the distance between two points, the distance between two points relative to their distance in the first frame and the angle made by the line connecting two points and the horizontal axis. Also, to capture some of the temporal dynamics of a facial expression, we compute the first temporal derivative df/dt of all above defined features. This results in a set F of 840 features per frame.

The features f_1-f_5 are computed using the tracking information of at most two frames. As such, their temporal scope is limited. Not only does this mean that it is hard to model continuous behaviour of facial actions, it also makes the system very sensitive to inaccuracies of the tracker. A single tracking error will likely result in an outlier in our data representation. This, in turn, will lead to lower classification accuracy. To increase the temporal scope, we add a second set of features. Within a temporal window of duration T we describe each of the original mid-level parameters f_1-f_5 with a p^{th} order polynomial. We choose T and p such that with a given framerate we can accurately describe the feature shape in the fastest facial segment (the onset of AU45, a blink). For our data, recorded at 25 frames per second, this results in $T = 7$ and $p = 2$. Thus, the mid-level parameters f_{11}-f_{25} are found to be the values that fit the polynomial best:

$$f_k = f_{2k+1}t^2 + f_{2k+2}t + f_{2k+3} \ , \ k \in [1 \ldots 5] \tag{6}$$

The addition of the polynomial description of mid-level parameters adds another 1260 features per frame, bringing the total feature dimensionality to 2100.

3 Hybrid Classification

While the temporal dynamics of a facial action can be represented very efficiently by HMMs, the multiclass classification of the features on a frame-by-frame basis is normally done using Gaussian mixture models as the emission probabilities. These Gaussian mixtures are trained by likelihood maximisation, which assumes correctness of the models and thus suffers from poor discrimination [21]. It results in mixtures trained to model each class and not to discriminate one class from the other.

SVMs on the other hand discriminate extremely well. Using them as emission probabilities might very well result in an improved recognition. We therefore train a set of SVMs, one for every combination of classes (i.e., temporal phases neutral, onset, apex, and offset) and use their output to compute emission probabilities. This way we effectively have a hybrid SVM-HMM system. This approach has been previously applied with success to speech recognition [22].

Unfortunately we cannot use the output of a SVM directly as a probability measure. The output $h(\boldsymbol{x})$ of a SVM is a distance measure between a test pattern and the separating hyper plane defined by the support vectors. There is no clear relationship with the posterior class probability $p(y = +1|\boldsymbol{x})$ that the pattern $\boldsymbol{x}$ belongs to the class $y = +1$. Fortunately, Platt proposed an estimate for this probability by fitting the SVM output $f(\boldsymbol{x})$ with a sigmoid function [23]:

$$p(y = +1|\boldsymbol{x}) = g(h(\boldsymbol{x}), A, B) \equiv \frac{1}{1 + \exp(Ah(\boldsymbol{x}) + B)} \tag{7}$$

The parameters A and B of eq.(7) are found using maximum likelihood estimation from a training set p_i, t_i with $p_i = g(f(\boldsymbol{x}_i, A, B)$ and target probabilities $t_i = (y_i + 1)/2$. The training set can but does not have to be the same set as used for training the SVM.

Since SVMs are *binary* classifiers we use a one-versus-one approach to come to a multiclass classifier. This approach is to be preferred over the one-versus-rest approach as it aims to learn the solution to a more specific problem, namely, distinguishing between one class from one other class at a time. For this pairwise classification we need to train $K(K-1)/2$ SVMs, where in our case $K = 4$ is the number of temporal phases.

Our HMM consists of four states, one for each temporal phase. For each SVM we get, using Platt's method, pairwise class probabilities $\mu_{ij} \equiv p(q_i|\text{or}q_j, \boldsymbol{x})$ of the class (HMM state) q_i given the feature vector $\boldsymbol{x}$ and that $\boldsymbol{x}$ belongs to either q_i or q_j. These pairwise probabilities are transformed into posterior probabilities $p(q_i|\boldsymbol{x})$ by

$$p(q_i|\boldsymbol{x}) = 1/\left[\sum_{j=1, j\neq i}^{K} \frac{1}{\mu_{ij}} - (K-2)\right] \tag{8}$$

Finally, the posteriors $p(q|\boldsymbol{x})$ have to be transformed into *emission probabilities* by using Bayes' rule

$$p(\boldsymbol{x}|q) \propto \frac{p(q|\boldsymbol{x})}{p(q)} \tag{9}$$

where the a-priori probability $p(q)$ of class q is estimated by the relative frequency of the class in the training data.

4 Evaluation

We have evaluated our proposed methods on 196 videos selected from the MMI-Facial Expression Database [24], containing videos of 23 different AUs. We have chosen this database, instead of for example the Cohn-Kanade DFAT-504 dataset [25], because the videos in the MMI-Facial Expression Database display the full neutral-expressive-neutral pattern. This is essential, as it is this temporal pattern of facial actions that we are interested in. The videos were chosen so that we selected at least 10 videos of every AU we want to analyse.

The tests investigate the effects of our two proposals: adding the polynomial features described in section 2, and using the hybrid SVM-HMM system described in section 3. For ease of reference, in this section we denote the system using only the mid-level parameters $f_1 \dots f_{10}$ as the Traude system (for tracked action unit detector) and the system that uses all 25 mid-level parameters Traude-Plus. Thus we have compare four methods: Traude with SVM, Traude with SVM-HMM, Traude-Plus with SVM and Traude-Plus with SVM-HMM. A separate set of classifiers was trained for each AU. All methods perform feature selection using GentleBoost before training their respective classifiers (see [4] for details on feature selection using GentleBoost). Evaluation was done using 10-fold cross validation and measured the number of correctly classified frames, where the classes are neutral, onset, apex or offset. Table 1 shows the F1-value results per AU. The F1-value ϕ is a measure of performance that values recall as important as precision and is computed as follows:

$$\phi = \frac{2ab}{a+b} \tag{10}$$

where a is the recall and b the precision of a test. As we can see from table 1, the hybrid SVM-HMM method outperforms the SVM method for almost every AU. The effect of adding features $f_{11} \dots f_{25}$ is not that obvious. However, the mean of the results does suggest that the parametric features do benefit performance. We also see that the effect of the new features is more pronounced for the SVM-HMM classifier. This is another evidence that the new features indeed capture the temporal dynamics of the facial action, of which the SVM-HMM classifier makes full use.

Figure 2 shows the relative increase of performance of the temporal segments separately. In this figure we have averaged the results per temporal phase over all AUs. Again, from inspection we see that the SVM-HMM approach outperforms the SVM method. Shown are the recall, precision separately and finally the F1-measure.

We now turn our attention to the significance of our results. We first compare the results of the Traude with the TraudePlus methods. When we consider the averaged performance rate of all AUs and all temporal segments together, the

Table 1. F1-value for the four tested systems, for all Action Units (AUs)

AU	Traude SVM	TraudePlus SVM	Traude SVM-HMM	TraudePlus SVM-HMM
1	0.614	0.612	0.703	0.714
2	0.590	0.657	0.663	0.734
4	0.584	0.570	0.623	0.607
5	0.447	0.480	0.482	0.560
6	0.579	0.496	0.672	0.614
7	0.275	0.289	0.362	0.327
9	0.699	0.638	0.763	0.790
10	0.620	0.656	0.715	0.678
12	0.719	0.780	0.797	0.786
13	0.817	0.804	0.836	0.839
15	0.491	0.523	0.583	0.612
16	0.615	0.620	0.581	0.637
18	0.719	0.723	0.742	0.726
20	0.732	0.731	0.801	0.804
22	0.570	0.611	0.677	0.658
24	0.443	0.380	0.393	0.379
25	0.694	0.715	0.705	0.735
26	0.582	0.607	0.636	0.607
27	0.743	0.686	0.840	0.815
30	0.527	0.484	0.564	0.595
43	0.662	0.710	0.716	0.778
45	0.685	0.718	0.714	0.787
46	0.378	0.382	0.332	0.397
Mean:	0.599	0.603	0.648	0.660

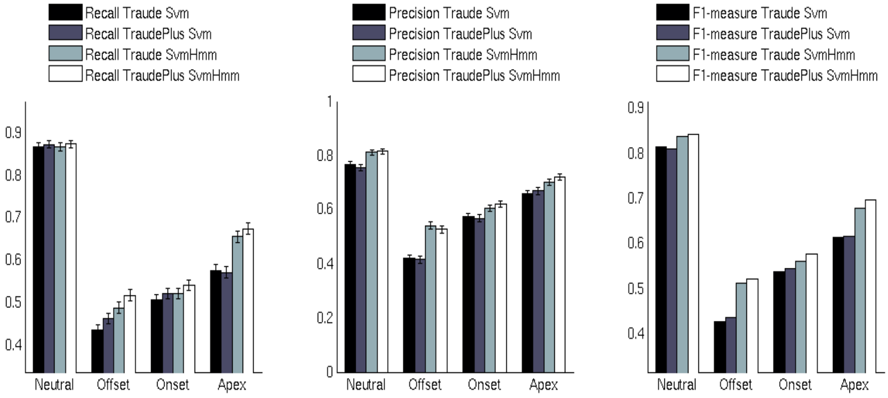


Fig. 2. Comparison of the classification results shown per temporal phase (onset, apex, offset and neutral). The results shown are the average over all 23 AUs, error bars depict the standard deviation.

recall of TraudePlus is 1.8% better and the precision is 0.5% better. However, these performance increases are not significant at a 95% confidence level. Comparing the SVM method with the SVM-HMM method, we find that the recall of SVM-HMM is 4.5% better and the precision even 7% better. Both performance increases *are* significant at a 95% confidence level.

When we take a closer look at the results of the detection of the temporal segments, we see that the detection of the offset phase has increased most from introducing the HMM, achieving an increase in recall of 6% and an increase in precision of 10.6%. The apex phase is a good second, with a recall increase of 10.1% and a precision increase of 5.1%. The neutral phase benefits least from the addition of the HMM. This is as expected, because by its very nature it is not a dynamic part of the facial action, apart from tracker noise the facial points are stationary during the neutral phase.

5 Conclusion

We have presented and evaluated two methods to improve the analysis of a facial action's temporal dynamics in terms of classifying the temporal phases neutral, onset, apex and offset. From our results we can conclude that replacing the SVM classifier with a hybrid SVM-HMM classifier results in a significant classification accuracy improvement. It is harder to judge the effect of adding the polynomial representation of the original mid-level parameters. Using these features, there seems to be a slight but statistically insignificant improvement, which is more pronounced in combination with the SVM-HMM classifier than it is with the original SVM classifier. However, the performance increase due to the polynomial mid-level parameters is not significant and might not be high enough to make up for the fact that we have to compute 2.5 times as many features.

