Real-time Vision-based Hand and Face Tracking and Recognition of Gesture

A PhD dissertation submitted in partial fulfillment of the requirement for the degree of

Doctor of Philosophy (Ph.D.)

in

Computer Science

by

Farhad Dadgostar

Institute of Information and Mathematical Sciences

College of Science

Massey University

December 2006

Real-time Vision-Based Hand and Face Tracking and Recognition of Gesture

Dedication

With love, to the one who always encouraged me and provided me indefinite unconditional support…

To my wife Nasim

Acknowledgements

I would like to thank Dr. Abdolhossein Sarrafzadeh, my supervisor, for all of his support, his patience, and his guidance and endless encouragement throughout my PhD study. Without his support, I would have never gained such an interest in research and nor learned as much as I have. I have been lucky and am proud of being a student of him. Working with him was a great pleasure.

I would like to thank to Dr. Scott P. Overmyer and Dr. Liyanage De Silva for their valuable suggestions and comments on my work.

I also have enjoyed working with a number of fellow PhD colleagues over the past years. In particular Andre Barczak and Chao Fan with whom sharing the authorship of several papers and reports reflect the closeness of our cooperation.

The financial support provided by the Massey University and the Institute of Information and Mathematical Sciences made my research and presenting the results in international conferences possible. In particular, I would like to thank Professor Robert McKibbin, the head of the institute for his support during my study.

This achievement would not have been possible without endless support provided to me by my family. I would like to thank my parents who have always been supportive and encouraging through all the time of my study. I would like to thank my precious daughter Kiana for her company and patience when I was working on weekends. Finally, I would like to express my deep thanks to my lovely wife, Nasim, for all her love, sacrifice, support and encouragement which could be found in every word, in this work.

List of Publications

Journal Articles

- **Farhad Dadgostar**, and Abdolhossein Sarrafzadeh, *"An adaptive real-time skin detector based on Hue thresholding: A comparison on two motion tracking methods"*, In Pattern Recognition Letters, Vol. 27, Issue 12, pp. 1342-1352, March 2006, Elsevier.
- Abdolhossein Sarrafzadeh, Samuel Alexander, **Farhad Dadgostar**, Chao Fan, and Abbas Bigdeli, *"How do you know that I don't understand? A look at the future of intelligent tutoring systems"*, Accepted for publication in Journal of Computers in Human Behavior, 2006.

Book Chapters

- **Farhad Dadgostar**, and Abdolhossein Sarrafzadeh, *"A Fast Skin Detection Algorithm for Video Sequences"*, In Mohamed Kamel, Aurelio Campilho (Ed.), Lecture Notes in Computer Science, ICIAR 2005, Vol. 3656, pp. 804-811, 2005, ISBN 3-540-29069-9, Springer-Verlag, Berlin, Heidelberg.
- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, and Scott P. Overmyer, *"Face Tracking Using Mean-Shift Algorithm: A Fuzzy Approach for Boundary Detection"*, In J. Tao, T. Tan and R. W. Picard (Ed.), Lecture Notes in Computer Science: Affective Computing and Intelligent User Interfaces, Vol. 3784, pp. 56-63, 2005, ISBN 3-540-29621-2, Springer-Verlag, Berlin, Heidelberg.
- **Farhad Dadgostar**, Hokyoung Ryu, Abdolhossein Sarrafzadeh, and Scott P. Overmyer, *"Making sense of student use of nonverbal cues for intelligent tutoring systems"*, In the ACM International Conference Proceeding Series: 19th conference of the computer-human interaction special interest group (CHISIG), Vol. 122, pp 1-4, 2005, ISBN 1-59593-222-4, Canberra, Australia.

International Refereed Conferences

- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, Chao Fan, Liyanage De Silva, and Chris Messom, *"Modeling and Recognition of Gesture Signals in 2D Space: A comparison of NN and SVM approaches"*, The 18th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2006, Washington D.C., USA.
- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, Scott P. Overmyer, and Liyanage De Silva, *"Is the Hand really quicker than the Eye? Variances of the Mean-Shift algorithm for real-time hand and face tracking"*, IEEE International Conference on

Computational Intelligence for Modelling, Control and Automation, CIMCA 2006, Sydney, Australia.

- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, and Scott P. Overmyer, *"Genetic Algorithms and Long-Haar features: A Method for Object Detection"*, The 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications, CITSA 2006, Orlando, Florida, USA.
- Abdolhossein Sarrafzadeh, Samuel Alexander, **Farhad Dadgostar**, Chao Fan, and Abbas Bigdeli, *"See Me, Teach Me: Facial Expression and Gesture Recognition for Intelligent Tutoring Systems"* , IEEE International Conference on Innovation in Information Technology, IIT 2006, Dubai, U.A.E.
- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, Hokyoung Ryu, *"A Macro model of Human Emotional-Response for Intelligent Agents Applications"*, 1st Korea-New Zealand Joint Workshop on Advances of Computational Intelligent Methods and Applications, Feb 2006, Auckland, New Zealand.
- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, *"A Component-based Architecture for Vision-based Gesture Recognition"*, In the proceedings of the International Image and Vision Computing Conference, IVCNZ 2005, pp. 322-328, Dunedin, New Zealand.
- Chao Fan, Abdolhossein Sarrafzadeh, **Farhad Dadgostar**, and Hamid Gholamhosseini, *"Facial Expression Analysis by Support Vector Regression"*, In the proceedings of the International Image and Vision Computing Conference, IVCNZ 2005, pp. 311-317, Dunedin, New Zealand.
- **Farhad Dadgostar**, Chao Fan, and Abdolhossein Sarrafzadeh, *"A Hybrid Approach for Robust Real-time Face Tracking in Video Sequences"*, In the proceedings of the Institute of Information and Mathematical Sciences Postgraduate Conference, pp. 35-42, 2005, Auckland, New Zealand.
- Chao Fan, Abdolhossein Sarrafzadeh, **Farhad Dadgostar**, and Hamid Gholamhosseini, *"Face and Eye Detection Using Support Vector Machines"*, The 3rd International Conference on Computational Intelligence, Robotics and Autonomous Systems, CIRAS 2005, Singapore.
- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, and Martin Johnson, *"An Adaptive Skin Detector for Video Sequences Based on Optical Flow Motion Features"* , In the proceedings of the 7th International IEEE Conference on Image and Signal Processing, IASTED-SIP 2005, In M.W. Marcellin (Ed.), Hawaii, USA.
- Andre L. C. Barczak, **Farhad Dadgostar**, and Martin Johnson, *"Real-time Hand Tracking Using the Viola and Jones Method"* , In the proceedings of the 7th International IEEE Conference on Image and Signal Processing, IASTED-SIP 2005, In M.W. Marcellin (Ed.), Hawaii, USA.
- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, and Scott P. Overmyer, *"An Adaptive Real-time Skin Detector for Video Sequences"*, In the Proceedings of the 2005 International Conference on Computer Vision, Vision 2005, H. R. Arabnia (Ed.), CSREA Press 2005, ISBN 1-932415-65-3, pp. 65-70, Las Vegas, Nevada, USA.
- Andre L. C. Barczak, **Farhad Dadgostar**, and Chris Messom, *"Real-time Hand Tracking Based on Non-invariant Features"*, Proceedings of the International IEEE Conference on Measurement and Instrumentation, IMTC 2005, Vol. 3, pp 2192-2192, Ottawa, Canada.

- Chao Fan, **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, Hamid Gholamhosseini, and Martin Johnson, *"Facial Expression Reconstruction Using Polygon Approximation"*, In the proceedings of the 7th International IEEE Conference on Image and Signal Processing, IASTED-SIP 2005, In M.W. Marcellin (Ed.), Hawaii, USA.
- Abdolhossein Sarrafzadeh, Chao Fan, **Farhad Dadgostar**, Sam Alexander, and Chris Messom, *"Frown Gives Game Away: Affect Sensitive Tutoring Systems for Elementary Mathematics"*, International IEEE Conference on Systems, Man and Cybernetics, SMC 2004, The Hague, The Netherlands.

Other Publications

- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, *Gesture recognition through angle space*, Research Letters in the Information and Mathematical Sciences, 2006, Vol. 9, pp 112-119.
- **Farhad Dadgostar**, Abdolhossein Sarrafzadeh, *A formal model of emotional-response, inspired from human cognition and emotion systems*, Research Letters in the Information and Mathematical Sciences, 2006, Vol. 9, pp 89-97.
- **Farhad Dadgostar**, Andre L. C. Barczak, Abdolhossein Sarrafzadeh, *A Color Hand Gesture Database for Evaluating and Improving Algorithms on Hand Gesture and Posture Recognition*, Research Letters in the Information and Mathematical Sciences, ISSN 1175-2777, 2005, Vol. 7, pp 127-134.
- Andre L. C. Barczak, **Farhad Dadgostar**, *Real-time Hand Tracking Using a Set of Cooperative Classifiers and Haar-Like Features*, Research Letters in the Information and Mathematical Sciences, ISSN 1175-2777, 2005, Vol. 7, pp 29-42.

Abstract

In this dissertation, we present the research pathway to the design and implementation of a real-time vision-based gesture recognition system. This system was built based on three components, representing three layers of abstraction: i) detection of skin and localization of hand and face, ii) tracking multiple skin blobs in video sequences and finally iii) recognition of gesture movement trajectories.

The adaptive skin detection, the first component, was implemented based on our novel *adaptive skin detection algorithm* for video sequences. This algorithm has two main sub-components: i) the static skin detector, which is a skin detection method based on the hue factor of the skin color, and ii) the adaptive skin detector which retrains itself based on new data gathered from movement of the user. The results of our experiments show that the algorithm improves the quality of skin detection within the video sequences.

For tracking, a new approach for boundary detection in blob tracking based on the Mean-shift algorithm was proposed. Our approach is based on continuous sampling of the boundaries of the kernel and changing the size of the kernel using our novel Fuzzy-based algorithm. We compared our approach to the kernel density-based approach, which is known as the CAM-Shift algorithm, in a set of different noise levels and conditions. The results show that the proposed approach is superior in stability against white noise, and also provides correct boundary detection for arbitrary hand postures, which is not achievable by the CAM-Shift algorithm.

Finally we presented a novel approach for gesture recognition. This approach includes two main parts: i) gesture modeling, and ii) gesture recognition. The gesture modeling technique is based on sampling the gradient of the gesture movement trajectory and presenting the gesture trajectory as a sequence of numbers. This technique has some important features for gesture recognition including robustness against slight rotation, a small number of required samples, invariance to the start position and device independence. For gesture recognition, we used a multi-layer feed-forward neural-network. The results of our experiments show that this approach provides 98.71% accuracy for gesture recognition, and provides a higher accuracy rate than other methods introduced in the literature.

These components form the required framework for vision-based real-time gesture recognition and hand and face tracking. The components, individually or as a framework, can be applied in scientific and commercial extensions of either vision-based or hybrid gesture recognition systems.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ANN	Artificial Neural Network
AP	Attention Point
ASD	Adaptive Skin Detector
ASL	American Sign Language
BBN	Bayesian Belief Network
CAM-Shift	Continuously-Adaptive Mean-Shift
CCD	Charged Coupled Device
CIELAB,CIEXYZ	Both color spaces, refer to perceptually linear color spaces known as CIE-L-A-B and CIE-X-Y-Z
CP	Color Predicate
CPU	Central processing Unit
CRCNN	Hyper Rectangular Composite Neural Network
DOG	Difference of Gaussian
DTW	Dynamic Time Warping
GHz	Giga Hertz
GMM	Gaussian Mixture Model
GSD	Global Skin Detector
HCE	Histogram Error
HCI	Human-Computer Interaction
HI	Histogram Intersection
HIS	Hue-Saturation-Intensity (Color model)
HMM	Hidden Markov Model
HS	Hue-Saturation (color model)
HSV	Hue-Saturation-Value (Color model)
ICrCb	Intensity-Chrominance red-Chrominance blue (Color model)
ICT	Information-Communication Technology
IUV	Intensity-Luminance-Chrominance (Color model)
LCS	Localized Contour Sequence
MSEPF	Mean-Shift Embedded Particle Filter
NN	Neural Network
PC	Personal Computer

PCA	Principal Component Analysis
PDA	Personal Digital Assistant
PDF	Probability Density Function
PUI	Perceptual User Interface
RBF	Radial Basis Function
RGB	Red-Green-Blue (Color model)
SASOM	Structure Adaptive Self-Organized Map
SMA	Specialized Mapping Architecture
SOM	Self-Organized Map
STFT	Short-Time Fourier Transform
SVM	Support Vector Machine
tMHI	Time-Motion History Image
TMS	Transcranial Magnetic Stimulation
UAV	Unmanned Aerial Vehicle
UI	User Interface
UR	Unreliability Rate
USB	Universal Serial Bus
YCrCb	YCrCb is an encoded nonlinear RGB signal, commonly used by European television studios and for image compression.
YIQ	Color model inspired from human vision system, formerly used in NTSC television broadcasting
YUV	Luminance-Chrominance (Color model, mainly is used in PAL analog television broadcasting)

Chapter 1. Introduction

The role of gesture on its own and the expressiveness it adds when used with verbal interaction, suggests that gesture recognition systems have the potential to open a new and effective communication channel in human-computer interaction. Gesture recognition systems identify human gestures and the information they convey. Hence, *gesture* can be used for controlling devices, games, PDAs, browsers, cell phones, and home appliances. Input devices which can provide information to a gesture recognition system are numerous. Vision system, data gloves, mouse, light-pen, joystick, track-ball, touch-tablet, foot controller, knee controllers, eye trackers, head trackers, data nose, and tongue-activated joysticks can be employed to provide input to a gesture recognizer [1]. Animation creation, virtual reality, performance measurement, and movement analysis are some typical applications of gesture recognition systems.

Gesture recognition systems are in some cases the optimal choice for the user and can potentially revolutionize some problem domains. For instance, physically disabled users frequently have difficulty providing the strength or precision necessary to use traditional computer input devices. Alternative non-verbal communications such as eye blinks, fixated eye gaze, head motions, slight finger motion or other gestures can be employed to remediate these difficulties [2].

In addition to communicative applications of "gesture", some involuntary gestures may give clues to an underlying mental state [3-5]. For example, drumming fingers on the table may indicate impatience or boredom, while rubbing the eyes might indicate that a person is tired. However, the exact impacts are yet unknown and the correlation between gesture and mental state is an ongoing research in behavioral science. In this context, it is worth pointing to Lu et al.'s work [6]. They used blob analysis of head and hands for deception detection. Their primary results show that this approach has the potential of exploring behavioral state identification in the detection of deception. Moreover, researchers are investigating an area in the brain's frontal cortex that interprets other people's action. In a study in the International School of Advanced Studies in Trieste, researchers used transcranial magnetic stimulation (TMS) for pinpointing the "centre of gesture recognition" in human brain. The research did so by sending TMS current through various likely points of the brain while the individuals were asked to watch a short film of someone picking up a box and guess how heavy it is. It was discovered that estimates are widely inaccurate when TMS was delivered to the frontal cortex [7].

1.1 Approaches

There are two distinctive approaches to hand and body-parts tracking. One approach is based on using position sensors attached to the body of the user or wearable devices. This approach is mostly used in virtual reality and recently in applications which require high accuracy in detecting position and recognizing postures. Although this technology is commercially available, it is yet too expensive to be accessible for a majority of computer users. Furthermore, approaches that require attachments to the body are considered more intrusive from the user's point of view. Hence, alternative approaches are sought by researchers to replace the sensor-based gesture recognition technology.

Vision-based techniques provide alternatives for capturing human hand motion [8], and could be very cost effective and are nonintrusive. Therefore, vision-based hand and face tracking can be considered as an alternative for sensor-based gesture recognition. In addition to being less intrusive, there are some scenarios where a vision-based gesture recognition system is possibly the only choice. For example, sterility constraint before operation limits a surgeon in touching mouse and keyboard. And this situation can be facilitated by applying gestural commands gesticulating in front of a video camera.

Research on vision-based gesture recognition systems has shown promising results which has also made this exciting area of research an increasingly popular research topic. Obviously, hands are the most frequent conveyor of gestures. Therefore, most of the research on gesture recognition has concentrated on hand gesture recognition.

1.2 Motivation

Although gesture recognition systems like what was in the Sci-Fi movies such as "S1MØN" or "The minority report" are still far from reality, recent research on applying hand and face tracking in the game environments is increasing. Sony and Microsoft by introducing their vision-enabled game consoles are trying to attract more gamers[1]. Hence, game developers have started to take advantage of the newly introduced hardware, and are making their products "gesture-enabled". The virtual air guitar[2] and the virtual basketball are instances of gesture-enabled game environments.

Gesture recognition and body tracking has hit the commercial fitness market as well. One new such game is Yourself!Fitness for Xbox and PlayStation which is a virtual personal trainer targeting women's fitness. This game adapts to the needs of the individual and offers a number of options based on the individual's personal daily preferences. Another potentially exciting fitness game that has been released is called EyeToy "Kinetic" from Sony Entertainment for PlayStation 2. This game is played using a USB camera that superimposes and integrates the player into the scene on the monitor and tracks body movements. It can be used in a number of modes, including "combat mode" where a player must kick and punch virtual objects (in the air) as they are displayed and move about on the screen[3]. However, the support for gesture recognition in these applications is limited to motion detection from specific view points.

Vision-based gesture recognition requires tracking and interpreting postural and movement patterns of body parts being used for communication. The flexibility of the

---

[1] http://news.portalit.net/fullnews_Microsoft-Aims-To-Challenge-Sony-EyeToy-With-Xbox-360-USB-Camera_1340.html
[2] http://www.tekes.fi/eng/news/uutis_tiedot.asp?id=5077
[3] Matt Slagle. "Video Game Review: Three video games to make you sweat." (New York: Associated Press, Nov 16, 2005, pg 1)

human body and the difficulty to predict its movement when compared to a robot makes human body part tracking, and particularly hand tracking, a difficult task to perform. The hand itself is an articulated and complex object. In addition to its rigid transformation, it has 14 joints which provide a very large number of possible configurations. Therefore, the variability of its 2D projection onto an image makes its detection and classification a non-trivial task. Vision-based hand gesture recognition technology is still evolving with the availability of new hardware and user requirements, and is still an ongoing research. These facts are the main motivations for research in the modeling, analysis, and recognition of gestures.

1.3 Thesis overview

This section states the main contributions of this dissertation and draws the outline of its chapters.

1.3.1 Contribution of this thesis

The main contributions of this dissertation are:

- A novel algorithm for adaptive skin detection in video sequences.
- A new approach for estimating the boundaries of the kernel for blob tracking using the Mean-shift algorithm as a basis.
- A technique for modeling and recognition of pointing gestures in 2D space.

Minor contributions include:

- Employing and comparing the efficiency of two methods of motion detection including the basic frame subtraction and LK-optical flow motion tracking in the adaptive skin detection algorithm.
- Performing a pilot study on the gestural behavior of a small group of primary school students in a math tutoring environment.
- Design and implementation of three variances of the Fuzzy-based Mean-shift tracker for real-time tracking of multiple blobs in video sequences developed in this research.
- Analysis of the Feed Forward ANN and choosing the optimum structure for classifying a 13 gesture alphabet.

1.3.2 Thesis outline

Chapter 2 presents the literature review on vision-based gesture recognition and particularly with a focus on detection, tracking and recognition as primary building blocks of gesture recognition systems.

In Chapter 3, the results of our pilot study on gestural behavior of a small group of primary school students in a math tutoring environment are presented. The required relevant background introducing gesture as a behavioral phenomenon is also introduced in this chapter.

In Chapter 4, a novel algorithm for adaptive skin detection in video sequences is presented. Additionally, the experiments and results of employing two motion detection techniques on this algorithm are presented.

Chapter 5 describes a novel technique for estimating the boundaries of the kernel for real-time tracking using the Mean-shift algorithm as a basis. Experiments and results demonstrating the robustness and stability of this technique in presence of various levels of white noise are presented in this chapter. In this chapter, also three instances of this technique have been proposed and implemented. The first instance is for tracking of multiple skin blobs in video sequences. The second instance is for tracking flocks of skin blobs in video sequences. And, the third instance represents how to apply depth information for robust tracking and occlusion prevention.

Chapter 6 presents a method for modeling pointing gestures in 2D space. Employing a supervised learning method for gesture recognition is also described. Moreover, the implementation of a computationally cost effective design is presented in this chapter.

Finally, summary, conclusions and future work is presented in Chapter 7.

Chapter 2. Literature review

2.1 Vision-based gesture recognition

The following sections present an overview of the current state-of-the-art approaches to hand and face tracking and vision-based gesture recognition. Earlier reviews have been published by Daugman [9], Pavlovic [10], Wu [11] and Geer [2], which indicates to us that there is still not a commonly accepted approach or set of specific techniques for vision-based gesture recognition.

Studying possible scenarios which a user may interact with the system can help to identify the required processing channels and in choosing appropriate techniques. In this context, Wu and Huang [8] in their survey have identified four hand gesture categories for different application scenarios: *conversational, controlling, manipulative* and *communicative.* These gestures can be represented by "temporal hand movements" and "static hand postures" as two separate input channels or a "mixed channel" of data. Hand postures express certain concepts through hand configuration while temporal hand gestures represent certain actions of hand and arm movements. An application's constraints may further enforce specific requirements that may limit the selection of technologies and methods. For instance, hand and face tracking for human-computer interaction requires applying fast image processing techniques while other applications like hand or body posture recognition have more relaxed constraints in terms of required processing power. Availability of the required hardware and robustness against environmental noise are other constraints which may be dictated by the application.