References

1. Ekman, P., Friesen, W.V., Hager, J.C.: Facial Action Coding System. A Human Face, Salt Lake City (2002)
2. Cohn, J.F.: Foundations of human computing: Facial expression and emotion. In: Proc. ACM Int'l Conf. Multimodal Interfaces, vol. 1, pp. 610–616 (2006)
3. Tian, Y., Kanade, T., Cohn, J.: Recognizing action units for facial expression analysis. IEEE Trans. Pattern Analysis and Machine Intelligence 23(2), 97–115 (2001)
4. Valstar, M.F., Pantic, M.: Fully automatic facial action unit detection and temporal analysis. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, p. 149 (2006)
5. Bartlett, M.S., Littlewort, G., Lainscsek, C., Fasel, I., Movellan, J.: Machine learning methods for fully automatic recognition of facial expressions and actions. In: Proc. IEEE Int'l Conf. on Systems, Man and Cybernetics, vol. 1, pp. 592–597 (2004)
6. Pantic, M., Rothkrantz, L.J.M.: Toward an affect-sensitive multimodal human-computer interaction. Proc. IEEE 91(9), 1370–1390 (2003)
7. Tian, Y.L., Kanade, T., Cohn, J.F.: Handbook of Face Recognition. Springer, Heidelberg (2005)
8. Cohn, J.F., Schmidt, K.L.: The timing of facial motion in posed and spontaneous smiles. J. Wavelets, Multi-resolution and Information Processing 2(2), 121–132 (2004)
9. Bassili, J.N.: Facial motion in the perception of faces and of emotional expression. J. Experimental Psychology 4(3), 373–379 (1978)

10. Hess, U., Kleck, R.E.: Differentiating emotion elicited and deliberate emotional facial expressions. European J. of Social Psychology 20(5), 369–385 (1990)
11. de Williams, A.C.: Facial expression of pain: An evolutionary account. Behavioral and Brain Sciences 25(4), 439–488 (2006)
12. Ekman, P.: Darwin, deception, and facial expression. Annals of New York Ac. of sciences 1000, 105–221 (2003)
13. Ekman, P., Rosenberg, E.L.: What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System. Oxford University Press, Oxford, UK (2005)
14. Valstar, M.F., Pantic, M., Ambadar, Z., Cohn, J.F.: Spontaneous vs. posed facial behavior: automatic analysis of brow actions. In: Proc. ACM Intl. conf. on Multimodal Interfaces, pp. 162–170 (2006)
15. Fasel, I.R., Fortenberry, B., Movellan, J.R.: A generative framework for real time object detection and classification. Comp. Vision, and Image Understanding 98(1), 181–210 (2005)
16. Viola, P., Jones, M.: Robust real-time object detection. Technical report CRL 200001/01 (2001)
17. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: A statistical view of boosting. The Annals of Statistics 28(2), 337–374 (2000)
18. Vukdadinovic, D., Pantic, M.: Fully automatic facial feature point detection using gabor feature based boosted features. In: Proc. IEEE Int'l Conf. on Systems, Man and Cybernetics, pp. 1692–1698 (2005)
19. Patras, I., Pantic, M.: Particle filtering with factorized likelihoods for tracking facial features. In: Proc. Int'l Conf. Automatic Face & Gesture Recognition, pp. 97–102 (2004)
20. Patras, I., Pantic, M.: Tracking deformable motion. In: Proc. Int'l Conf. Systems, Man and Cybernetics, pp. 1066–1071 (2005)
21. Bourlard, H., Morgan, N.: Hybrid hmm/ann systemds for speech recognition: Overview and new research directions. In: Giles, C.L., Gori, M. (eds.) Adaptive Processing of Sequences and Data Structures. LNCS (LNAI), vol. 1387, pp. 389–417. Springer, Heidelberg (1998)
22. Kruger, S.E., Schaffner, M., Katz, M., Andelic, E., Wendemuth, A.: Speech recognition with support vector machines in a hybrid system. In: Interspeech, pp. 993–996 (2005)
23. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularized likelihood methods, pp. 61–74. Cambridge, MA (2000)
24. Pantic, M., Valstar, M.F., Rademaker, R., Maat, L.: Web-based database for facial expression analysis. In: Proc. Int'l Conf. Multimedia & Expo, pp. 317–321 (2005)
25. Kanade, T., Cohn, J., Tian, Y.: Comprehensive database for facial expression analysis. In: IEEE Int'l Conf. on Automatic Face and Gesture Recognition, pp. 46–53 (2000)

Pose and Gaze Estimation in Multi-camera Networks for Non-restrictive HCI

Chung-Ching Chang, Chen Wu, and Hamid Aghajan

Wireless Sensor Networks Lab, Stanford University, Stanford, CA 94305 USA

Abstract. Multi-camera networks offer potentials for a variety of novel human-centric applications through provisioning of rich visual information. In this paper, face orientation analysis and posture analysis are combined as components of a human-centered interface system that allows the user's intentions and region of interest to be estimated without requiring carried or wearable sensors. In pose estimation, image observations at the cameras are first locally reduced to parametrical descriptions, and Particle Swarm Optimization (PSO) is then used for optimization of the kinematics chain of the 3D human model. In face analysis, a discrete-time linear dynamical system (LDS), based on kinematics of the head, combines the local estimates of the user's gaze angle produced by the cameras and employs spatiotemporal filters to correct any inconsistencies. Knowing the intention and the region of interest of the user facilitates further interpretation of human behavior, which is the key to non-restrictive and intuitive human-centered interfaces. Applications in assisted living, speaker tracking, and gaming can benefit from such unobtrusive interfaces.

1 Introduction

Human computer interaction by means of video analysis has found a variety of applications including gaming, assisted living, smart presentations, virtual reality and many other smart interfaces. Traditionally, the user is supposed to face the camera to facilitate gesture estimation and achieve system robustness. However, fixing the user's facing direction on one hand reduces flexibility of gestures and on the other hand limits the interacting subject being restricted to the camera / computer facing the user. In some more complex scenarios, the user may be moving around interacting with different subjects, or the user may be distracted by other events in the environment thus doing some gestures not intended to the system. In such situations, it is necessary to add face orientation in addition to gestures to allow for flexibility for the user to naturally participate in the virtual environment without switching the context manually. Furthermore, combining these two observations may give more variety in explaining the user's behaviors.

Enabling a smart environment which is aware of both the specific interacting subject and the command motivates us to combine face orientation and human posture estimation. The face orientation provides the gaze focus or region of

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 128–137, 2007.

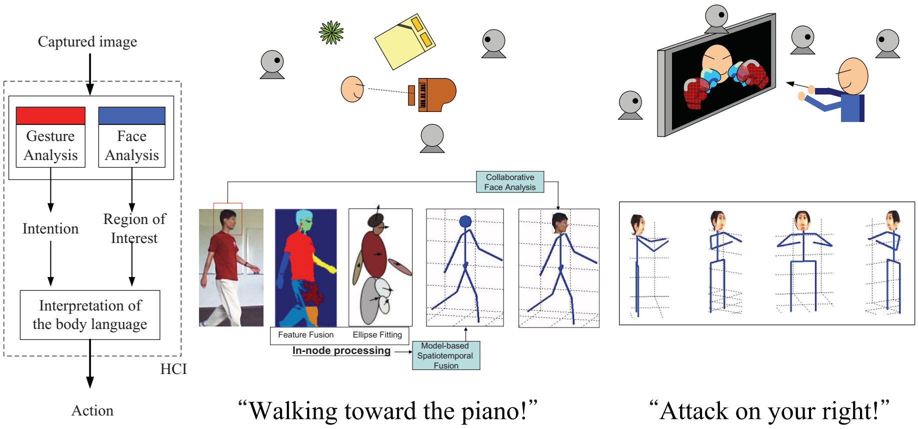


Fig. 1. Framework and two examples of the HCI

interest of the user, and the posture conveys commands to the computer. Human posture estimation is a research topic with growing interest. Among the approaches based on monocular cameras, a number of optimization techniques have been adopted to attack the high dimensionality. Good performance has been achieved using techniques such as annealed particle filtering [1], covariance scaled sampling [2] etc. Dynamic motion models have also been applied to predict and remove ambiguity [3]. In multi-view approaches, given calibrated cameras 3D voxels of the person can be reconstructed which transforms the problem into optimization in the direct 3D space [4,5]. Sigal et al. [6] combine a bottom-up and discriminative approach to detect and assemble body parts in images, and infer a 3D pose from training data. Typically estimating face orientation is not within the scope of posture estimation since the head is not modeled in details there. However, in many situations as stated above, face orientation is of particular interest. In our conception the cameras in the network implement two task modules, face orientation estimation and posture estimation. Combining these two outputs, a human model can be recreated with sufficient details for smart interfaces. They can also be used for high-level activity analysis and communication without transferring high-bandwidth video data.

In Section 2, the schematic of the whole system is depicted. In the following sections, face orientation and posture estimations in a multi-camera network are described and examples are given.

2 Framework

In this paper, we propose a framework for HCI in vision sensor networks that incorporates both gesture analysis and face analysis. The face orientation analysis provides the information of where the user is looking at, in other words, the user's region of interest. On the other hand, the gesture analysis module provides information on what the person is doing, which reveals the user's intention. Based on the region of interest and the intention of the person, the network can interpret the user's body language and receive commands, predict imminent events or actions such as accidents in assisted living, and provide suitable responses in applications such as gaming and virtual reality.

Fig.1 illustrates the idea through an example. Three cameras are deployed casually in an indoor environment, each with two context algorithms, one running on full-body images to get the person's pose and one focusing only on the head and face. Practically, this can be done by starting from high-resolution images (e.g. VGA) and giving two proper areas in the images to each algorithm. Face analysis provides information about the gaze of the user, e.g. that the user is looking at the piano. Pose and gesture analysis provides information about the action of the user,e.g. that the user is walking. Therefore, the network may predict that the user is walking toward the piano. The HCI system, which is effectively a part of a smart environment application in this case, may react accordingly; for example instead of turning the light off (is the user was walking towards the bed), decides to turn on the spotlight on top of the piano.

3 Gesture Analysis

We propose a framework of opportunistic fusion in multi-camera networks in order to both employ the rich visual information provided by cameras and incorporate learned knowledge of the subject into active vision analysis. The opportunistic fusion framework is composed of three dimensions, space, time, and feature levels, as shown in Fig. 2. On the top of Fig. 2 are spatial fusion modules. In parallel is the progression of the 3D human body model. Suppose at time t_0 we have the model with the collection of parameters as M_0. At the next instance t_1, the current model M_0 is input to the spatial fusion module for t_1, and the output decisions are used to update M_0 from which we get the new 3D model M_1. Details of the layered gesture analysis on the bottom-left of Fig. 2 see [7].

The implementation of the 3D human body posture estimation incorporating elements in the opportunistic fusion framework described above is illustrated in Fig. 3. Local processing in a single camera includes segmentation and ellipse fitting for a concise parameterization of segments. We assume the 3D model is initialized with a distinct color distribution for the subject. For each camera, the color distribution is first refined using the EM (Expectation Maximization) algorithm and then used for segmentation. Undetermined pixels from EM are assigned labels through watershed segmentation. For spatial collaboration, ellipses from all cameras are merged to find the geometric configuration of the 3D skeleton model. Candidate configurations are examined using PSO (Particle Swarm Optimization).

3.1 In-Node Processing

The goal of local processing in a single camera is to reduce raw images/videos to simple descriptions so that they can be efficiently transmitted between the

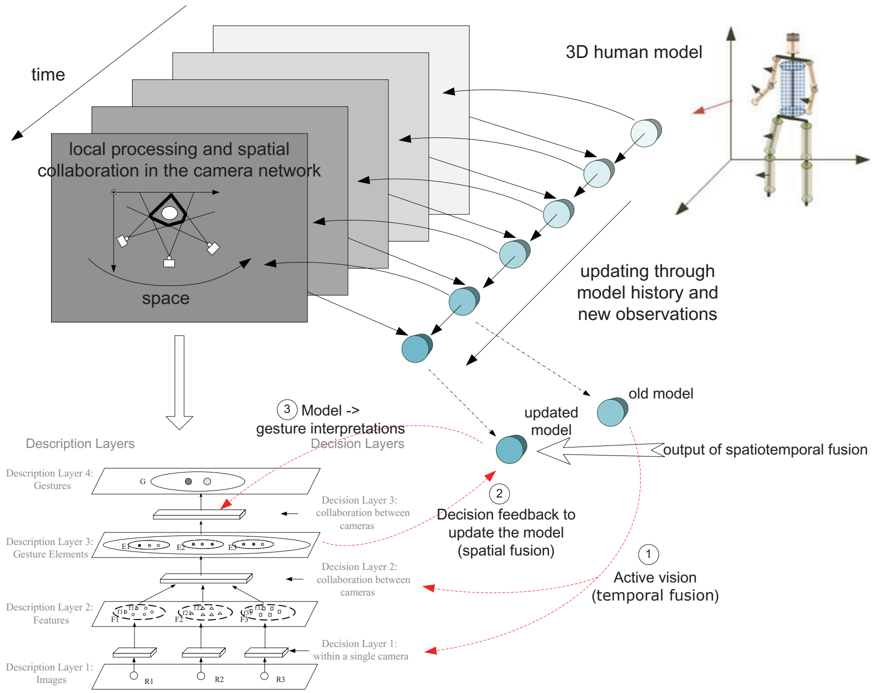

time
local processing and spatial collaboration in the camera network
space
3D human model
updating through model history and new observations
old model
updated model
output of spatiotemporal fusion
3 Model -> gesture interpretations
Description Layers
Decision Layers
Description Layer 4: Gestures
G
Description Layer 3: Gesture Elements
E1
E2
E3
Description Layer 2: Features
f11
f12
f21
f22
f31
f32
F1
F2
F3
Description Layer 1: Images
R1
R2
R3
Decision Layer 3: collaboration between cameras
Decision Layer 2: collaboration between cameras
Decision Layer 1: within a single camera
2 Decision feedback to update the model (spatial fusion)
1 Active vision (temporal fusion)