In the literature, the term "*vision-based gesture recognition system*" refers to systems which are able to provide the required functions for recognition of static or temporally sequenced patterns of hand, arm, head, body and other body parts from an image sequence. In this context, a *general purpose vision-based gesture recognition system* can be decomposed to three distinctive sub-systems: 1) detection 2) tracking 3) and recognition. The role of the detection sub-system is finding the object of interest within the image (i.e., the hand, face, or body). The tracking sub-system indicates the movement trajectory of the object of interest within a sequence of images and collects the necessary information for the recognition sub-system. The recognition system interprets the gesture meaning by finding a known pattern in the information collected by the other two sub-systems. In the following sections we discuss the relevant research background of these sub-systems.

2.2 Detection

Detection takes place when the required features in an image of interest are found. The features represent the object of interest and can virtually be any selection of information collectible from the image. Edges, curves, shading and color are examples of the features which can be considered for this purpose. There are two characteristically distinctive feature sets which have been used for hand and face detection: invariant features and non-invariant features. The term "invariant" refers to robustness against possible modifiers on the imager such as *rotation*, *scale*, *view* *point*, and *change of shape*. Ideally, invariant features should be able to identify those objects for which geometry changes due to motion relative to the camera. These types of objects may have different patterns of 2D image over time making the detection process very challenging. For instance, the human face is a three-dimensional object and might be observed under a distorted perspective and uneven illumination [12]. This is particularly true in face detection, where other modifiers such as eyeglasses, different skin colors, gender, facial hair, and facial expressions can affect the detection within the image as well.

These modifiers are usually not independent of each other and therefore often exist simultaneously. It is also agreed that there is no truly pure invariant feature [13]. Rather, there are features that are more or less robust for one or more modifiers. The initial detection of hand, face and body parts within an image is done by locating the relevant features. Therefore selecting the right set of features is the first step toward object detection.

The algorithm which decides whether or not an image contains the object of interest is called a classifier. The combination of chosen features and the classification algorithm comprises the detection technique. In the context of vision-based gesture recognition systems, the detection techniques which have been used and reported in the literature are Eigenface matching [14], support vector machines [15, 16], rule-based techniques [17-20], neural networks [21-24], statistical approaches [25-27], view-based approaches [28-31], dynamic programming [32, 33], and genetic algorithms [34]. Some of these techniques have been successful for posture recognition and frontal face detection under normal conditions. In the following sections, we discuss successful applications of these techniques for gesture recognition.

2.2.1 Shape and contour

The shape silhouette and the edge formation can reveal useful information about the object of interest. Specifically, using shape and contour has been popular for hand detection within still images because the hand silhouette can be extracted either by employing skin color or by using a non-cluttered dark background. In this context there are two design considerations which should be taken into account: i) how to represent the shape, and ii) and how to detect the shape. Considering the fact that an object may have a different shape from each view point, object detection based on shape normally requires detection of a large number of shapes. The preferred approach for this problem in general requires storing the possible posture shapes in a database and searching the database to find the best match for each input posture's shape.

Gupta and Ma [35] proposed a method for static hand gesture recognition from grey-scale images. Static hand gestures are represented by their contours by measuring the similarity between contour representations. The processing steps to classify a gesture included gesture acquisition, segmentation and alignment. They have used the Otsu algorithm [36] to autonomously segment the gesture images and a morphological filtering approach for removing the background noise. The contour of gestures was represented by the LCS (localized contour sequence) and linear and non-linear alignment methods were formulated to determine the similarity between two LCS's. For a set of ten ASL (American Sign Language) gestures, their experiment showed that no misclassification is made using the methods.

While linear search of the shape database is computationally expensive, employing a proper search method is one of the challenges of using shape databases. Different varieties of the search optimization methods can be used in this context. For instance, Lockton and Fitzgibbon's work [37] is based on a method for single hand gesture recognition using template matching with boosting. This technique makes the search hierarchical based on features and therefore much faster than the linear search. Their system interprets 46 gesture elements including 36 letters and digits of the American Sign Language finger-spelling alphabet. The main assumption of this method is that the hand's silhouette is detectable within the image and the classifier matches the hand silhouette to the gesture database. They have reported a 99.87% success rate using 3000 gesture images.

Stenger, Thayananthan, Torr and Cipolla [38, 39] proposed a method for hierarchical object recognition. In their approach, a cascade of classifiers is arranged in the form of a tree in order to recognize multiple object classes. Specifically, for hand pose detection, the required database for the classifier can be obtained from synthesized or real images for training. They have compared a variety of template matching techniques and concluded that *oriented edge marginalized* template using pixel-wise averaging and negative weights result in better performance. They also concluded that a classifier trained with real data provides better results than synthesized data. We have to highlight that the computational cost of curve matching could be higher than silhouette matching and for real-time applications the matching speed is another constraint which should be considered as well. In this context it is worth mentioning Dias et al.'s [40] vision-based open gesture recognition engine called ORGE. They have reported that their platform is capable of detecting hand contours using template matching with an accuracy of 80% to 90%.

Shamaei and Sutherland's work [41] is one of the most interesting approaches in this context. They have proposed a multi-scale approach based on Principal Component Analysis (PCA) and graph matching for video sequences. PCA is a technique which allows the points of a high dimensional space to be represented in a low dimensional space. Each frame of the image sequence can be represented as a single point in a multi-dimensional space. For instance, each image of size 24x24 pixels represents one point in the image space with 576 dimensions. Therefore, a gesture as a sequence of images represents a sequence of points in the multi-dimensional image space. The PCA technique enables the visualization of the gesture sequence in a 2D or 3D space. Their approach for template matching and searching the gesture database was based on locating matches within a hierarchy of images instead of a linear search. They divided up the dataset into groups of images which are similar to each other by blurring the images at different levels such that there is a small difference between similar images. Thus, the whole group of original images may be reduced to just one image which represents the entire group. This approach is interesting because it simplifies the gesture recognition to a curve fitting problem in a high-dimensional space. However, the computational cost of the point matching and curve fitting is high. Moreover, there is a debate on whether this search technique is suitable for large number of shapes or not. Blurring causes the edge information to be smoothed and consequently one of the important features to be ignored in the first step.

A similar approach was also used by Bretzner et al. [42] for a specific application. They proposed an algorithm for hand posture recognition. Hand postures are represented in terms of hierarchies of multi-scale color image features in different scales, position and orientation. In each image, detection of multi-scale color features is performed and hand states are simultaneously detected and tracked using particle filtering for estimating the probability of the likelihood of an image to a set of model hypotheses. Their main aim was detection of five finger counting hand postures as a source of commands to an external device. Image features together with information about their relative orientation, position and scale define a view-based object model. Hand was represented by (i) a model of the palm, as a coarse blob, (ii) five fingers as ridges at finger scales and (iii) finger tips as even finger scale blobs. The different states of the hand model were defined based on the number of open fingers. Their experiments showed that the hierarchical layered sampling approach improves the computational efficiency of the tracker by a factor of two, compared to the standard sampling method in particle filtering.

In addition to searching the shape database, it is also possible to train a classifier for shape detection. In comparison to using the shape database alone, this method is less flexible when new shapes are introduced. However, for a limited number of shapes this approach may be preferable due to the availability of tools and its simplicity. The work of Ardizzone, Chella and Pirrone [43] is one of the examples in using this approach. They have applied a Support Vector Machine (SVM) for body posture recognition from the silhouette of the body. They have used about 100 manually segmented binary images of seven different poses as training data. Based on the training data and the SVM classification approach, 7 different support vectors were found. They have reported that the classifier has successfully been applied for giving manual commands to a robot but they have not reported the precise accuracy of their system.

2.2.2 Skin color as a supportive cue

Approaches based on static shape detection are not suitable for general hand tracking, due to the deformable model of the human hand. The hand is a highly articulated object and its view-based appearance can change significantly. The alternative approach is using a colored glove or skin color as an invariant feature [44-50]. Skin color as a feature has been successfully used in many vision-based gesture recognition systems. There are also a limited number of implementations using infra-red imaging or background elimination techniques, which despite being different, can provide similar data localization. A detailed discussion on skin color detection and related algorithms is beyond the scope of this research, and can be found in Brand and Mason [51], Jones and Rehg [47], and Dadgostar and Sarrafzadeh [52]. In this section, we describe research in which skin color was used as an extra cue for hand and face detection.

It is possible to use skin color together with another detection technique to reach a higher level of accuracy. This approach can be applied for face localization in face detection algorithms. For instance, Ruiz-del-Solar, Shats and Verschae [53] proposed a mixed approach for face tracking in video sequences. They have used *rg* (red-green) color space for skin color segmentation to reduce the search space for the Viola-Jones face detection algorithm. Brethes et al. [54] proposed a method for skin color segmentation based on watershed on the skin-like color pixels. They used chrominance and luminance sequentially for skin color segmentation. By using this technique, candidate face areas are limited in the image. Then, using the discrete Adaboost algorithm [55], the face area is detected. They have concluded that using color as an extra cue significantly improves the hand and face tracking. These are not the only works in this context, and this approach was also applied by other researchers.

The descriptive characteristics of the object also can be used to make narrow the data localization. Although employing this approach is probably not feasible for objects like the hand, it can be used for face detection. Wong et al. [56] propose a hybrid approach for face detection in color images. The main idea is detecting a candidate region as face based on skin color distribution and applying extra checking for reducing false recognitions. The main steps are as follows:

1) Filter the image by a skin detector and consider the biggest blob as face

2) Select the candidate points for eyes using valley detection

3) Test all of the combinations of the candidate eye pixels for validity of the recognition.

The third step is based on the fact that face is a symmetrical object. Then the axis between the eyes based on the approximate position of the eyes on the face represents two mirror images of the same shape. For dealing with the asymmetrical lighting conditions, they used histogram equalization for each side of the axis. If two of the candidate pixels for the eye satisfy the symmetry constraint, they are considered as eyes. To reduce the false detection ratio, the recognized face is compared to an average image of the samples using an Eigenmask of the face. The main reason for eye detection prior to using an Eigenmask image is to find an approximate rotation of the face and to match the rotated face with the Eigenmask. They have reported a 93.39% detection rate in this application.

2.2.3 Accuracy enhancement - Boosting

Boosting is a general method that can be used for improving the accuracy of a given learning algorithm [57]. More specifically, it is based on the principle that a highly accurate or "strong" classifier can be produced through the linear combination of many less accurate or "weak" classifiers each of which may be slightly better than a random generator classifier. The Viola and Jones [58, 59] object detection method is based on boosting and statistical analysis of the basic image block differences called Haar[4]-like features. The detector of each Haar-like feature acts as a weak classifier

[4] The Haar wavelet proposed in 1909 by Alfred Haar is the first known wavelet [www.wikipedia.org].

and the final classifier is built on a cluster of these weak classifiers. Viola and Jones in particular have used this technique successfully in frontal face detection [37, 55, 60-65].