Fig. 2. Spatiotemporal fusion framework for human gesture analysis

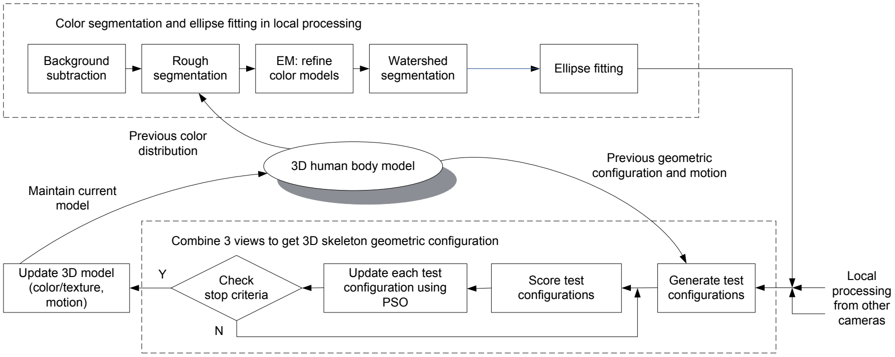

Color segmentation and ellipse fitting in local processing
Background subtraction
Rough segmentation
EM: refine color models
Watershed segmentation
Ellipse fitting
Previous color distribution
3D human body model
Previous geometric configuration and motion
Maintain current model
Combine 3 views to get 3D skeleton geometric configuration
Update 3D model (color/texture, motion)
Y
Check stop criteria
N
Update each test configuration using PSO
Score test configurations
Generate test configurations
Local processing from other cameras

Fig. 3. Algorithm flowchart for 3D human skeleton model reconstruction

cameras. The output of the algorithm will be ellipses fitted from segments and the mean color of the segments. As shown in the upper part of Fig. 3, local processing includes image segmentation for the subject and ellipse fitting to the extracted segments.

We assume the subject is characterized by a distinct color distribution. Foreground area is obtained through background subtraction. Pixels with high or low illumination are also removed since for those pixels chrominance may not be reliable. Then a rough segmentation for the foreground is done either based on K-means on the chrominance of the foreground pixels, or color distributions from the model. A POEM (Perceptually Organized EM) algorithm is then used to refine the color distribution for the current image. It takes into account both pixel value and spatial approximities. The initial estimated color distribution plays an important role because it can prevent EM from being trapped in local minima. After refinement of the color distribution with POEM, we set pixels with high probability (e.g., larger than 99.9%) that belong to a certain mode as markers for that mode. Then a watershed segmentation algorithm is implemented to assign labels for undecided pixels. Finally, in order to obtain a concise parameterization for each segment, an ellipse is fitted to it. Some examples are shown in Fig. 4(a).

3.2 Multi-view Posture Estimation

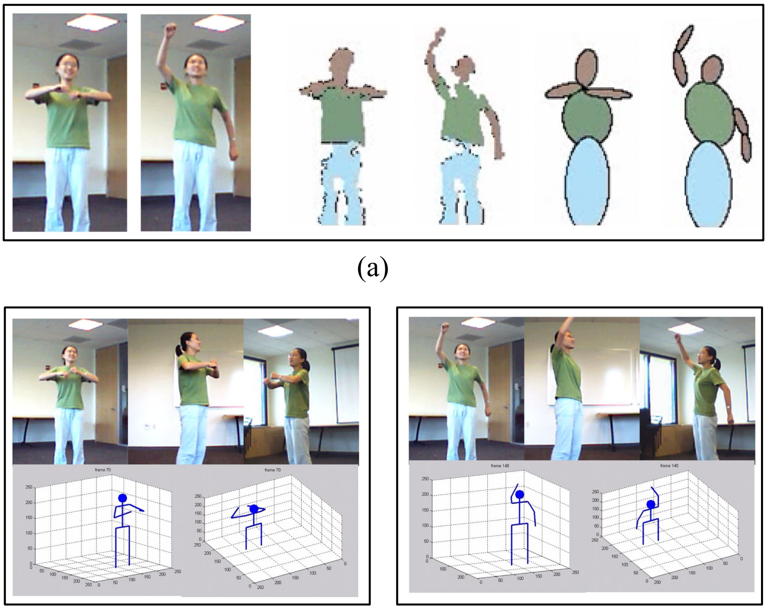
Particle Swarm Optimization (PSO) is used for the optimization of the kinematic chain of the human model, motivated by [8,9]. The assumption is that projection matrices from 3D skeleton to 2D image planes are known. PSO is suitable for posture estimation as an evolutionary optimization mechanism. It starts from a group of initial particles. During the evolution the particles are directed to the good position while keeping some randomness to explore the search space. PSO is likely to converge to local optimum without carefully choosing the initial particles similar to the other searching techniques. In the experiment we assume that the 3D skeleton will not go through a big change in a time interval. Therefore, at time t_1 the search space formed by the particles is centered around the optimal solution of the geometric configuration at time t_0. We can also allow for larger randomness at the beginning to explore more of the search space for local minima, and gradually increase inertia later on for convergence. Some examples showing images from 3 views and the posture estimates are shown in Fig. 4(b).

4 Face Orientation Analysis

In multiple camera settings, face orientation estimation is composed of three dimensions: time, space, and feature levels. The in-node processing is designed to extract two features, face orientation and angular motion, from the captured frames. The following collaborative face orientation analysis, as shown in Fig. 2, forms a linear dynamical system based on the relations between these two features, one being the derivative of the other, along with spatiotemporal constraints to optimize the estimates. The methodology described in this section follows the work proposed by Chang et al. [10].

4.1 In-Node Processing

The purpose of the in-node processing is to extract two features from the captured frames, face orientation and face angular motion, which we call local orientation/angular estimates in the rest of the paper. By doing this we reduce



(a)

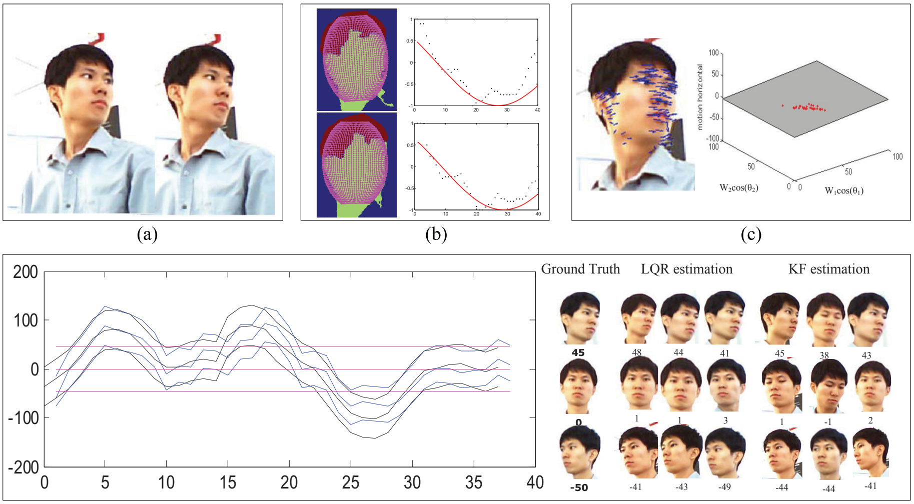
(b)

Fig. 4. Examples for gesture analysis in the vision network. (a) In-node segmentation results. (b) Skeleton model reconstruction by collaborative fusion.

the amount of data transferred in the network, enabling the scalability of the camera networks. In addition, in-node processing algorithms are designed to be lightweight to reduce computational complexity in each node, with the some of the resulting local estimates with higher errors. On the other hand, we aim to incorporate information from multiple nodes to enhance the accuracy of the estimates, which is called the joint estimate.

Following the in-node processing of the posture estimation as in Fig.3, head ellipse can be depicted through color and spatial information (normally the ellipse with skin color). Further hair and skin region segmentation based on color is applied on the head ellipse, as the red and green regions shown in Fig. 5(b). Next, we sample the hair-to-face ratio on each slice of the head. The slices are equally spaced along the longest axis of the head, as shown by the magenta lines. The hair-to-face ratio is periodic around the head, smaller in the front and bigger in the back, and, without loss of generality, it is assumed to be sinusoidal. Therefore, the face orientation can be predicted by the phase of the fitted sinusoidal curve to the hair-to-face ratio, as shown in Fig. 5(b).

The local angular motion estimation is calculated by statistical model-based analysis. First of all, we find the corresponding interest points in face ellipsoid between two consecutive frames. The displacement between them is the motion vectors of the head, as shown in Fig.5(c). When the head turns, the geometrical projection of the motion vector on the image frame in different parts of the face



(a) (b) (c)

(d)

Fig. 5. Examples for face orientation analysis in the vision network. (a) two consecutive captured images. (b) Local face orientation estimation results. (c) Local face angular motion estimation result. (d) Collaborative face orientation analysis and the corresponding face profile collections.

may differ. Assuming the head to be an ellipsoid, the translational and angular motion of the head are estimated.

The confidence level can be defined through some nonparametric statistical algorithms, such as Central Limit Theorem or Bootstrap Percentile Confidence Interval [11]. For simplicity, we adopt the former idea and taking the residual of the LS analysis, which is the sum of the bias and variance, as the confidence level of each local estimate with respect to the spatial information. On the other hand, we also measure the temporal inconsistency, such as the parameter differences of the head fitted between two consecutive frames. The temporal confidence level is a nonlinear function of the temporal inconsistency, as defined in [10]. We assume that the temporal and the spatial confidence levels are independent; therefore, the overall confidence level for each local estimate is defined as the multiplication of the spatial and the temporal confidence level.

Since in-node signal processing is intentionally designed to be lightweight, we should accept the fact that the resulting feature estimates in each camera might be erroneous. On the other hand, the camera nodes exchange information with each other, allowing for the network to enhance the estimates' accuracy, and to produce accurate results describing the orientation of each facial view.

4.2 Collaborative Face Orientation Estimation

The joint estimation problem can be defined as in Fig. 6. All the nonlinearity of 3D geometry is handled by local estimation, yielding orientation and angular motion in each direction that can be linearly combined. Based on kinematics of

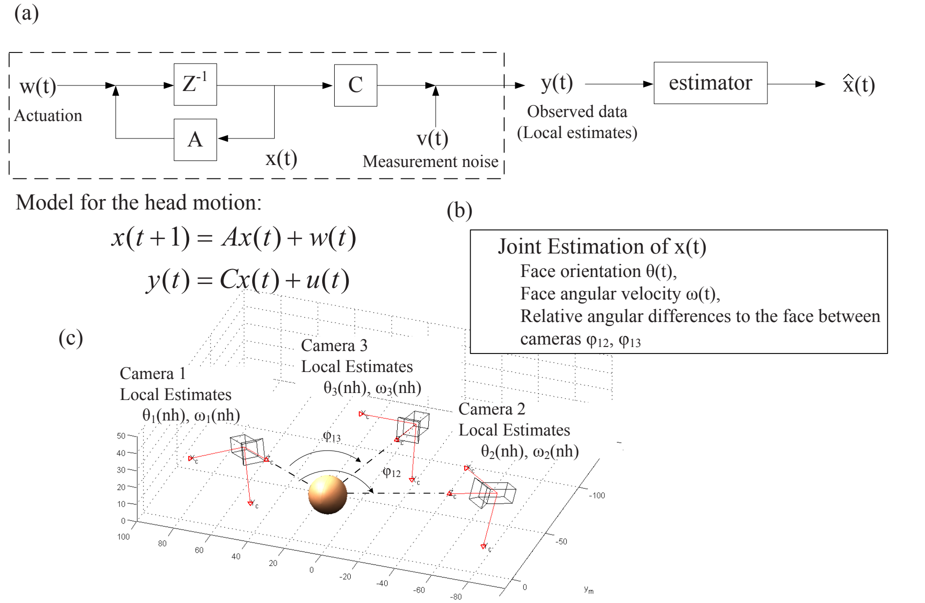


Fig. 6. Problem definition and modeling of the collaborative face orientation estimation problem. (a) Block diagram of the modeled system. (b) The objective of the problem. (c) the observation given for the joint analysis problem.

the head, we define the relations between in-node estimation attributes by a Linear Dynamical System (LDS) and measure the spatiotemporal inconsistencies. Let the state x represent the true face orientation, true face angular motion, and true relative angular differences to the face between cameras. The update matrix A is defined based on the fact that the face orientation is the integral of the face angular motion. In addition, we assume that the head angular motion is actuated by $w(t)$, an IID Gaussian distributed actuation factor.

To estimate face orientation over a LDS, a direct approach would be to formulate the problem based on MMSE, minimizing the mean square error between the joint estimates and the observed data $y(t)$. This can be done by least squares analysis (LS) or, equivalently, Linear Quadratic Regulation (LQR), an iterative method to solve the LS problems. The spatial inconsistency is described by a quadratic cost function. By minimizing the cost function by LQR, a stable close-loop feedback system that gives robust estimates is obtained.

Alternatively, the Kalman Filter (KF) approach employs selective feature measurements and their corresponding confidence levels. In doing measurement updates, local estimates are weighted according to their confidence levels. Moreover, the current joint estimate depends only on the previous joint estimate and the current local estimates. Therefore, this method is more practical for real-time systems.

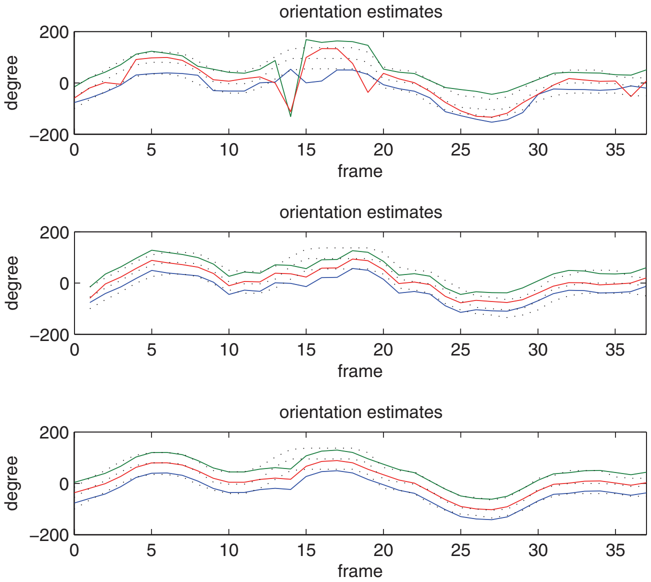


Fig. 7. Collaborative face analysis before and after data fusion. From top to bottom, estimation results for local estimation, KF-based joint estimation, and LQR-based estimation, respectively. The dotted black lines in all three figures show the true face orientation.

5 Human Behavior Interpretation

Having access to interpretations of region of interest and intention elements obtained from visual data over time enables higher-level reasoning modules to deduct the user's actions, context, and behavior models, and decide upon suitable actions or responses to the situation.

Our notion of the role a vision network can play in enabling novel intelligent human behavior interpretation is illustrated in the rightmost column in Fig.1 (take gaming applications for example). One recent success in the product "Wii" shows the importance of having non-restrictive and intuitive human-computer interactions. Instead of holding or wearing sensors, the vision network can interpret the human gestures and actions from the images, and interact with the virtual opponent. In addition, the face orientation, i.e. the field of view, is estimated in the vision network, creating the screen display with the right view angle to the user.

Besides gaming applications, capability of interpreting human behaviors enables the distributed vision-based reasoning for smart home care, which has a significant impact on the assisted living and monitored care for the elderly and persons in need of such care. The pose and gaze analysis provide information that triggers alert when an accident, such as cardiac arrest, happens.

6 Conclusion

In a multi-camera network novel potentials exist for efficient vision-based HCI applications if the rich visual information is appropriately employed. In this paper, a framework for opportunistic joint face analysis and gesture analysis is proposed. The face analysis predicts the region of interest of the person while the gesture analysis module estimates the intention of the person. Both face and gesture analysis encompass the three possible dimensions of data fusion, i.e., space, time, and feature levels. Capability of estimating the region of interest and the intention of the user facilitates further human behavior understanding in the context of smart environment applications.

References

1. Deutscher, J., Blake, A., Reid, I.D.: Articulated body motion capture by annealed particle filtering. In: CVPR, pp. 126–133 (2000)
2. Sminchisescu, C., Triggs, B.: Estimating articulated human motion with covariance scaled sampling. International Journal of Robotics Research 22(6), 371–393 (2003)
3. Sidenbladh, H., Black, M.J., Sigal, L.: Implicit probabilistic models of human motion for synthesis and tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2350, pp. 784–800. Springer, Heidelberg (2002)
4. Cheung, K.M., Baker, S., Kanade, T.: Shape-from-silhouette across time: Part ii: Applications to human modeling and markerless motion tracking. International Journal of Computer Vision 63(3), 225–245 (2005)
5. Mikic, I., Trivedi, M., Hunter, E., Cosman, P.: Human body model acquisition and tracking using voxel data. Int. J. Comput. Vision 53(3), 199–223 (2003)
6. Sigal, L., Black, M.J.: Predicting 3d people from 2d pictures. In: Perales, F.J., Fisher, R.B. (eds.) AMDO 2006. LNCS, vol. 4069, pp. 185–195. Springer, Heidelberg (2006)
7. Wu, C., Aghajan, H.: Layered and collaborative gesture analysis in multi-camera networks. In: ICASSP (April 2007)
8. Ivecovic, S., Trucco, E.: Human body pose estimation with pso. In: IEEE Congress on Evolutionary Computation, pp. 1256–1263. IEEE Computer Society Press, Los Alamitos (2006)
9. Robertson, C., Trucco, E.: Human body posture via hierarchical evolutionary optimization. In: BMVC 2006, vol. III, p. 999 (2006)
10. Chang, C., Aghajan, H.: Linear dynamic data fusion techniques for face orientation estimation in smart camera networks. In: ICDSC 2007, Vienna, Austria (to appear, 2007)
11. Rice, J.A.: Mathematical Statistics and Data Analysis, 3rd edn. Thomson Brooks/Cole, California, USA (2007)

Exact Eye Contact with Virtual Humans

Andrei State

Department of Computer Science, University of North Carolina at Chapel Hill
CB 3175 Sitterson Hall, Chapel Hill, North Carolina 27599, USA
and
InnerOptic Technology Inc., P. O. Box 824
Chapel Hill, NC 27514-0824, USA
andrei@cs.unc.edu

Abstract. This paper describes a simple yet effective method for achieving accurate, believable eye contact between humans and computer-generated characters, which to the author's knowledge is demonstrated here for the first time. A prototype system provides a high-fidelity stereoscopic head-tracked virtual environment, within which the user can engage in eye contact with a near-photorealistic virtual human model. The system does not require eye tracking. The paper describes design and implementation details, and reports qualitative positive feedback from initial testers.

Keywords: Eye Contact, Virtual Human, Gaze Direction, Interactive Virtual Environment.

1 Introduction and Related Work

Future Human-Computer Interface technologies will include virtual humans (VHs) capable of fluent, meaningful conversations and collaborations with humans [1]. The effectiveness of such interactions will depend in part on VHs' ability to appear and act like real people [2][3]. Eye contact is essential in human interaction [4] and therefore should be given careful consideration in the design of VHs and other virtual interaction partners. Believable eye contact with such entities will be most important in stereoscopic head-tracked virtual environments (SHVEs), whether using tracked head-mounted displays [5] or head-tracked "fish tank" techniques [6], since the viewpoint-matched stereoscopic imagery continually maintains the user's awareness of spatial relationships.

Beyond HCI and gaming, other areas that may benefit from accurate eye contact with virtual entities include robotics [7] and advertising, which is likely to conceive intrusive virtual characters that stare at us. Notwithstanding the latter, this paper presents simple yet effective techniques for exact eye contact (EEC) between a human user and a VH. The high accuracy is demonstrated in a precisely calibrated, stereoscopic head-tracked viewing environment. While the current implementation requires a head-mounted tracker, future embodiments may use un-encumbering tracking, such as vision-based head pose recovery. It is important to note that the technique described here does not require pupil tracking; it uses only head pose, which can

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 138–145, 2007.

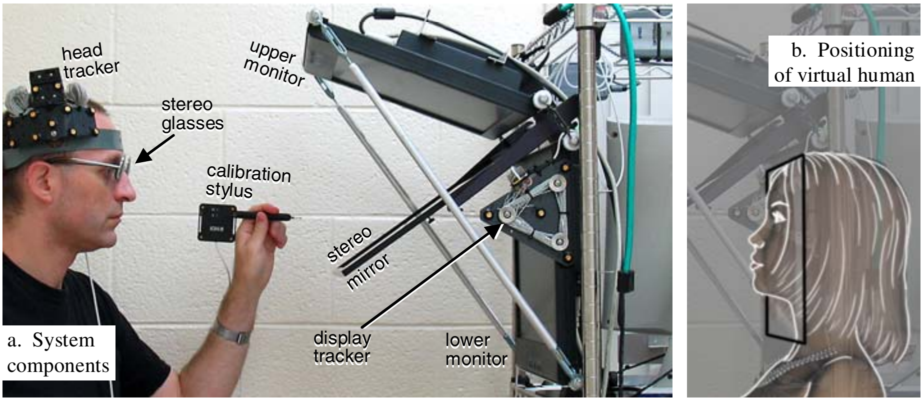
generally be obtained less intrusively, with higher reliability, and from a greater distance away than camera-based pupil tracking. An additional pupil tracker is not required unless the system must know the user's gaze direction, for example in order to record user behavior in training applications [3].

2 A Prototype System for Exact Eye Contact

The EEC prototype uses a fish tank SHVE (Fig. 1) consisting of a personal computer and the following main components:

- a Planar Systems SD1710 ("Planar") stereoscopic display with two 17" LCD monitors and a semi-transparent mirror that reflects the upper monitor's image onto the lower monitor. The user wears linearly polarized glasses that restrict viewing of the lower monitor to the left eye and viewing of the upper monitor's reflection to the right eye. The LCDs' native resolution is 1280×1024. To improve alignment between the monitors, an adjustable stabilization frame was added (Fig. 1a).
- a sub-millimeter precision Northern Digital Optotrak Certus opto-electronic tracking system ("Certus"). Both the Planar and the user's head are tracked by the Certus in all six degrees of freedom with clusters of infrared (IR) LEDs (11 on the head, 4 on the Planar). The advantage of tracking the display as in handheld augmented reality applications [8] is that both the display and the tracker can be moved with respect to each other while the system is running, for example, to improve LED visibility. The Certus also provides a calibration stylus for precise measurements.

To use the EEC prototype, the user dons the head tracker and first performs a simple, fast eye calibration procedure. This is followed by a live interaction phase, during which the user can engage in eye contact and interact with the VH.



a. System components

b. Positioning of virtual human

Fig. 1. Head-tracked virtual environment for eye contact with a virtual human uses a two-LCD stereoscopic display. The right-hand image shows the positioning of the life-size virtual human within the (rather small) display.