It should be highlighted that this method requires careful selection of the Haar-like features. By choosing the right set of Haar-like features this method may also be suitable for other applications. The Viola-Jones method was also applied for detection by other researchers. In this context, Braczak, Dadgostar and Johnson [66] have applied the Viola-Jones algorithm for detecting fixed hand palm postures. For general hand posture detection applications, they have identified some challenges which should be addressed. First, the hand is an articulated and non-rigid object that makes its shape to be variable over time. Hence, a hand detector based on the Viola-Jones method should be capable of detecting a large number of possible hand shapes requiring a large number of classifiers. This requires a huge database of different hand posture images for training the classifiers. Secondly, applying a large number of classifiers makes the final classifier which is a CPU bound process very slow. Specifically, they have indicated that using more than 8 individual classifiers which are being evaluated sequentially can not be done in real-time on a personal computer (a Pentium 4 – 2 GHz processor). They have also studied the rotated Haar-like feature for tracking hand palm using the Viola-Jones algorithm. Their results show that the robustness of tracking dramatically decreases when rotation is more than 45^{o}. This results shows that rotated features may not give the same results as non-rotated features.

One of the explanations for this phenomenon is the method of implementing the Viola-Jones algorithm. To make the tracking method capable of detecting the object of interest in different distances from the camera, the normal approach is using low resolution image samples. The chosen Haar-like features will be distorted due to the discrete space limitations. The low resolution training makes this distortion higher and the final result would be less robust against rotation. In Figure 2.1, although the same examples were used, the selected features were not equivalent for rotated examples.

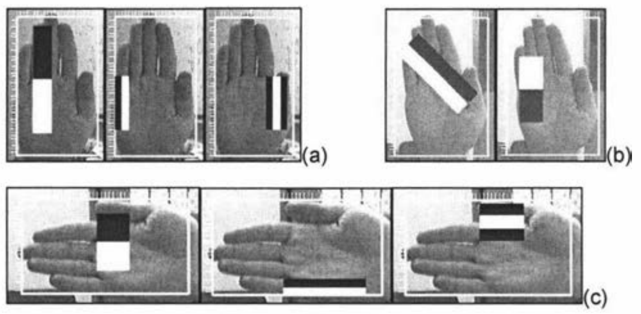


Figure 2.1. a) The first set of features for: a) 0° rotation, b) -15° rotation, and c) -84° rotation.

The performance of the Viola-Jones algorithm using multiple classifiers can be improved using a classifier tree to speed up the search within the classifiers. Ong and Bowden [67] proposed a boosted classifier for hand shape detection based on a tree structure of boosted cascades of weak classifiers. The root of the tree forms the general hand detector. Successfully detected frames by the root are then passed to the branches of the tree where there are specific cascades designed only to detect a specific hand shape. They have used the shape context [68] and the k-mediod for automatic grouping of hand shapes. The k-medoid clustering algorithm is similar to the k-means clustering algorithm. It associates each cluster with a set of images that most closely resemble the cluster centre. The distance metric between the cluster centre and the image is given by the shape context cost, and each cluster is defined by one branch in the second layer of the classifier tree. They reported 97.4% and 99.8% accuracy on the same video sequence of the same person's hand postures they used for training.

2.2.4 Appearance, model and texture

In addition to the shape information other features such as geometrical and kinematic constraints can be employed for hand and body posture detection. For instance, the hand's articulation does not allow the fingers to rotate in any arbitrary direction (at least in a normal hand). Therefore, having the kinematic model, possible states of the hand can be estimated for a certain shape. The geometrical constraint can also be employed for this purpose. For example, the distance between hand blob and face blob could at most be equal to the length of the arm. Hence, this can be employed to limit the search for hand blob within an image after detecting the face or vice versa.

Based on this idea, Urano, Matsui, Nakata and Mizogouchi [69] suggested a body posture recognition system based on outline diameter and higher order local auto correction features of the shape. As the first step, their system uses depth thresholding for detecting the person in the scene. In the next step, it uses the pre-recorded background information to eliminate some of the unwanted areas not eliminated in the first step. Finally, in the third step, the diameters and high-order local correlation features are used for template matching. They have tested their system using 6 different body postures and reported 76.8% to 100% accuracy in detecting different body postures. The role of background elimination using a pre-recorded background is not clear in this research. Assuming that this is a necessary step in the pre-processing, employing this technique could be impractical due to two facts. Firstly, recording the initial background should be performed before the detection phase which requires the intervention of user or another mechanism. Secondly, the environmental changes, such as change of lighting, make the initial pre-recorded background obsolete. Re-recording the background requires that the user leaves the scene for a moment. According to our observation this process could be very annoying with cameras equipped with an auto shutter mechanism. When user leaves the scene, the camera adjusts its shutter to adapt to the new lighting condition. And very often the background's intensity would be different in the presence of the user in the scene.

Rosales et al. [70] have addressed the problem of recovering a 3D hand pose from a monocular color sequence. They have proposed a system that tracks the hand and estimates its 3D configuration on every frame. Their approach is based on a probabilistic modeling method called specialized mapping architecture (SMA) which is used for mapping image features to likely 3D hand poses. SMA is related to machine learning models that use the principle of divide-and-conquer to reduce the complexity of the learning problem by splitting it into several similar ones. In general, these algorithms try to fit surfaces to the observed data by (i) splitting the input space into several regions, and (ii) approximating simpler functions to fit the input-output relationship inside these regions. For more accurate locating and segmenting of the hands, they have used the similarity of the color of face and hands. They have used a dataset of 8000 synthetically generated hand images for training a feed-forward neural network with 5 hidden layers. The main advantage of this algorithms is its linear growth rate of O(M) (M is the number of specialized functions). Employing the divide-and-conquer for shape matching is an interesting technique for this purpose. However, its implementation will have some side effects on the ability of shape detection. Specifically, in their implementation, the detector would be sensitive to rotation. Hence, the detector may fail in a rotated hand posture.

Poppe et al. [71] described a vision-based approach for body pose estimation in video sequences in the context of a meeting environment. In the first step, the silhouette of the body is extracted using a frame subtraction technique. In the next step, using skin color segmentation based on the HS (Hue-Saturation) color space, the face and hands of the speaker are separated from the silhouette of the body. Finally, using inverse kinematics and silhouette matching, the locations of elbows and knees are calculated.

Employing the kinematic model and other cues together, however, raises a new level of complexity and computational cost, although it may provide a more robust framework for hand and body posture detection. In this context, Loutas et al. [72] proposed a mutual information approach for articulated object tracking based on a similarity measure. The measure is calculated on the tracked object image or alternatively on the tracked object texture map accompanied by a confidence map. The use of the object texture map was found to improve the tracker's performance. Articulated constraints are included using a kinematic model on the tracker search range and initial conditions based on the anatomy and the kinematic capabilities of each joint.

Lu et al [73] proposed a model-based integration of visual cues for hand tracking. They have used multiple sources of information which come from edges, optical flow and shading. A hand in their model consists of a base link (palm), and five chains (fingers) connected to the base link through five two-degree-of-freedom revolute joints. Finger parts are modeled as cylinders and the palm is modeled as a six-rectangle-side-solid.

In summary, the applications of model-based hand posture detection are still limited mostly due to the computational cost of employing the inverse kinematic model. In addition, it requires studying the kinematic model of the object in advance which is itself a time consuming task. However, considering the fact that the kinematic model of the hand and body is almost the same for all people, it can be used as a framework for other research. It is expected that this technique will be employed more frequently with availability of more powerful hardware in the future.

2.3 Vision-based object tracking

Tracking is another essential part of a vision-based gesture recognition system. Several different approaches are commonly used for tracking. In some of these methods, the detection of the hand and face are done on each frame of the image sequence. This reduces the tracking problem to a frame-based detection problem. Some others use tracking over the image sequence by initializing a set of features, and tracking and evolving them in the next input frame. The Mean-shift algorithm, in particular, is being used for this purpose and is introduced in the following sections.

2.3.1 The Kalman filter

In gesture recognition systems there are situations where hands or face cover each other. In these scenarios, based on the available information in a single frame, an accurate indication of the position of hand and face is impossible. In such cases, prior knowledge of the direction and the speed of the object of interest can be used to predict its position over time. Considering the fact that our prior measurements of the position of the object of interest may also not be accurate, special considerations for predicting the next position are required.

The popular approach to this problem is using the Kalman[5] filter. The Kalman filter is an efficient recursive filter which estimates the state of a dynamic system from a series of incomplete and noisy measurements. The Kalman filter has two distinctive phases: *predict* and *update*. The predict phase uses the estimate of the previous time-step to produce an "estimate" for the current state. In the *update* phase, measured information from the current time-step is used to refine the prediction to calculate a more accurate estimation. Detailed information about the Kalman filter can be found in Welch and Bishop [74].

[5] In 1960, R.E. Kalman published his famous paper describing a recursive solution to the discrete-data linear filtering problem. Since that time, due in large part to advances in digital computing, the Kalman filter has been the subject of extensive research and application, particularly in the area of autonomous or assisted navigation [www.wikipedia.org].

In one of the earliest works employing Kalman filters for tracking hands, Imagawa and Igi [75] have used the Kalman filter for hand tracking in a vision-based sign language interpretation system. To overcome the problem of hand/face occlusion, they have assumed that the face is not moving and therefore the non-moving region including face can be disregarded in case of hand and face overlap. They have reported an average accuracy of 90% for hand tracking in five video sequences of one person's gesticulations.

In another study, KaeTraKulPong and Bowden [76] presented a method for tracking low-resolution moving objects in the image. In their system, Kalman filters were applied to track multiple objects based on measurement of their position, motion, simple shape features and color contents. The coordinates of the object centroid and the minimum bounding box of the object were modeled by a discrete-time kinematic model. The centroid was modeled by a white noise acceleration model while the height and width of the bounding box were modeled by a white noise velocity model. They assumed that the object moves with a constant velocity and the bounding box does not change extensively. Of course this assumption is not valid for hand tracking. The hand's movement speed is normally variable over time.

The variable speed of the hand is one of the challenges of employing the Kalman filter for hand tracking. The accuracy of the prediction in the Kalman filter is dependent on the accuracy of the information in the past state. In theory, smaller time distances in taking new samples from the system makes the prediction more accurate. However, in a vision-based hand tracking system, the minimum time distance is limited to the maximum frame-rate of the image grabber which is normally about 30 frames per second. This sampling rate may not be sufficient to estimate some hand maneuvers. In some hand gestures, change of direction may occur in a hundredth of second which cannot be sensed using a normal image grabbing device. To overcome to this problem, either using an enhancement to the tracking algorithm or using a faster image grabbing device is required[6]. Despite this limitation, the Kalman filter is still attractive for researchers particularly due to its theoretical foundation and more importantly in this context its capability to limit the overlapping occlusion which is the main concern in some applications.

Another interesting application of hand gesture was proposed by Oka, Sato and Kioke [77, 78]. They proposed a method for tracking user's hand and multiple fingertip trajectories for augmented desktop interface systems. Initially, using a fixed size search window (approximately equal to the size of the hand in the image), the hand is searched for in the input of an infrared camera. Once a search window has been determined for a hand region, the fingertips are searched within that window. The overall shape of a human finger is approximated by a cylinder with hemispherical cap. The location of the fingertips in each frame is predicted based on the location detected in the previous image frame using a Kalman filter. They have suggested that the system noise and observation noise are constant Gaussians noise with zero mean. The system noise model and observation noise model in a Kalman filter are

[6] At present time, fast image grabbers are mostly being used for industrial applications, and are still too expensive for home users.

dependent on the application and should be estimated by the system designers. Therefore, the assertion of constant Gaussian noise model may not be accurate for any arbitrary system. They have reported 99.2% and 97.5% accuracy for single-finger and double-finger tracking in drawing three shapes (circle, triangle and square) with gesture movements using HMM (Hidden Markov Model).