2.1 Projection Origin and Eye Calibration

In SHVEs, the calibration between the head tracker and the eyes is usually obtained from measurements such as the user's inter-pupillary distance (IPD, measured with a pupillometer) [5], the location of the tracker on the user's head, as well as from assumptions about the most suitable location of the projection origin inside the eye. Popular choices for the latter include the eye's 1st nodal point [6], the entrance pupil [9], and the center of the eye [10]. The EEC prototype uses the eye center [10] because it is easy to calibrate and yields exact synthetic imagery in the center of the field of view regardless of the user's gaze; a further advantage will be described in 2.2. However, the 1st nodal point and the entrance pupil are better approximations for the actual optics within the eye. Therefore, by rendering stereo images from the eye centers, i.e. from a few mm too far back, and thus with a slightly exaggerated separation, the EEC system deforms the stereoscopic field [11] ever so slightly. For higher accuracy, a pupil tracker could detect the user's gaze directions, and assuming that the user converges onto the virtual object found along those directions, one could move the projection origins forward to the 1st nodal point, or all the way to the pupil.

Calibration. The eye calibration technique (Fig. 2) was inspired by previous methods [12][13] and modified for the additional display tracker. A small panel with a circular hole is temporarily mounted in front of the bottom LCD panel. Both the hole and the bottom LCD monitor are pre-calibrated (one-time only) to the Planar's tracker with the Certus stylus. The eye calibration program shows a circular disk on the display. Using a "mirror image" of the user's head as a guide, the user moves and orients his head to line up the disk through the hole, twice through each eye, under different head orientations. To avoid confusion, users wear frames with one eye masked off, as shown by the "mirror" guides at the top of Fig. 2. The program collects four line equations in head tracker coordinates. In pairs of two, these four lines define the eye centers at their intersections—or rather, at the closest points between them. The entire task takes 1-2 minutes except for inexperienced first-time users, which take longer mostly because they must receive and follow instructions.

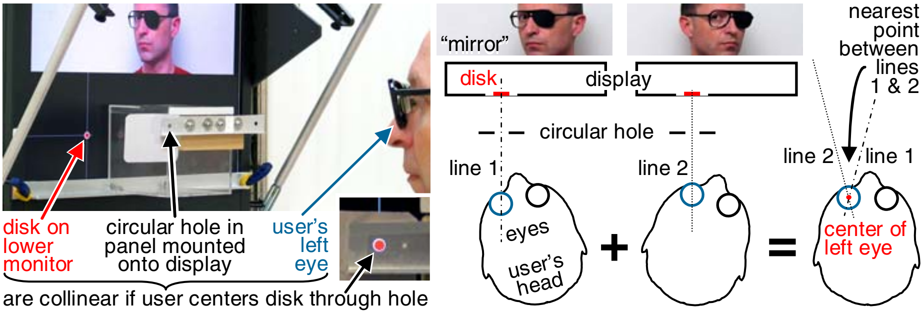


Fig. 2. Eye calibration setup and sequence, shown here for the left eye only

Since the current head tracker (Fig. 1a) does not guarantee repeatable positioning on the user's head, the user should not remove it between calibration and the following interactive phase. User-specific head-conforming gear equipped with IR LEDs—or with passive markers for camera-based head tracking—could remove this restriction and could thus reduce each user's eye calibration to a one-time procedure.

2.2 Interactive Operation

During this phase, the system displays a life-size stereoscopic head-and-shoulders view of a VH as seen from the user's eye centers. The latter are computed from the head tracker reports and from the eye calibration measurements described above.

Virtual Human Model. The human model (Fig. 3, "Carina" from Handspan Studios, slightly reshaped and edited for fast rendering with the DAZ Studio modeler from DAZ Productions, Inc.), is displayed via OpenGL in separate views for the Planar's two monitors, using off-center viewports; the model is near-photorealistic but simple enough for real-time rendering. The 2.6GHz dual-core Opteron PC equipped with an NVIDIA 7800GT graphics adapter achieves a frame rate of approximately 60Hz. Only head and shoulders could be fitted in life size within the limited virtual space of the small display (Fig. 1b). The model is completely static except for the eyeballs, which have separate rotation transformations for azimuth and elevation. Thus the eyes can be oriented toward a given target. A polygonal "occlusion envelope" in each eye's azimuth-elevation space (Fig. 3, right) approximates the eyelid contour and helps detect if the eye "tries" to look at a target "through" skin or eyelids, which could cause the eye to roll up into the head, for example. Whenever either eye encounters such a situation, *both* eyes switch to looking straight ahead. The occlusion envelopes are not anatomically accurate; they simply match the 3D model. The eyes are displayed with high-detail textures for the iris as well as with transparent, specular corneas to catch reflected light (Fig. 3, top center and top right).

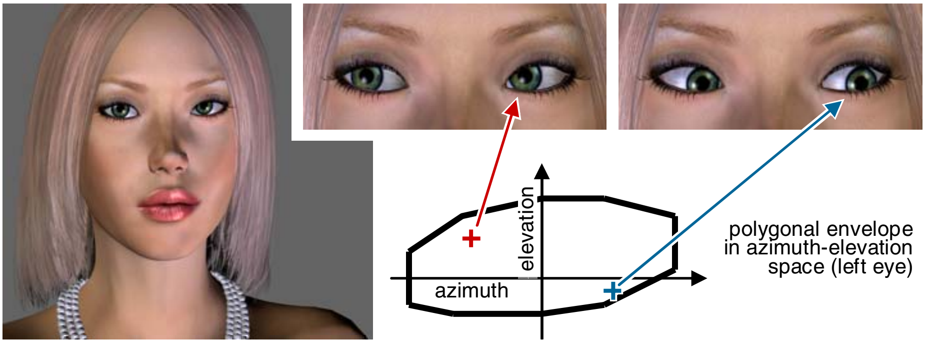


Fig. 3. Near-photorealistic human model with orientable eyes and eyelid occlusion envelope. Left and top images are real-time screen captures from the EEC system. Note highly detailed eyes (model by Handspan Studios).

To minimize the display's accommodation-convergence conflict, the most important components of the VH, its eyes, are positioned roughly at the LCD panel surface (Fig. 1b), with nose and chin protruding slightly in front of the display. The face looks straight ahead, while the rotating eyes seek out the head-tracked user's eyes. They can alternately target the user's left and right eyes, a hallmark of bad acting according to Sir Michael Caine [14], or they can fixate one eye (recommended by Caine).

Eye Contact Targets. In the current EEC implementation, the targets for the VH's eyes are the user's eye centers; but in real life, people look at irises, not eye centers. Without a pupil tracker, the EEC system cannot know which way the user's pupils point, so it cannot target them. Fig. 4 shows that only one eye from each person (**Lu** and **Rv** in Fig. 4) looks directly into one eye of the other person; these two "principal" eyes have their centers and pupils aligned on a single line, which is already guaranteed in the EEC system even though it uses eye centers as targets. Hence the VH's principal eye **Rv** requires no correction. Its other, "non-principal" eye **Lv** is slightly mis-oriented; it points to the center of the human's principal eye **Lu** instead of to the pupil (whereas the human's non-principal eye **Ru** fixates **Rv**'s pupil, not its center). Humans have extraordinary acuity for assessing gaze direction [15], and in an extreme close-up situation—probably closer than the current prototype permits—a perceptive user might notice that the VH converges about 10mm behind the user's pupil. Then again, it may not be possible to detect this slight error since in that situation the user is not directly looking at the slightly misaligned non-principal eye **Lv** but at **Rv**, the correctly oriented principal one. In other words, **Lv** does not appear in the center of either **Lu**'s or **Ru**'s fields of view. Still, **Lv** could be easily corrected if the EEC system were able to determine which of the VH's eyes is in fact the principal one at a given moment. However, measuring this dynamically changing condition requires pupil tracking.

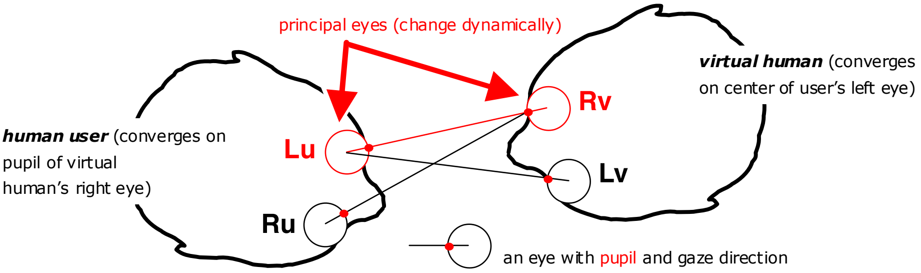

principal eyes (change dynamically)
virtual human (converges on center of user's left eye)
human user (converges on pupil of virtual human's right eye)
Rv
Lu
Lv
Ru
an eye with pupil and gaze direction

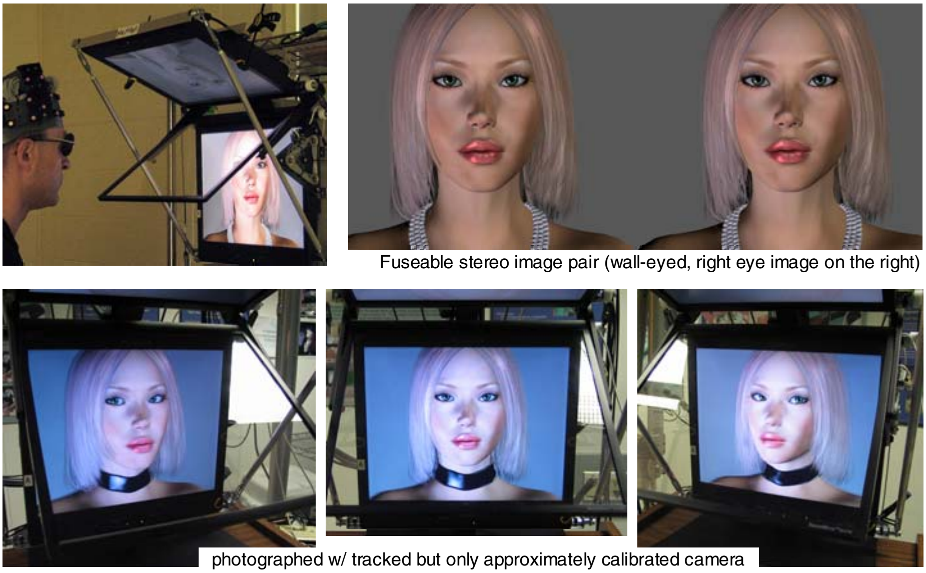
Fig. 4. A closer look at the geometry of eye contact in the EEC system

Dynamic Behavior. As described so far, the EEC SVHE system computes fresh eye gaze targets and associated azimuth and elevation angles for its VH's eyes for each new stereo frame pair presented to the head-tracked user. This overly simplistic approach results in rather bizarre, neurotic-seeming behavior. For example, as long as the gaze directions towards the user's eyes fall within the VH's occlusion envelopes, the VH will continually and instantaneously track the user with unnatural precision; or, if the user dwells at the envelope margins, the VH's eyes will oscillate, constantly alternating between looking straight ahead and making eye contact with the user. These issues were eliminated by temporally filtering the azimuth and elevation angles over the most recent *n* frames (currently *n*=12, a fifth of a second at 60Hz). A simple box filter already yields much improvement: the VH's eyes are stable and follow the user with some lag as he moves about. The lag also creates opportunities to engage in and disengage from eye contact, which more closely approximates human behavior.

2.3 Initial User Experience

The EEC prototype described here is a very simple proof-of-concept system limited to gaze interaction with a VH. A formal study has not been conducted or designed yet, but the EEC system was demonstrated to several experts in the field, as well as to several non-expert acquaintances of the author. All judged the illusion to be quite convincing and the eye contact to be realistic (though none felt *intimidated*). Most were able to tell when the VH was switching targets from one eye to another. One expert in graphics and vision declared that until experiencing the EEC system, he never thought much of stereo displays in general. All testers had their eye centers calibrated as described in Section 2.1 above; for some, an additional sanity check was performed and the distance between their eye centers, as resulting from calibration, was found to be within no more than 1-2mm of their infinity IPD, measured with a pupillometer.

The author also implemented a simple additional interaction in the EEC system: the Certus calibration stylus can act as a light source for the VH, which is programmed to fixate it if the stylus is in range (otherwise it looks for the user as already



Fuseable stereo image pair (wall-eyed, right eye image on the right)

photographed w/ tracked but only approximately calibrated camera

Fig. 5. As the user moves about the display, the virtual human acquires and attempts to maintain eye contact. The bottom set was photographed through the stereo mirror and shows both eyes' views superimposed (the mirror reflects the right eye view from the top LCD monitor).

described). Another simple feature, a software switch that toggles head tracking, turns the system into a simple but compelling educational tool that lets even naïve users instantly comprehend the difference between stereoscopy with and without head tracking.

A video segment accompanies this paper. It shows the EEC system in use, but it obviously cannot convey the live SVHE's sense of presence (Fig. 5). The video can be accessed at `http://www.cs.unc.edu/~us/EEC/`.

3 Conclusions

The EEC method described here is simple and easy to implement. It can replace camera-based pupil tracking where gaze direction knowledge is not required. After a brief initial calibration step, the system accurately tracks a user's eyes in a virtual environment with only real-time head pose tracking as input, which can be obtained from any type of tracker, including a non-intrusive, completely untethered computer vision system. As for the display, a much cheaper CRT-based stereo display with active shutter glasses could replace the Planar while still providing high image quality.

The author attributes the positive initial feedback to the extreme accuracy of the optoelectronic tracker, the precise eye calibration, and the high-quality, near photorealistic stereoscopic rendering presented at high frame rates.

The EEC system also confirms that eye centers are suitable projection origins and extends the eye center approximation to small-scale desktop displays; it further shows that eye centers are appropriate gaze targets for eye contact simulations.

4 Future Work

Prismatic distortions introduced by users' eyeglasses move the eye centers calculated by the calibration. Nearsighted users' eye centers move forward, farsighted users' backward. Wearing eyeglasses during both calibration and live interaction compensates partially, but the nonlinear nature of the distortion requires further investigation.

The virtual human's appearance leaves vast room for enhancements. To mention only the eye area, anatomical accuracy could be improved through blinking, upper and lower eyelid motion, as well as pupil enlargement and contraction, all of which could be easily synthesized in real time on a modern computer, for example by means of OpenGL vertex shaders.

An untethered camera-based head tracker would greatly improve user comfort. While head pose tracking alone already provides believable eye contact, an additional pupil tracker would enable correction of the remaining gaze direction error (in the non-principal eye, see 2.2), as well as lead to much additional experimentation. Specifically, with the ability to measure user's gaze behavior, a controlled user study becomes feasible, even though its design would be quite challenging in the author's opinion. Furthermore, the integrated pupil tracker could open the door to interdisciplinary research, for example in the realm of behavioral psychology.

Acknowledgments. The author thanks the testers for their insights and suggestions. They include Anna Bulysheva, Charles Burke (MD), Henry Fuchs, Martina Gargard, Bil Hays, Kurtis Keller, Sharif Razzaque, Herman Towles, Greg Welch, and Hua Yang. Anna Bulysheva and Herman Towles reviewed draft versions of the paper. Tabitha Peck prepared the initial version of the GLUT-based framework used by the EEC prototype. Hua Yang prepared interface software for the Certus. The EEC real-time renderer uses libwave by Dave Pape, ported to Windows with help from Hua Yang. Partial funding for this work was provided by the National Institutes of Health (1 R01 CA101186-01A2).

References

1. Takacs, B., Kiss, B.: The virtual human interface: A photorealistic digital human. IEEE Computer Graphics and Applications 23(5), 38–45 (2003)
2. Badler, N.I., Phillips, C.B., Webber, B.L.: Simulating Humans: Computer Graphics, Animation, and Control. Oxford Univ. Press (1993)
3. Raij, A.B., Johnsen, K., Dickerson, R.F., Lok, B.C., Cohen, M.S., Duerson, M., Pauly, R.R., Stevens, A.O., Wagner, P., Lind, D.S.: Comparing Interpersonal Interactions with a Virtual Human to Those with a Real Human. IEEE Transactions on Visualization and Computer Graphics 13(3), 443–457 (2007)
4. Argyle, M., Cook, M.: Gaze and Mutual Gaze. Cambridge University Press, Cambridge (1976)
5. Meehan, M., Razzaque, S., Whitton, M., Brooks, F.: Effects of Latency on Presence in Stressful Virtual Environments. In: Proceedings of IEEE Virtual Reality 2003, pp. 141–148. IEEE Computer Society Press, Los Alamitos (2003)
6. Deering, M.: High Resolution Virtual Reality. In: Thomas, J.J. (ed.) SIGGRAPH 1992. Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, vol. 26(2), pp. 195–202. ACM Press, New York (1992)
7. Takayama, A., Sugimoto, Y., Okuie, A., Suzuki, T., Kato, K.: Virtual human with regard to physical contact and eye contact. In: Kishino, F., Kitamura, Y., Kato, H., Nagata, N. (eds.) ICEC 2005. LNCS, vol. 3711, pp. 268–278. Springer, Heidelberg (2005)
8. Billinghurst, M., Henrysson, A.: Research Directions in Handheld AR. Int. J. of Virtual Reality 5(2), 51–58 (2006)
9. Rolland, J.P., Burbeck, C.A., Gibson, W., Ariely, D.: Towards Quantifying Depth and Size Perception in 3D Virtual Environments. Presence: Teleoperators and Virtual Environments 4(1), 24–48 (1995)
10. Holloway, R.: Registration Error Analysis for Augmented Reality. Presence: Teleoperators and Virtual Environments 6(4), 413–432 (1997)
11. Lipton, L.: Foundations of the Stereoscopic Cinema. Van Nostrand Reinhold (1982)
12. Azuma, R., Bishop, G.: Improving Static and Dynamic Registration in an Optical See-Through HMD. In: Proceedings of SIGGRAPH 1994, Computer Graphics, Annual Conference Series, 1994, pp. 197–204 (1994)
13. Fuhrmann, A., Splechtna, R., Pikryl, J.: Comprehensive calibration and registration procedures for augmented reality. In: Proc. Eurographics Workshop on Virtual Envi-ronments 2001, pp. 219–228 (2001)
14. Michael Caine on Acting in Film. TV miniseries episode produced by BBC Entertainment and Dramatis Personae Ltd. (1987)
15. Symons, L.A., Lee, K., Cedrone, C.C., Nishimura, M.: What are you looking at? Acuity for triadic eye gaze. J. Gen. Psychology 131(4), 451–469 (2004)

Real Time Body Pose Tracking in an Immersive Training Environment

Chi-Wei Chu and Ramakant Nevatia

Institute for Robotics and Intelligent System
University of Southern California
Los Angeles, CA 90089-0273
{chuc,nevatia}@usc.edu

Abstract. We describe a visual communication application for a dark, theater-like interactive virtual simulation training environment. Our system visually estimates and tracks the body position, orientation and the arm-pointing direction of the trainee. This system uses a near-IR camera array to capture images of the trainee from different angles in the dim-lighted theater. Image features like silhouettes and intermediate silhouette body axis points are then segmented and extracted from image backgrounds. 3D body shape information such as 3D body skeleton points and visual hulls can be reconstructed from these 2D features in multiple calibrated images. We proposed a particle-filtering based method that fits an articulated body model to the observed image features. Currently we focus on the arm-pointing gesture of either limb. From the fitted articulated model we can derive the position on the screen the user is pointing to. We use current graphic hardware to accelerate the processing speed so the system is able to work in real-time. The system serves as part of multi-modal user-input device in the interactive simulation.

1 Introduction

Training humans for demanding task in a simulated environment is becoming of increasing importance, not only to save costs but also to reduce training risk for hazardous tasks. A key issue then becomes the modalities by which the human trainee needs to communicate with the characters in the synthetic environments. Speech is one natural modality but visual communications, such as gestures and facial expressions, are also important for a seamless human-computer interaction (HCI) interface. Our objective is to achieve such communication, coupled with other modalities such as speech in the longer term. Here, we describe the first steps of body position and pose tracking that are essential to both forms of communication.

A synthetic training environment must be immersive to be effective. Instead of wearing head mounted displays, the user is positioned on a stage facing a large screen that display a 3D rendered virtual environment, such as a war-zone city street or a field hospital. This environment makes visual sensing very challenging. The environment is very dark and the illumination fluctuates rapidly as the scenes on the screen change. The sensing system must be passive and not interfere with communication or the displays.The trainee can walk around in a limited area for natural responses and does not

M. Lew et al. (Eds.): HCI 2007, LNCS 4796, pp. 146–156, 2007.

have to wear special clothing with markers or other sensors (e.g. magnetic sensors). The system must be able to operate in real-time to be responsive as a input channel. We consider presence of only one trainee at a time in the current phase of our work. Two examples of our environments and the floor plane of one of the environments, the virtual reality theater, are shown in Fig.1.

We solve the illumination difficulty by illuminating the scene with infrared (IR) lights and imaging it with normal cameras with near-IR filter lenses. While this allows images to be acquired in the dark, flickering environment, the images are rather featureless with little texture or interior detail visible. We use multiple cameras to partially compensate for these deficiencies and to make acquisition of 3-D position and pose easier. The body pose tracking uses a particle filtering method. Our method uses a bottom-up approach that guides the configuration of models. Our method uses a bottom-up approach that guides the fitting of the configuration of models by lower level visual cues. We describe these methods in detail, later in the paper.

At present, we only estimate the location of the user on the stage and the direction of the arm in 3D space when the user points at a certain entity on the screen. This arm direction can then be integrated into a higher level of user interface system to estimate the screen coordinate the arm pointing direction.

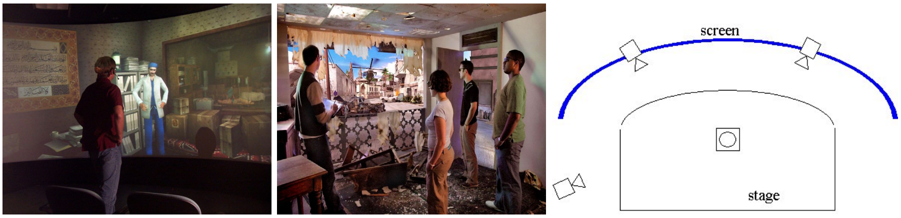


Fig. 1. Virtual Training systems and Flood Plane Example

1.1 Related Work

Various methods have been proposed for the estimation and tracking of of human body postures. Many approaches recognize the postures directly from 2D images, as single camera image sequences are easier to acquire. Some of them try to fit body models (3D or 2D) into single-view images [12] [8][15], or classify postures by image features [7,16]. There are two main difficulties those methods must overcome. One is the loss of depth information. Thus it's difficult to tell if the body parts are orienting toward to or away from the camera. These system must either maintain both hypotheses or regulate the estimate with human body kinetic constraints. The other problem comes from self-occlusion: body parts that are not seen from the image cannot be estimated. Roughly one third of the degrees of freedom of the human model are unobservable due to motion ambiguities and self-occlusions.

To compensate for these ambiguities due to 2D acquisition, several approaches rely on using multiple cameras. These approaches use an array of two or more cameras or 3D body scanners to capture the human shape and motion from different views. Some of these approaches extract 2D image features from each camera and use these features to search for, or update the configuration of, a 3D body model [9][4] [18]. Others introduce an intermediate step of reconstructing the 3D shape of human body. The characterization of the human pose is then done by fitting a kinematics model into the 3D shape information [10] [3].