The implementation of the Kalman filter for gesture recognition is also publicly available. The Computational Biomedicine Imaging and Modeling Centre at Rutgers University provided a foundation for tracking human body parts. Their system provides facilities for color analysis, Eigenspace-based shape segmentation and Kalman filters with the ability to track the position, size and angle of different body parts [6].

2.3.2 Tracking using multiple cues

Contextual knowledge can be employed to overcome ambiguities and uncertainties in measurements. Belief networks are an effective method for combining user-supplied semantics with conflicting and noisy observations to deduce an overall consistent interpretation of the scene. A Bayesian Belief Network (BBN) is a directed acyclic graph that explicitly defines the statistical (or "casual") dependencies amongst all variables.

Wang et al. [79] proposed a visual tracking approach based on multiple cues of the image sequence. They used a dynamic Bayesian network for estimating the face area by multiple cues which are (i) frontal face detection using the Viola-Jones algorithm, (ii) the skin color based on Hue thresholding and (iii) the fact that the head can be modeled by an ellipse with aspect ratio of 1:1.2. They have shown that this approach works better in case of having a cluttered background or occluded face. The tracking procedure is based on matching the 3D hand model to a segmented 2D hand image. The matching is based on joint constraints and multi-cue data collected from optical-flow, edges and shadings of the image sequence. Although adding different sources of information increases the accuracy of the tracking algorithm, it also increases the processing time and is therefore less likely to be used as a real-time solution. The experiments in their paper were done on grey-scale image sequences with dark backgrounds. That makes the background elimination less time consuming in comparison to real-word applications in which hand segmentation is sometimes not as easy.

Wern, Clarkson and Pentland from the MIT Media Lab [80] have done an interesting study on hand and body movements. Considering the fact that the human body is a complex dynamic system and that its visual features are time varying noisy signals, they suggested applying a recursive estimation framework (e.g Kalman filter) for accurately tracking movement. They suggest that because human movements are constrained by physics laws they are somewhat limited. The personality of the person also makes the gesture movements even more limited. In their study, they have shown that adding another limiting factor which depends on the behavior of the person makes the estimation more accurate. Their key idea is applying "Physics + Behavior" as an additional control signal for predicting hand movements which makes Kalman prediction more accurate in comparison to using the "Physics Only" model.

Therefore, there is a debate in their study on whether or not the accuracy comes from a more complete model of hand movement or their theory which lies in behavioral science.

2.4 Gesture recognition

Gestures are usually represented by a number of features including templates, global transformation, zones and geometric features. Several methods have been used for gesture recognition: template matching, dictionary lookup, statistical matching, linguistic matching, neural-networks, Hidden-Markov models, and ad hoc methods [1].

Gesture recognition from either a sequence of hand postures or movement trajectory of hand requires analysis of information over time. From this point of view, the HMM is one of the most common approaches to gesture recognition.

2.4.1 Hidden Markov Model

HMM is used widely in speech recognition and recently many researchers are applying HMM to temporal gesture recognition. HMMs are probabilistic models used to represent non-deterministic processes in partially observable domains. They are defined over a set of states, transitions and observations. HMM gets its name from two defining properties. Firstly, it assumes that the observation at time t is generated by some process the state S_t of which is hidden from the observer. Secondly, it assumes that the process state S_t is always independent of all the states prior to t-1[7]. This attribute is called the first order Markov property. In other words, the current state at the same time describes what we need to know about the history of the process, to predict its future state. The first order HMM, is a kind of a discrete model of the Kalman filter.

In an HMM, the state is not directly visible but variables influenced by the state are visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. The challenge in employing HMM is to determine the hidden parameters from the observable parameters. The extracted model parameters can then be used to perform further analysis (e.g. for pattern recognition applications).

In summary, the Markov properties mean that the joint distribution of a sequence of states and observations can be factored in the following way [81]:

$$P(S_{1:T}, Y_{1:T}) = P(S_1)P(Y_1 \mid S_1)\prod_{t=2}^{T} P(S_t \mid S_{t-1})P(Y_t \mid S_t) \qquad \textbf{(2.1)}$$

To define a probability distribution over sequences of observations, the final step is specifying the probability distribution over the initial state $P(S_1)$, the KxK state transition matrix defining $P(S_t|S_{t-1})$ and the output model defining $P(Y_t|S_t)$. Modeling

[7] This is a first-order Markov property.

a system with HMM and indicating the initial and transition probabilities are challenging tasks in this context.

Schlenzin, Hunter and Jain [82] divided the gesture recognition process into two stages: identification of the hand pose within the current image frame and incorporation of the new information into the probability estimation. Their approach was particularly focused on improving the second stage using recursive estimation and HMM. They have modeled a gesture as an unobserved random sequence whose behavior can be summarized with a state transition matrix consisting of the probabilities of each state occurring given only the previous state. The pose information is the observation sequence whose output depends on the current gesture.

The recursion can be expressed mathematically as:

$$\mathbf{x_n = x_{n-1}+y_n v_n} \qquad \textbf{(2.2)}$$

where, x_n is the estimate of the state of the system at time n, v_n is the new information, and y_n is the weighting factor. At each time step new information is obtained from the current frame, and is used to modify the estimate of what gesture is occurring.

There are also some software platforms or open source libraries to facilitate developing a gesture-enabled application. The most comprehensive platform is the Georgia Tech Gesture Toolkit (GT^{2}k) which is based on Cambridge University's speech recognition toolkit, adapted for gesture recognition [83]. GT^{2}k assumes data is being provided by a Data Generator, such as camera, digital glove or accelerometer, in the form of a feature vector. The resulting GT^{2}k classification is then handled by a Results Interpreter as appropriate for the application. Westeyn et al. [83] studied four applications of GT^{2}k in gesture recognition. The first application was "The Gesture Panel" for gesture recognition in automobiles. A low resolution infra-red camera records the silhouette of the driver's hand gesticulated to find a radio station. They used a grammar of 8 hand postures and reported that the accuracy of the system was 249 out of 251 samples (99.2%).

The second application of GT^{2}k was "Patterned Blink Recognition". A prototype system was used to investigate if the intrinsic properties of a person's blink pattern were a personal identification characteristic. They have reported that this system correctly classified 43 of the 48 examples (89.6%) for three participants.

The third application was a "Mobile Sign Language Recognition", limited to 40 words vocabulary and a controlled environment. For this system, they have reported 52.38% accuracy of detection for a vision-based input data, 65.87% accuracy for accelerometer input data and 90.48% accuracy for combined input data.

The fourth application was "Workshop Activity Recognition" system, to monitor the typical actions of a user to determine the user's context. They have reported that this system is able to classify 10 individual gestures with accuracy of 93.33%.

According to GT^{2}k's experiments, it is being concluded that: i) combined features can improve the accuracy of the classifier, particularly for gesture recognition systems, and ii) the topology of the HMM is dependent to the gesture alphabet which requires modification of the topology in case of modifying the alphabet. The second issue is one of the disadvantages of using HMMs in gesture recognition. Finding an alternative approach is still a topic of on-going research.

2.4.2 Temporal gesture recognition

A hand gesture can also be represented as the movement trajectory of hand. Based on this representation, the spatial or temporal appearance of a gesture can be represented as a set of carrying features (e.g. hand postures). In the following paragraphs we discuss some of the research in this context.

Zho, Ren, Xu and Lin [84] described the characteristics of a real-time gesture-controlled system. Movement analysis is based on an intensity image sequence and a hand mask image sequence where each mask covers the moving hand region within the corresponding intensity image. A spatio-temporal appearance is extracted by integrating fine image motion estimation and shape analysis. The extracted spatio-temporal appearance is a temporal trajectory of feature vectors. The feature vector includes horizontal translation, vertical translation, isotropic expansion, deformation, 2D rotation, as well as yaw and pitch extracted from the motion gradient of the shape movement. The average accuracy of the system was reported as 89% for twelve hand gestures.

Su et al. [19, 20] proposed a fuzzy rule-based approach for spatio-temporal hand gesture recognition for sign language detection. Their approach was based on employing hyper-rectangular composite neural-networks (HRCNN) for selecting hand-shape templates and extracting weights of the HRCNN to form IF-THEN rules for a rule-based system. The accuracy of their system is 91.2% on their test dataset with a size of 90 containing 34 basic hand shapes.

2.5 Applications of vision-based gesture recognition systems

There is a wide range of applications in which vision-based gesture recognition systems can be employed. While some applications like sign language interpretation require full capability of vision-based gesture recognition including detection, tracking and recognition for both hands and even in occluded situations, some other applications like computer games can provide more relax constraints. Employing hand gestures as a source of commands to computer or robot is another interesting application which has recently attracted some of the research in human-computer interaction. In the following sections, we introduce the research which focuses particularly on the application side of vision-based gesture interaction.

2.5.1 Gesture enabled applications

Wu, Shah and Lobo [85] have proposed a method for tracking 3D position of hand using a single camera. Their system can infer the 3D location of the finger, letting users describe a geometrical object in 3D through their gestures. They used a pre-calculated histogram-like structure called a Color Predicate (CP) to detect the skin color. Their main assumption was that after skin segmentation and removing disperse noise pixels only both hands and face are left on the image. The next assumption was that the arm gesticulations are around the articulation joints (e.g. elbow). Therefore, the part which is farther to the joint has the fastest centroid. For instance, the fastest centroid in a gesture in drawing a circle in space belongs to hand. For finding the pointing finger they have used the approximation of k-curvature which is defined by the angle between two sequential vectors of the edge information. As they have indicated, this approach works well while the pointing finger is visible in the segmented image but in occluded situations, estimation based on the previous position is required. For determining the position of the elbow and the shoulder their estimation is based on the assumption of having a frontal view of the user which is unlikely to be satisfiable in a real-world application. On the other hand, determining the position of the hand in 3D space is based on knowledge of the length of the arm segment in the image which depends on the subject, which in their experiment, is wearing a shirt with short sleeves.

Licas and Szirany [86] implemented a virtual mouse system in a projector camera configuration. The configuration of the system is based on a digital camera which grabs the output of a video projector with a known image. Matching the grabbed image to the original image requires two transformations. The first transform produces an image geometrically matching to the original input. The second transform then matches the colors of these two images to indicate the difference of the images. The image difference represents the area which is possibly the silhouette of the hand or body of the speaker standing in front of the screen. The detected silhouette is matched to the feature vectors of known hand gestures which is based on Maximum likelihood recognition. They have reported 97.9% to 100% accuracy for detecting 9 gestures, gesticulated by 4 users.

Lementec and Bajcsy [87] proposed a gesture recognition algorithm from Euler angles acquired using multiple orientation sensors. This algorithm is a part of a system for controlling Unmanned Aerial Vehicles (UAVs) in the presence of manned aircrafts on gesture recognition. Their method is based on gesture template matching. Each gesture template is specified by a set of rules indicating the angle of the sensors attached to both arms. They have reported that the system is able to detect 11 bi-manual arm gestures.