Particle Filtering (PF) has become increasingly popular for tracking body poses in recent work. Bernieret al. [2] combine Particle Filtering and Markov Network belief propagation to track 3D articulated pose in real-time, using stereo depth images. In [17], Schmidt et al. use Kernel Particle Filter and mean-shift methods to avoid large number of particles using intensity and color images cues, while also achieving near real-time performance.

Many of the methods above make an implicit assumption that the image quality is stable and high enough to extract useful image features. They utilize features such as motion flow, color and texture of skin and clothing to improve the classification or model-fitting accuracy. Such features are not available in our environment.

1.2 Outline of Our Approach

Due to environment limits, we have to illuminate the room with IR lights. The only useful feature in the IR images is the silhouette. We first segment the silhouettes of the user from the grey-scale IR images. 2D symmetry axis points are then extracted from the silhouettes. These axis points can be used to estimate the location and direction of different body parts such as arms and torso in 2D images. With 2D silhouettes and axis points from different calibrated cameras, we can construct 3D visual hulls and 3D body axis points.

The body of the user is modeled as an articulated model with 7 degrees of freedom (DOF) and ellipsoids as body segment shapes. We use a Particle Filter to fit the body model to these image feature. Sampling of new particles is based on the 3D shape and axis information to create particles that are more likely to match the observation. Without the guidance of the axis points, the filter can only randomly spawn new particles by brute force and thus wastes computation resources. The silhouettes are matched against the estimated body pose, resulting in observation probability densities. With the aid of current graphical hardware, this system can run in real-time and provide reliable pose estimation.

2 Environment and Sensor System Design

In our environment(s), we set up an array of four or more synchronized cameras around the stage. Each cameras is equipped with an IR lens filter. This eliminates the effects of flickering light and allows the cameras to sense people in the dim environment. The images captured in near-IR spectrum are bare of most of the common image features seen in a visible spectrum image, such as color and texture. The only usable feature is the pixel blob silhouette segmented from the background by motion, as shown in Fig.2. Even this pixel blob segmentation is noisy, given the grainy and washed-out property of IR images. One of the reasons to use multiple cameras, in addition to acquiring 3D shape information, is to gather more usable information to compensate for the poor quality of the images. We place two cameras in the front of the user, either above or below the screen; another at one side and one overhead camera on the ceiling. This way we can best capture the shape of the user with minimum number of cameras. Additional cameras may be included to provide more information. All cameras are calibrated referencing to a common global coordinate system. IR spot lights are placed around the stage and directed to the screen or walls, so they can reflect and diffuse the lights.

3 Feature Extraction

We model the intensity value of each pixel in the image as a Gaussian distribution, which is learnt during a background training phase. Each pixel of new incoming images is compared against its corresponding distribution. If the difference is larger than certain variance threshold from the mean pixel value, the pixel is classified as the foreground object pixel. Pixel blobs of sizes too small are considered as noise and are removed.

This method, however, segments the shadows of the user as well as his/her actual body. To effectively distinguish the shadows from user, we diffuse the IR lights when setting up the lighting condition. This can be achieved by reflecting IR lights by diffuse materials such as the screen, instead of directly illuminate the user with IR spot-lights. Thus the shadows cast by the IR lighting have blurry boundaries and edge properties can be incorporated to eliminate shadow regions.

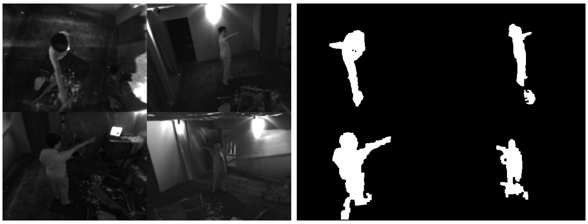


Fig. 2. Example IR images and Background Segmentation

3.1 2D Body Symmetry Axis Points

Next, we want to find the symmetry axes of the 2D silhouettes to enable segmentation of the blob into body parts; the 2D axis points and their 3D reconstructions are then used to guide the fitting of an articulated body model. Several techniques exist to find the skeleton axis points, such as medial axis points [13] or generalized cylinders [19]. Here we scan the silhouettes with arrays of scan lines of four different directions (horizontal, vertical and two diagonal directions) and intersect the scan-lines with the silhouettes.The middle point of each segment is extracted as an axis point. The line segment length information is also stored as the ”diameter” of the axis point. An example is shown in Fig.3 below. Axis points of adjacent scan-lines are considered ”neighbors” and a body axis graph is inferred from the neighborhood information.

Given our task of estimating arm pointing directions, we want to find the possible arm segments in the body axis graph. Given a fixed camera, the projected images of the user’s arms are restricted to certain regions of the images. Also the images of the arms have smaller diameter (thinner) than other body parts. Thus, we select segments of graph that lie in predefined image regions and have diameter below certain threshold.

The torso of the user should remain relatively up-right in normal simulation activities and image of the torso should be the central bulk of the pixel blobs. Thus we select axis points scanned by horizontal scan-lines and have scan diameter above the threshold. Figure 4 illustrates an example of arm and torso segmentation.

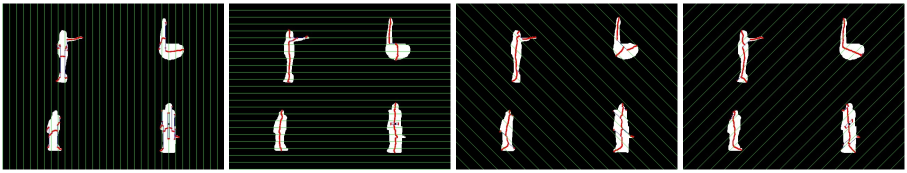


Fig. 3. Silhouettes Scan-Lines and 2D Axis Points

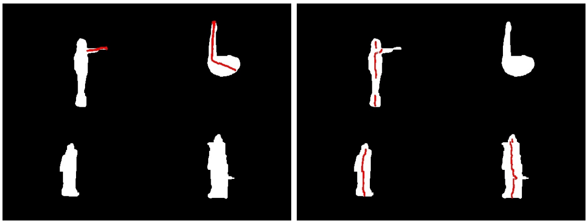


Fig. 4. Arm and Torso 2D Axis Points

3.2 3D Body Axis Points and Visual Hull

Given a set of 2D silhouette images of human body from different angles, we can compute the polyhedral approximation of the 3D shape (Fig.5 Left) [11,14]. This method is fast and allows us to achieve real-time reconstruction. The 3D blob shape is used as a *proposal function* to estimate the floor position of the person in pose tracking described later.

For the arms and torso, their 3D body axis points can also be constructed from the 2D counterparts (Fig.5 Right). Assume we have two 2D axis graphs, S_i and S_j, of two different images, I_i, I_j. For each point P_{ik} in S_i, we compute the epipolar line L_{ikj} that map from P_{ik} to the I_j. Let point P_{ikj} be the the intersection point of L_{ikj} and the edges of $S_{j.}$. We can reconstruct 3D points from P_{ik} and P_{ikj}, given the calibration information of the two cameras. Both 2D silhouette information and 3D shape information can be used to estimate 3D body pose configurations.

4 Pose Tracking

Given our task of finding the arm pointing direction of the user, we model only four segments of the articulated model - body (torso and legs), head and two arms. 7 degrees

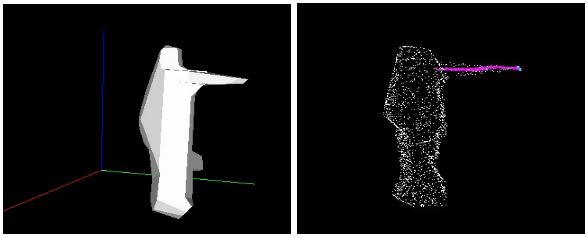


Fig. 5. Visual Hull and 3D Arm Axis Points

of freedom (DOF) of the body are estimated - positions on the stage floor, orientation of the body and the rotations of each shoulder joint. The shapes of body segments are modeled as one or more 3D ellipsoid,as shown in Fig.6 below.

4.1 Particle Filtering

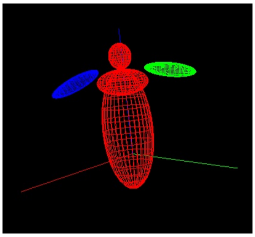


Fig. 6. Articulated Body Model

Assume x_t is the joint angles of the body model and y_t is the captured image information at time t. We want to estimate the posterior probabilities $p\left(x_t \mid y_{1:t}\right)$ given the sequence of images $y_{1:t}$, to find the most likely x_t. However, it's difficult to derive the probability function directly. One reason is at the dimensionality of x_t is high (7 DOF plus their derivatives). And another reason is that the camera projection from x_t to y_t is not a linear process. One way that is often used to estimate the posterior probability is the *Sequential Monte Carlo method*, also known as *Particle Filtering* [1,5]. Instead of deriving the actual analytical form of the posterior, Particle Filter estimates the distribution with set of data samples called particles. Each particle represents one possible user state (position and joint angles).

We also make an assumption that our system is *Markovian Model*. i.e. the state x_t depends on only the state x_{t-1} of previous time instance. And that the captured images y_t only result from the current user state x_t.

$$p\left(x_t \mid x_{0:t-1}\right) = p\left(x_t \mid x_{t-1}\right) \tag{1}$$

$$p\left(y_t \mid x_{0:t}\right) = p\left(y_t \mid x_t\right) \tag{2}$$

The Particle Filter works as follows:

- **Initialization:** at beginning, N data samples $\alpha_0^i, i = 1...N$ are drawn from a prior distribution $p\left(x_0\right)$
- **Weighted sampling:** at each time step t, for each particles α_{t-1}^i at time $t-1$, sample a new particle according to the density distribution of the proposal function:

$$\widetilde{\alpha}_t^i \sim q\left(\alpha_t^i \mid y_{1:t}, \alpha_{0:t-1}^i\right) = q\left(\alpha_t^i \mid y_t, \alpha_{t-1}^i\right) \tag{3}$$

Evaluate the weight w_t^i of each particle α_t^i by

$$w_t^i = \frac{p\left(y_t \mid \widetilde{\alpha}_t^i\right) p\left(\widetilde{\alpha}_t^i \mid \alpha_{t-1}^i\right)}{q\left(\widetilde{\alpha}_t^i \mid y_t, \alpha_{t-1}^i\right)} \tag{4}$$

and all w_t^i, $i = 1...N$ are normalized to the sum of one.

- **Resampling:** sample N new particles α_t , from the set $\left\{\widetilde{\alpha}_t^1 ... \widetilde{\alpha}_t^N\right\}$ with replacements, according to the probability:

$$p\left(\alpha_t^i = \widetilde{\alpha}_t^i\right) = w_t^i \tag{5}$$

Thus at the end of each iteration (time frame), the system would consists N samples representing N different possible body pose configurations.

4.2 Observation Probability

The observation probability $p\left(y_t \mid \alpha_t^i\right)$ is evaluated by projecting the shape model of particle α_t^i into the each 2D image. Assume P_c^i is the projected image of α_t^i projected into camera $c, c = 1 \ldots C$. and S_c is the silhouette in image c. Then the probability is estimated as following:

$$p\left(y_t \mid \alpha_t^i\right) = \prod_{c=1}^{C} \frac{\|P_c^i \cap S_c\|}{\|P_c^i \cup S_c\|} \tag{6}$$

where $\|\cdot\|$ is the size of image region in pixels.