The motion information can also be used for some specific applications. Bradski [88] has applied a motion gradient of the image sequence to control a virtual orchestra. This method is based on time motion history image (tMHI) for representing motion from the gradients in successively layered silhouettes. This representation is used to segment and measure motion in a video sequence which is connected to parts of the moving object. They demonstrated the approach for recognition of waving and overhead clapping motions to control a music synthesis program.

2.5.2 Perceptual user interfaces

There is little doubt that vision-based human computer interaction including gesture recognition will have a significant role in the new generation of user interfaces called perceptual user interfaces (PUI). In addition to the recent advances in developing the gesture-based PUIs, studying the scope of applications is an interesting research which may reveal some other potential applications for gesture-based human-computer interaction.

A camera-based mouse system was developed by Betke, Gips and Fleming [89] to provide computer access for people with severe disabilities. Using a camera, the system, tracks the computer user's movements and translates them into the movements of the mouse pointer. The visual tracking algorithm is based on cropping an online template of the tracked feature from the current image frame and testing where this template correlates with respect to the subsequent frame. The location of the highest correlation is interpreted as the new location of the feature in the subsequent frame. They have reported that this system is able to track many body features and it is therefore easily adaptable to serve the special needs of people with various disabilities.

Feature tracking in this research seems not to be robust for general applications. As the results of their experiments show, increasing the width of the search window decreases the accuracy of feature tracking at some stage. This occurs because the selected features, which are rather simple, are too simple to identify the object of interest within the image. Choosing more complex features may be possible, although it may add some more constraints (e.g. always having a frontal view of the face), which may make it inappropriate for some potential disability support applications.

Wilson and Cutrell [90] introduced FlowMouse, a computer vision-based pointing device that is based on hand gesture. FlowMouse uses optical flow techniques for modeling the motion of the hand and a capacitive touch sensor to enable and disable interaction. However, the quality of the optical flow field derived from hand motion is not as good as traditional input devices such as a mouse or a Trackpad. They have, however, presented some scenarios in which the ability to control the environment is acceptable using this technique. Therefore, this approach can potentially provide opportunities for user interface innovations.

Wilson and Oliver [91] introduced a fast stereo-vision algorithm for recognizing hand position and gestures. The depth information is extracted using a stereo-vision system mounted on top of the monitor pointing to the surface of the desk. This system is able to calculate the position of the object relative to the surface of the monitor and therefore provides better location of the hand or pointing gestures for controlling the objects on the screen. They have incorporated this technique into a prototype called GWindows to demonstrate the use of perceptual user interfaces in everyday GUI tasks.

2.6 Chapter summary

In this chapter we introduced the state of the art of research in vision-based gesture recognition and critically reviewed the literature in this area. As stated in the chapter, there are three distinctive sub-systems that a general purpose vision-based gesture recognition system should have. The role of the first sub-system is detecting the object of interest which is typically hand, face or body in this context. The main detection techniques are: shape and contour matching, using a classifier such as NN or SVM, boosting, skin color segmentation and employing the kinematic model of hand or body to estimate its 2D appearance. The role of the second sub-system is tracking the object of interest. Although, the tracking can also be translated to detection in each frame of the image sequence, it is also possible to use the information collected from the previous frame to make tracking more accurate and less computationally intensive. The Mean-Shift algorithm and the Kalman filter are the core approaches which have been successfully used for tracking in vision-based gesture recognition systems. The role of the third sub-system is interpreting the collected information from the previous sub-systems in a form usable in the application at hand. HMM, and Spatial and Temporal analysis of gesture are the main approaches which have been used in this context.

Finally, we referred to some of the applications of the vision-based gesture recognition systems and their increasing role in the human-computer interaction, such as perceptual user interfaces, which are becoming an active research in human-computer interaction.

Chapter 3. Understanding gesture in daily communications: An empirical study

3.1 Introduction

Human computer interaction is one of the major applications of vision-based gesture recognition systems. Hence, designing its components requires having adequate background knowledge about the typical data which should be processed by the system. Designing a general purpose gesture recognition system would require studying typical gestures being used in every context. Such investigations can include comprehensive studies in behavioral science and psychology which is beyond the scope of this research. For the purpose of this research, however, we limited this part of the work to studying the foundations of gesture interpretation which could be useful in determining the more suitable features to extract. In this context, an experiment was designed and carried out on the gesticulation of primary school students while explaining their solution for a counting math problem.

For this study, we analyzed a corpus of over 3000 video clips of mathematics tutoring sessions. This collection was recorded in a primary school in Auckland, New Zealand, and the participants were children aged 5-12. The study revealed how these young students employ different gestures in their communication in a mathematical problem-solving context.

The result of the research presented in this chapter is evidence to the usability of a gesture recognition system in the context of one-to-one tutoring in mathematics. The results also identify the required information which is needs to be collected in an automatic gesture recognition system for intelligent tutoring system applications.

3.2 Research background

In the literature, several definitions and categorizations have been proposed for the term *gesture*. Oxford dictionary defines a gesture as "a movement of part of the body, especially a hand or the head, to express an idea or meaning", implying that gestures may be used for conveying some meaning in communication activities for the purpose of adding extra information of the intention. To relate the use of gestures to Human-Computer Interaction (HCI), Cadoz [92] employs the term "*gesture channel*", associating it with three functions in communication: ergotic, epistemic, and semiotic. Firstly, the ergotic gesture is associated with the notion of work. It allows a direct manipulation with matters, such as typing on a keyboard, moving a mouse and clicking buttons. Secondly, the epistemic gesture gives information relating to temperature, pressure, the surface quality of an object, its hardness, form, orientation, and weight. This allows humans to learn from the environment through tactile experience. The epistemic role of gesture has emerged effectively from pen-based computing and virtual reality. For instance, environmental information can be virtually recognizable by some epistemic gestures in virtual reality environments.

Whilst many gesture-based multimodal interfaces, e.g., Ymir [93] have been designed to deliver epistemic functions of gestures, recently much attention has been paid to semiotic functions, which produce an informational message for the environment. This function gathers the gestures which accompany language, like sign language and symbolic gestures. For instance, a "goodbye" gesture recognized using a smart data glove illustrates how the semiotic functions of gesture can be used in the HCI domain.

Brereton et al. [94] have done research to explore how gestures are used in the context of everyday work. Their goal was to find the gestural themes that might inspire design solutions. They have concluded that some of the gestural themes have overlaps. Their observations also led to the idea that mirroring gestures which are widely used in every day human-to-human conversation have important implications for devices that support communication between people. Therefore, there is potential for couples or groups using gestural devices to easily develop a set of gestures that they agree upon, which could be learned by gestural input devices. However, this conclusion may be biased by cultural knowledge and some *"primal mechanisms that appear to have evolved for the vital management of gestural information"* as they have stated. This suggestion is not commonly used in developing gesture-based UI and approaches employed are mostly application-based as discussed in the following sections. Relevant to the functional understanding of gestures in communicative activities, McNeill [95] and Cassell [96] categorized gestures into three types: *deictic,* *iconic, and metaphoric.* These three types of gestures have different roles in communication.

> *Deictic gestures also called pointing gestures, highlight objects, events and locations in the environment. They have no particular meaning on their own but frequently convey information solely by connecting a communicator to a context. Deictic gestures generally spatialise or locate the physical space in front of the communicator with aspects of the discourse. An example of this type of gesture could be pointing hand left and then right, saying "well, Jane (pointing to the left) was looking at Peter (pointing to the right) across the table. . .". Iconic gestures convey much clearer meaning out of the context than deictic gestures. These gestures represent information about such things as object attributes, actions, and spatial relations. Iconic gestures may specify the manner in which an action is carried out, even if this information is not given in accompanying speech. As Cassell exemplified, only in "gesture" the speaker does specify the essential information of how the handle of the caulk gun is to be manipulated.*
>
> *Finally, metaphoric gestures are more representational but the concept they represent has no physical form; instead the form of the gesture comes from a common metaphor. An example is "the meeting went on and on" accompanied by a hand indicating rolling motion. It is not mandatory to have a productive metaphor in the speech accompanying metaphoric gestures; sometimes the "metaphors" that are represented in gesture have become entirely conventionalised in the language, e.g., describing the solution of a mathematical equation or a physics problem by students [96].*

Kwon et al. [97] showed that the student's gesture is often transformed from a pictorial metaphoric/iconic gesture to a deictic gesture of simple pointing. The use of the three types of gestures would vary with different learning contexts. Specifically, recent work on children communicational behavior shows that deictic and iconic gestures are pervasive in children's speech. Interestingly, children produce deictic gestures before they begin to talk [4]. Recent research within psychology and mathematics education has looked at the role of gesture and embodiment in the different problem domains such as counting [98] and arithmetic problem solving [5, 99], suggesting that different learning situations might allow different uses of nonverbal cues. However, none of the research projects has identified how the learner's level of skill may affect the different uses of gestures.

3.2.1 Why humans use gestures? An account with common ground theory

Many studies have already observed that humans tend to use some sorts of gestures in communication but the plausible reasons for the phenomenon are not clearly accounted in the literature. In the view of the theory of communicative action [100], verbal communication can have the result of binding humans to each other in a mutually-shared pursuit of understanding. This narrow view of communication intentionally excludes the role of nonverbal information geared towards selfish ends. By contrast, Social presence theory [101] and Media richness theory [102] proposed that effective communication, results from matching between the characteristics of the media (here verbal or nonverbal communication) and the task at hands. They implied that nonverbal communication would sometimes be better than verbal communication for some tasks, e.g., nodding in a dispute context.

Accompanying with these two distinctive aspects of communication theory, Clark's common ground theory [103] suggests that if we consider communication as a collaborative activity, it can be viewed to establish a common ground between the speaker and the listener to develop further common ground and hence to communicate efficiently. For instance, pointing is the most effective deictic gesture to support the common ground, making both the speaker and the listener look at the same position. Monk [104] extended Clark's theory, suggesting that all the gestures employed by humans would be peripheral information to assist the communication. From a pedagogical perspective, when children are asked to explain their answers to a particular problem, they convey their thought not only in *speech* but also in *gestures* that accompany that speech. In particular, Patterson and Cosgrove [105] identified that children aged 4 to 8 take longer to respond, shifting their bodies more frequently, and moving their hands more often to convey the message that they do not understand than to convey the message that they do understand.

Other studies, e.g., Goldin-Meadow and Alibali [3] also showed that children can express what they cannot express in speech via gestures. In the light of both the experimental results and Monk's account, the use of nonverbal cues imply that speakers are having some level of difficulties to express themselves, in that they cannot offer the correct verbal explanations of the common ground. Therefore, if a gesture recognition system can detect such gestures in that context, combined with the understanding of what kinds of common ground is pursued, they can recognize the cognitive state of the user.

3.3 A study on gesture-enabled applications

To gain an understanding of the gestures in the domain of the intended application, we performed a study of videoed tutoring sessions. Our study was based on the recorded videos of primary school children while describing a solution based on the Numeracy project learning approach. The focus of the *Numeracy* project is to improve student performance in mathematics through improving the professional capability of teachers. A key feature of the project is the dynamic and evolutionary implementation of the teaching method of mathematics. The Numeracy project suggests the eight developmental stages of mathematical development[8].

The next section describes how we performed an empirical study to see how gestures were being used by students in a mathematical problem solving situation.

3.3.1 Experimentation method

The experiment was based on 37 videoed tutoring sessions of primary school children (17 male and 20 female) from New Zealand primary schools, aged 5 to 10 years, being tutored for mathematics. Sitting in front of their tutor, they were asked to answer two separate questions. The first question was a math problem, followed by how they reached that answer. While they were answering both questions, they were not interrupted by a tutor to ensure the learner behaves in a natural way as much as possible. Participations in the empirical study were on a voluntary basis, and as a consequence, there was not an equal sample size over the different age groups.

[8] See http://www.nzmaths.co.nz/Numeracy for more information.

The mathematical problems were adjusted to the children's developmental stage of mathematical skill. This is explained in the following section. Each video clip was reviewed three times by an expert to identify what kinds of gestures were being used in the learning situation. The first review was to extract facial expression and speech; the second was for body and head movements; the third review was for extracting hand gestures including deictic, iconic, and metaphorical gestures, as McNeil's typology [95].

3.3.2 Participants of the experiment

The participants of the experiment based on the Numeracy project's categorization, are divided into stage 0 to stage 4 (Table 3.1). According to this categorization[9]:

> *Students at Stage 0 are unable to consistently count a given number of objects because the lack of knowledge in counting sequences and/or the ability to match things in one-to-one correspondence. Stage 1 is characterized by the ability of students to count and form a set of objects up to ten but inability to solve simple problems that involve joining and separating sets, e.g., 4+3. Given a joining or separating of sets problem, students at Stage 2 rely on counting physical materials, like their fingers. They count all the objects in both sets to find an answer, as in "Five lollies and three lollies. How many lollies is that altogether?". Students at Stage 3 can count all of the objects in simple joining and separating problems. Students at this stage are able to image visual patterns of the objects in their mind and count them. Finally, students at stage 4 understand that the end number in a counting sequence measures the whole set and can relate the addition or subtraction of objects to the forward and*

[9] http://www.nzmaths.co.nz/Numeracy

backward number sequences by ones, tens, etc. For example, instead of counting all objects to solve 6+5, the student recognizes that "6" represents all six objects and counts on forward: "7,8,9,10,11.". Students at this stage also have the ability to coordinate equivalent counts, such as "10, 20, 30, 40, 50" to get $50 in $10 notes. This is the beginning of grouping to solve multiplication and division problems.

Table 3.1. Different stages of mathematical skills development based on the Numeracy project

Stage	Students skills	Number of students	Yr
Zero	Emergent	10	1-2
One	One-to-one counting	9	2-3
Two	Counting from one on materials	6	3-4
Three	Counting from one by imaging	8	4-5
Four	Advanced counting	4	5-6

3.4 Results of the experiment

A primary concern of this research was studying how young students in the mathematical learning context were using non-verbal cues such as hand and head gestures including iconic, deictic or metaphoric gestures. Table 3.2 demonstrates that about 70% students were using some sort of gesture. In particular, all the students at Stage 4 were using gestures. A chi-square test was employed on this data, revealing a significant use of gesture over all development stages (Chi-square(4)= 2.017, $p < 0.05$). This was further investigated by what gestures they were employing in the mathematical problem-solving context.

Table 3.2. Gesture use in the experiment.

	N	Using gesture	No gesture
Stage 0	10	7	3
Stage 1	9	6	3
Stage 2	6	4	2
Stage 3	8	5	3
Stage 4	4	4	0
Total	37	26	11

We categorized hand and head-body gestures into three levels: low, medium and high. Table 3.3 shows the different gesture patterns over the developmental stages. The students at the high developmental stages were using more hand gestures than those who at the low developmental stages; however, body gestures cannot be so much representative as hand gestures. This may come from the intrinsic nature of a mathematical solution which is very unlikely to be conveyed with head or body gesture.

Non-parametric tests revealed that hand gesture was dominant over the body gesture, and the use of hand gestures was highly dependent on the developmental stage of mathematical skill (Chi-square (8) = 18.55, $p < 0.05$).

Table 3.3. Hand or body gesture use over developmental stage

	Hand gesture			Body/Head gesture		
	Low	Moderate	High	Low	Moderate	High
Stage 0	6	0	1	3	2	0
Stage 1	3	0	3	3	2	1
Stage 2	1	0	3	1	3	0
Stage 3	1	2	2	2	1	1
Stage 4	0	0	4	2	4	0
Total	11	2	13	11	12	2

In most cases, students used their hands for describing their thoughts or the solution to the problems. For instance, "more" or "less" consisted of a single touch of the table or indication of a particular point in the air, followed by another tap or touch to the right of the first one (more) or the left of the first one (less). Figure 3.1 shows some of the samples selected from the recorded videos in a primary School in New Zealand. Figure 3.1a, illustrates the starting of one of these frequent gestures, with arrow indicating that the gesture concluded with a tap to the right. The single gesture that referred explicitly to "adding" consisted of waving two hands crossed, as illustrated in Figure 3.1b. Those two cases were considered as metaphoric gestures. We also categorized pointing and counting gestures as deictic gestures. Figure 3.1c demonstrates a placing gesture toward the right, indicating the location in gesture space, and Figure 3.1d shows a counting gesture for an adding task at hand.

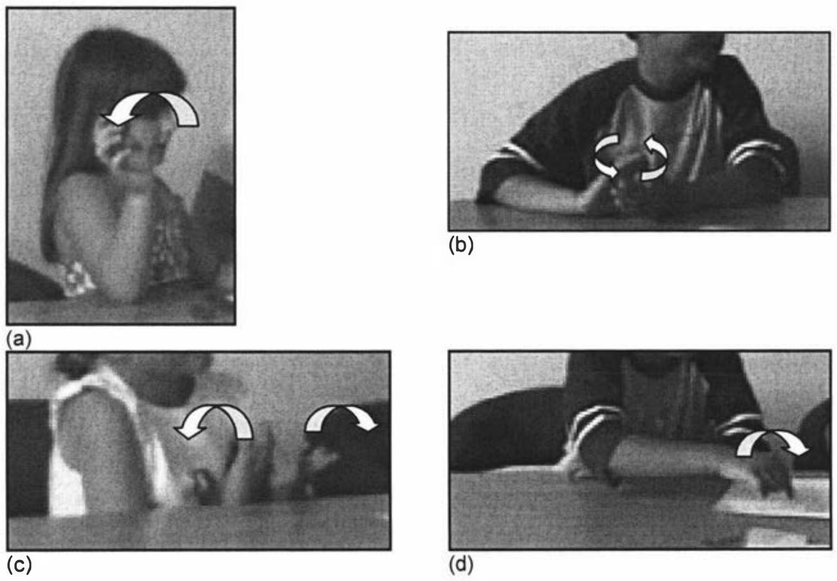


Figure 3.1. (a) A metaphoric gesture: indicating an imaginary point accompanying speech, (b) A metaphoric gesture: waving hands accompanying speech, (c) A deictic gesture: counting, and (d) A deictic gesture: pointing.

However, the iconic gestures introduced by McNeil's typology have not been recognized in this experiment. Figure 3.2 shows a clear pattern of gesture use over the development stages. In the higher stages such as stage 3 and 4, the children used more metaphoric gestures rather than deictic gestures. However, this is not the case for the students at Stages 0 – 2. These results ware assessed for each developmental stage using non-parametric analyses. In Stage 0, significantly many children used deictic gesture, whereas in more advanced stages (Stage 3 and Stage 4), they were using metaphoric gestures.

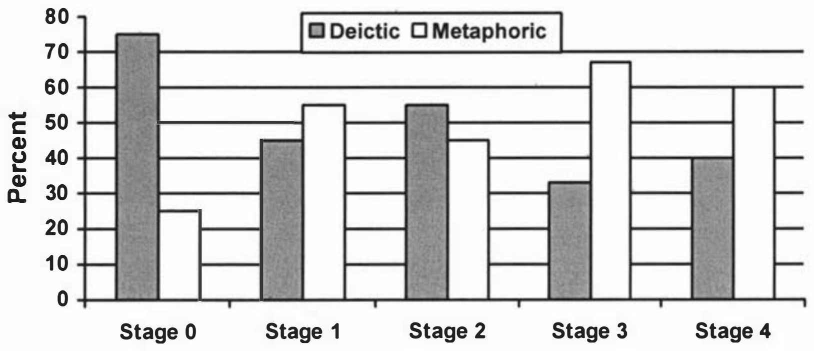


Figure 3.2. Distribution of gestures over development stages

3.5 Conclusions

The result of the experiment in this chapter showed that children in the lower age groups under consideration had mostly deictic hand gestures which were not dependant on speech. These gestures are silent messages that may solely reveal the cognitive state of the learner. In summary, children in the higher developmental stages use more speech-relevant gestures, i.e., metaphoric gestures. On the other hand, the experimental result also highlighted the relatively high rate of 30% using no gestures shows that gesture as the only assessment factor is not reliable in tutoring environments. However, it can be considered as an adjunct technique along with other nonverbal cues and verbal information.

Although, results of our study were based on a small group of children, and cannot therefore be relied as a foundation, they highlight two factors which should be considered in a gesture-enabled system in this context. Firstly, children use different gesture patterns at different developmental stages. Secondly, it is obvious that recognizing metaphoric gestures, which convey more representational meaning, cannot be accomplished using a snapshot of student's hands. This type of gesture is always described by a combination of hand movements and conveyed by characteristic such as direction, force or tempo, even with verbal information. Therefore, it is necessary that other characteristics be collected and analyzed on a sequence of images.

Chapter 4. Adaptive skin detection

4.1 Introduction

One of the problems in vision-based gesture recognition systems is segmenting the hand image from the background [39]. The approaches for solving this problem are either to limit the environment to a non-cluttered background or to use markers to make the hand region easily distinguishable from the background [106]. In many cases using hand markers is impossible due to the type of application (e.g. Analyzing a recorded video). The use of available features of the image is more practical and therefore preferable. In this context, the skin color as a feature for segmentation to detect and track human-body parts is a popular and promising technique.

Although skin color is invariant against modifiers like scale, rotation and shape, segmentation of skin color from a cluttered background is not accurate enough to be applicable to real-world applications. A fast and robust technique for skin color segmentation is still an open research question.

4.2 Research background

Skin color detection and segmentation can be identified as a conjunction of two methods: “color space” and “skin color model” [50].

4.2.1 Popular color spaces used for skin color segmentation

Conceptually, color is not a physical phenomenon. It is a perceptual phenomenon that is related to the spectral characteristics of electro-magnetic radiation in the visible wavelengths striking retina [46]. Digital image processing and different video signal transmission standards have introduced many color spaces carrying different attributes and characteristics. Some of the color spaces have been studied for certain circumstances and for different applications. Meanwhile a wide variety of these color spaces have been applied to skin color modeling and segmentation, including RGB, ICrCb, HSV, HSI, HS and IUV color spaces [12, 42, 46, 56, 75, 107-112].