4.3 Proposal Functions

The proposal function $q\left(\alpha_t^i \mid y_{1:t}, \alpha_{t-1}^i\right)$ is an important part of the Particle Filter method. Theoretically, if the proposal function is defined as the posterior:

$$q\left(\alpha_t^i \mid y_t, \alpha_{t-1}^i\right) = p\left(\alpha_t^i \mid y_t, \alpha_{t-1}^i\right) \tag{7}$$

it will minimize the estimation error. But it's usually not possible to sample directly from the posterior distribution. We define a set of different proposal functions, $Q = \{q_i\}$. When drawing new samples in the weighted sampling step, proposal functions are randomly selected from this set and samples are drawn according to their probability distribution. The probability of each function being selected are set empirically. The proposal functions we uses are:

Random Walk function: The proposal function is simply defined as:

$$q\left(\alpha_t^i \mid y_t, \alpha_{t-1}^i\right) = p\left(\alpha_t^i \mid \alpha_{t-1}^i\right) \tag{8}$$

New observation is not taken into account when spawning new samples. The system models not only the current value of each degree of freedom, but also their first and second order time derivatives (velocity and acceleration). Gaussian noises are added to the new value and its derivatives.

Body Position Function: This function estimates the new body position. It randomly samples points from 3D torso axis points as new body position on the stage. Another way is to compute the center of mass of the visual hull as the new body position.

Arm Direction Function: The 3D arm axis points are used to estimate the shoulder joint angles in the new samples. Assume for particle α_t^i, $\boldsymbol{m}$ is the vector pointing from the shoulder joint position s_t^i to the end of the arm, normalized to unit length. Then for each 3D arm axis point p_j, we compute the vector $\boldsymbol{p}_j = p_j - s_t^i$. One of the 3D axis points is then randomly selected according to the value of $\boldsymbol{m} \cdot \boldsymbol{p}_j$. And the shoulder joint angles are alter to make $\boldsymbol{m}'$ aligns with $\boldsymbol{p}$. This way, 3D axis points that are of opposite direction (possibly belong to the other arm) would not be selected.

However, there may be no 3D axis or shape information available either because the user is not performing pointing gesture at all, the extended arm is occluded by body due to viewing angle limit, or there is severe error in pixel segmentation. Then the body position or arm direction methods would have no effect and make no modifications to the corresponding DOFs.

By using proposal functions, our method is able to utilize a bottom-up approach that uses visual cues to guide the transition of particles. In the future, other different proposal functions that take advantages of different image features can be introduced into this framework to expand the capability of the system.

The Particle Filter not only provides the advantage of estimating non-linear posterior probability. It also increases the robustness of the tracking system. Our silhouettes observation is noisy and unstable given the environment constraints. The Particle Filter maintains multiple hypotheses of the current user state and thus is able to keep tracking user pose in the absence of usable image features, and quickly recover from erroneous state when correct silhouettes are available. In our experiments, the segmentation of the arm may occasionally fail and causes the model to swing wildly, but it always recovers back after only 2 to 3 frames.

The system also does not need manual initialization. If the system finds that the observation densities of the particles are all near zero, it assumes that the stage is empty and does not output model configurations. When a user enters the stage from any location, the proposal functions quickly converges the particles around the user and automatically start tracking.

4.4 Improving Computational Performance

As our system is intended to be used as an on-line human-computer interaction interface, it must be able to operate in real-time. To improve computational efficiency, we use an approximation for the body part shapes and also use the Graphical Processing Unit (GPU) available in a normal desktop computer.

The most computationally intensive part of the system is the inference of the observation probability $p\left(y_t \mid \alpha_t^i\right)$. It needs to project (render) the body shape model into each camera image, for each particle and compare with the silhouettes. To render the body model consisting of 3D ellipsoids, we directly compute the analytical form of the projected ellipses[6]. An ellipsoid in 3D space can be defined as

$$\boldsymbol{x}^t A \boldsymbol{x} + \boldsymbol{b} \cdot \boldsymbol{x} + c = 0 \tag{9}$$

Then the projected ellipse can be defined as

$$(\boldsymbol{x} - \boldsymbol{e})^t M (\boldsymbol{x} - \boldsymbol{e}) \geq 0 \tag{10}$$

where $\boldsymbol{e}$ is the 3D coordinate of the camera optical center, $\boldsymbol{x}$ is the coordinate of the point in the 3D image plane. And

$$M = (\boldsymbol{b} + 2A\boldsymbol{e})(\boldsymbol{b} + 2A\boldsymbol{e})^t - 4\left(\boldsymbol{e}^t A \boldsymbol{e} + \boldsymbol{b} \cdot \boldsymbol{e} + c\right) A \tag{11}$$

We can derive the pixels inside the ellipses directly from the analytical form, instead of performing perspective projection of a polyhedral approximation. Silhouette pixels are compared against those ellipse equations instead of rendered body images. Since such computation is highly parallel in nature, we utilize the GPU hardware to compute the ellipsoid mapping and the pixel evaluation. This gains several magnitude of order of performance increase (See results section).

4.5 Integration

In each time frame, the particle with the highest weight in the the sampling step is selected and the articulated body configuration represented by this particle is used to estimate the pointing direction. The vector that originates from the estimated 3D coordinate of the eyes to the tip of the arms are used as pointing directions. Currently the eye coordinate is defined as the center of 3D head shape. In future, face tracking module may be integrated to provide more accurate eye position estimation. Intersecting this vector with screen results the estimated screen coordination the user is pointing at.

The visual body tracking system interconnects with the virtual environment training system via a network. The current pointing direction in 3D space is constantly updated. The VR system then transforms the real-world 3D space coordinate into virtual-world coordinate and recognizes the virtual entity the user is pointing at.

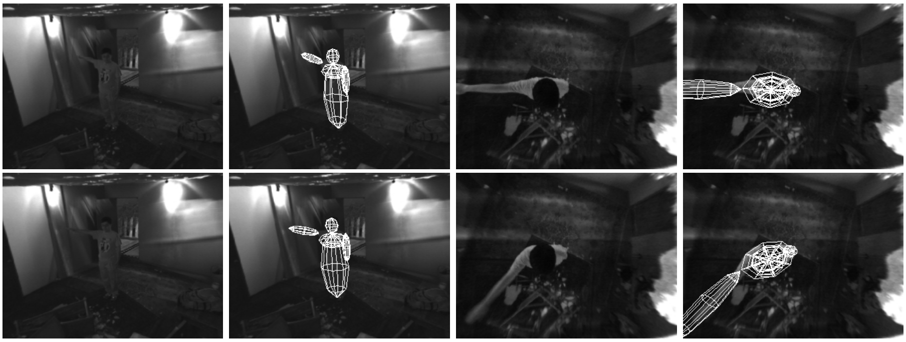


Fig. 7. Example Tracking Sequences

5 Experiment Results

We have set up the system in two different environments (see Fig.1). One is used in a virtual-reality theater with a curved surrounding screen, the other is in a set simulating a war-torn building with projection screens as walls and doors. We use four to five synchronized IR firewire cameras connected to a 3.2 GHZ Dual-Xeon workstation; the GPU is an nVidia Geforce 8800. The system is able to run in 9 frame per second on average with 150 particles; this speed is acceptable for simple interactions with the environment. Without the use of GPU, the speed is 3-4 seconds per frame. More particles provide better estimation robustness and accuracy but slow down the performance. Some frames from a tracking sequence are shown in Fig.7.

To test the accuracy quantitatively, we collected video sequences of users interacting in the system with principal action being that of pointing. We recorded video of 5 different people, each of length 200-300 frames. We manually configured the articulated model to fit the video data (visually) frame by frame, to create the ”ground truth” target. This approach is time-consuming and the amount of target we could acquire is limited. More data would be readily available when this modal is full integrated into a larger system and we can receive the pointing target feedback from the module. The accuracy is estimated by the angle between the target and estimated pointing vectors. The average error in angle is 8.11 degrees. We expect that this pointing accuracy is satisfactory for many interaction tasks where the virtual entities of interest are not very dense though integrated experiments need to be performed to confirm this.

Jitter, representing rapid fluctuations of the estimated pose, can occur because our proposal functions do not take the motion smoothness into account when spawning new particles. We could gain some smoothness by backward filtering but this would introduce additional latency into the system; we have not explored the trade-off between the two for a user. Currently our system works with a single user only. Segmentation of multiple users is made more difficult in IR images due to lack of color or texture. We intend to explore this in our future work.

6 Conclusion and Future Work

We described a near real-time body pose estimation and tracking system for a user in a poorly illuminated environment by use of IR lights. Use of axis finding methods provides a rough part segmentation and hypotheses for part orientations; use of particle filtering provides robust estimation of the pose from noisy sequences.

In the future, we plan to integrate a face tracking system into the body posture estimation system. The body tracking system will provide the face tracking system an estimated location of the face so it would not need to scan the whole image for faces. The face tracker will also provide improved eye position estimated for better deriving arm-pointing direction. We then plan to integrate the visual modality into a larger system and evaluate its effectiveness in the training task.

Acknowledgment

The research in this paper was sponsored, in part, by the U.S. Army Research, Development and Engineering Command (RDECOM). The content of the information does not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred

References

1. Arulampalam, S., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. IEEE Transactions on Signal Processing 50(2), 174–188 (2002)
2. Bernier, O., Cheung Mon, P.: Real-time 3d articulated pose tracking using particle filtering and belief propagation on factor graphs. In: BMVC 2006, p. 7 (2006)

3. Demirdjian, D., Darrell, T.: 3-d articulated pose tracking for untethered diectic reference. In: Proceedings of ICMI 2002, Washington, DC, USA, p. 267. IEEE Computer Society Press, Los Alamitos (2002)
4. Deutscher, J., Blake, A., Reid, I.D.: Articulated body motion capture by annealed particle filtering. In: CVPR 2000, pp. 126–133 (2000)
5. Doucet, A., de Freitas, N., Gordon, N.: Sequential Monte Carlo Methods in Practice. Springer, Heidelberg
6. Eberly, D.: Perspective projection of an ellipsoid
7. Elgammal, A., Lee, C.-S.: Inferring 3d body pose from silhouettes using activity manifold learning (2004)
8. Horain, P., Bomb, M.: 3d model based gesture acquisition using a single camera. In: WACV 2002. Proceedings. Sixth IEEE Workshop, pp. 158–162. IEEE, Los Alamitos (2002)
9. Izo, T., Grimson, W.: Simultaneous pose estimation and camera calibration from multiple views. In: Non-Rigid 2004, p. 14 (2004)
10. Kehl, R., Bray, M., Van Gool, L.: Full body tracking from multiple views using stochastic sampling. In: CVPR 2005, vol. 2, pp. 129–136. IEEE Computer Society Conference, Los Alamitos (2005)
11. Laurentini, A.: The visual hull concept for silhouette-based image understanding. PAMI 16(2), 150–162 (1994)
12. Lee, M., Nevatia, R.: Dynamic human pose estimation using markov chain monte carlo approach. In: Motion 2005, pp. 168–175 (2005)
13. Leymarie, F.F., Kimia, B.B.: Medial Representations: Mathematics, Algorithms and Applications. Kluwer Academic Publishers, Dordrecht, ch. 11, (to be Published)
14. Matusik, W., Buehler, C., McMillan, L.: Polyhedral visual hulls for real-time rendering. In: Proceedings of the 12th Eurographics Workshop on Rendering Techniques, London, UK, pp. 115–126. Springer, Heidelberg (2001)
15. Parameswaran, V., Chellappa, R.: View independent human body pose estimation from a single perspective image. In: CVPR 2004, pp. 16–22 (2004)
16. Rahman, M., Ishikawa, S.: Appearance-based representation and recognition of human motions. In: ICRA 2003. Proceedings of IEEE International Conference on Robotics and Automation, 2003, vol. 1, pp. 1410–1415. IEEE Computer Society Press, Los Alamitos (2003)
17. Schmidt, J., Fritsch, J., Kwolek, B.: Kernel particle filter for real-time 3d body tracking in monocular color images. In: FGR 2006, pp. 567–572 (2006)
18. Yoshimoto, H., Date, N., Yonemoto, S.: Vision-based real-time motion capture system using multiple cameras. In: MFI, pp. 247–251 (2003)
19. Zerroug, M., Nevatia, R.: Segmentation and 3-d recovery of curved-axis generalized cylinders from an intensity image. In: Computer Vision and Image Processing, Proceedings of the 12th IAPR International Conference, vol. 1, pp. 678–681 (1994)

Author Index