The main idea in these approaches is using a set of manually segmented images of skin patches as sample data, to estimate a skin color probability density function (PDF) in the color space. This procedure is referred to in the literature as "training". The result of training in most of the color spaces is a connected region, and the boundaries of the probability density function in the color space can be estimated by a limited number of surfaces, lines or points. Obviously, a larger number of dimensions in the color space would require a larger amount of training data and memory space, in addition to more thresholds to specify the desired region.

In some of these color spaces, intensity is implicit (e.g. RGB color space) or it is one of the dimensions (e.g. HSI, IUV, ICrCb). This makes a classifier based on these color spaces vulnerable to intensity changes in the detection phase. For covering a wider range of intensities, a bigger region in the color space is required which may produce more false detections and in turn less accuracy in the final results. A smaller region in the color space on the other hand reduces the number of false detections while also reducing the number of correct detections resulting in a weaker classifier. An overview of different color spaces and their features is presented in Table 4.1.

Table 4.1. An overview on color spaces and their applications in skin detection

Color Space	Formulation (based on RGB)	Applications
RGB	Red, Green and Blue color components each normally represented by a byte (0..255)	Very popular due to its wide use in CRT devices. High correlation between channels and mixing of chrominance and luminance data make RGB color space not very suitable for color analysis. However this color model was applied in some of the skin detection methods.
HSV HIS HS	$H = \arctan\left(\frac{\sqrt{3}(G-B)}{(R-G)+(R-B)}\right)$ $S = 1 - 3\frac{\min(R,G,B)}{R+G+B}$ $I = \frac{1}{3}(R+G+B)$ $V = \max(R,G,B)$	Hue, Saturation and Intensity is a color representation based which is supported in most of the image editing applications, and was also applied successfully for skin detection [71, 75].

XY	X=S cos(H)
	Y=S sin(H)
		A different representation of Hu-Saturation using Cartesian coordinates The polar coordinate system of Hue-Saturation spaces, resulting in cyclic nature of the color space makes it inconvenient for parametric skin color models that need tight cluster for best performance [50].

Opponent	o1(R,G,B) = (R-G)/2 o2(R,G,B) = (2B-R-G)/4	Inspired from the fundamentals of human perception According to Gevers [113] it is independent of highlights and robust to changes in intensity.

rgb
rg
	r = R/(R+G+B)
	g = G/(R+G+B)
	b = B/(R+G+B)
		Discarded intensity, and less sensitive to intensity changes Can be applied for color detection in very low or very high intensity [114]. Hence r+g+b=1, then one of the color components can be omitted in analysis. This feature decreases the color space to 2 dimensions making processing and visualizing statistical analysis (e.g. Histograms) easier [70, 115, 116].

YCrCb
CrCb
	Y=0.299R+0.587G+0.114B
	Cr=R-Y
	Cb=B-Y
		YCrCb is an encoded nonlinear RGB signal, commonly used by European television studios and for image compression. The transformation simplicity and explicit separation of luminance and chrominance components makes this color space attractive for skin color modeling [50, 54].

luv
uv
	I = (R+G+B)/3
	u = R – G
	v = G – B
		Some of the video cameras support this method of color representation [42, 117].

4.2.2 Is there an optimum[10] color space for skin color detection?

Albiol, Torres and Delp in [118] assert that because the separability of the skin and non-skin classes is independent of the color space chosen, the color space itself does

[10] The word "optimum" here, refers to a color model which using that we can gain the highest correct skin detection ratio and the least false skin detection ratio in the image.

not have any influence on the optimum skin color detector for that color space. The mathematical proof is as follows.

Let $D(x_p)$ be an optimum skin detector defined in color space C and $T(x_p)$ be an inverse function that transforms the pixel values x_p from color space C into color vector x'_p in color space C'. Then there exists another skin detector $D'(x_p)$ with the same detection rate and false alarm rate given by: $D'(x'_p) = D(T^{-1}(x'_p))$

Now, assume there is another optimum skin detector in color space C' which has higher detection rate than D' called D". This means we should have another function $(D'')^{-1}$ in color space C, with an accuracy better than D, which is contrary to the hypothesis. Based on this assertion there is no advantage in choosing a specific color space.

Although the above assertion is mathematically correct, some limitations including processing speed, training time and complexity of the classifiers have made this remain as a dilemma. Computation of the skin detection function *D*, in a certain color space may be significantly more expensive than another color space. Therefore, the choice of a color space together with a skin model to classify the skin region with maximum accuracy is still a topic of ongoing research.

In another research, Shin, Chang and Tsap [111] have reported the result of a study on using 8 different color spaces. They have compared clusters of skin and non-skin in CILAB, CIEXYZ, HSI, rgb, CrCb, YIQ, YUV and RGB color spaces, based on four metrics. The matrices they have used are, $tr[S_w]/tr[S_B]$ and $tr[S^{-1}{}_w S_B]$ (which are based on scatter matrix) and histogram intersection (HI) and histogram error (HCE) (that are based on histogram analysis). Their experiment showed that the results of scatter matrix analysis vary for different color spaces, but for other metrics especially the histogram based metrics, the difference is minimal. They finally conclude that the RGB and CrCb are better than other color spaces for skin color detection because the separation factor of *skin/non skin* is higher in these color spaces.

4.2.3 Skin color segmentation techniques

Color models for estimating the distribution of skin color in the color space can be classified into two general categories [119], parametric and nonparametric. These approaches are fundamentally the same and require statistical analysis of the sample data. The result of analysis is a PDF which can be estimated by a set of parameters. Hence, the skin detection can be implemented either using the parameters specifying the PDF or the PDF itself. Generally, nonparametric approaches work effectively when the quantization level can be set properly and sufficient data is available. However, *how* to select a good quantization level for color histogram is not trivial.

Vezhnevest et al. [50] in a survey on pixel-based skin color detection techniques concluded that:

- Parametric skin modeling methods are better suited for constructing classifiers in presence of limited training and expected target data set. The generalization and interpolation ability of these methods makes it possible to construct a classifier with acceptable performance from incomplete training data.

- The methods which are less dependent on the skin cluster shape (e.g. SOM) seem more promising

- Excluding color luminance from the classification process cannot help in achieving better discrimination of skin and non-skin colors, but can help to generalize sparse training data
- Evaluation of color space regardless of a specific skin color modeling can not give a clear indication of how good the color space is. This is due to the fact that different color models react very differently on the color space changes.

A comparative evaluation of different skin detectors is presented in Table 4.2.

Table 4.2. Comparative evaluation of different skin detectors

Method	True Positive	False Positive
RGB Statistical modeling [47]	80% 90%	8.5% 14.2%
Bayes SPM in RG [51]	93.4%	19.8%
Maximum Entropy Model in RGB [120]	80%	8%
SOM and Hue [121]	78%	32%
Elliptical boundary model in CIE-xy [122]	90%	20.9%
Single Gaussian in CbCr [122]	90%	33.3%
Thresholding of I axis in YIQ [51]	94.7%	30.2%

4.2.3.1 None-Parametric skin color classification

One of the most comprehensive studies on statistical skin detection in RGB color space, is the work of Jones and Rehg [47] at Compaq's Cambridge Research Laboratory. They have used a large data set (4675 images containing skin and 8965 images not containing skin) for training and testing an RGB-based statistical model of skin. They compared the histogram and Gaussian mixture model, and showed that the histogram models slightly outperformed the Gaussian models in this context. In addition to the performance they have stated that the training of the Gaussian classifier took 24 hours using 10 parallel Alpha workstation. This volume of computation is considerable in comparison to the histogram model which took several minutes on a single workstation.

They derived a skin pixel classifier through the standard likelihood ratio approach. A particular RGB value is labeled skin if:

$$\frac{P(rgb/skin)}{P(rgb/\neg skin)} \geq \Theta \qquad \textbf{(4.1)}$$

Where $0 \leq \theta \leq 1$ is the threshold which can be adjusted to trade-off between correct detections and false positives. They have reported a detection rate of 80% with 8.5% false positives.

This research with regards to the preparation of such a large dataset for training the classifier is unique[11]. They also claimed that using a large amount of training data, even simple learning rules can yield good performance. This is generally not true. The over training is a known problem in some classifiers, and according to their article as more data is added, performance of the training set decreases because of the increasing overlap between skin and non-skin data.

Kuchi et al. [123] used CrCb color space and statistical color modeling for skin color segmentation and face tracking in video sequences. They have shown that the color of human skin pixels in confined to a small region in CrCb color space which according

[11] The dataset known as the "Compaq skin dataset" in the literature is no longer publicly available.

to Menser and Wien [124] can be formulated by a Gaussian distribution. Therefore, the probability of a skin pixels versus non-skin would be:

$$p(c/skin)=\frac{\exp\left(-\frac{1}{2}(c-\mu_s)^T\sum_s^{-1}(c-\mu_s)\right)}{2\pi\sqrt{|\sigma_s|}} \tag{4.2}$$

where μ_s and σ_s respectively represent the mean vector and the covariance matrix of the training data. Now the probability that a pixel *c* represents skin can be evaluated using Bayes' theorem as follows:

$$p(skin/c)=\frac{p(c/skin)}{p(c/skin)+p(c/non\,skin)} \tag{4.3}$$

Calculating the above probability requires knowledge of the "*skin/non skin*" which is nondeterminist in real-life applications but can be estimated using preprocessing. Kuchi et al. [123] have not indicated the accuracy of their skin detection method and the details of the experiments, but they have mentioned that they achieved an overall accuracy of 82% for face detection in 120 general images downloaded from the internet.

Kjeldsen and Kender [48, 125] have described a technique based on HS color space for separating a single hand from a cluttered background in a gesture recognition system. Their segmentation approach is based on a histogram-like structure called Color Predicate (CP), which is basically an HS histogram which can hold negative values belonging to the background pixels. Each cell of the CP is equal to "n_s-n_b", were n_s and n_b are the number of skin hits and the number of background hits in that cell respectively. They have not indicated the accuracy of this method but they have mentioned that the result of using CP without negative training has been very poor.

The concept of CP for calculating the intersection of two histograms, has correlation to the method introduced by Zhu et al. [109], with the advantage of requiring less computation and therefore faster processing speed.

Hua, De Silva and Vadakkepat [117] applied skin color segmentation based on uv color space for face tracking in noiseless environment. They reported the skin color as a feature that is non-invariant to rotation to keep the face in view by a pan-tilt camera. Their experimental results show that this color model is sensitive to different skin tones, and therefore the skin detector might be adjusted for different skin tones.

4.2.3.2 Parametric skin color classification

Kolsch and Turk [126] have used a parametric approach for detecting a group of features they called "flocks of features", for hand tracking.

Ruiz-del-Solar and Verschea [115] have proposed a fuzzy skin segmentation approach that uses neighborhood information of a given pixel and a diffusion process. The diffusion process is controlled using three thresholds: one for determining the minimal acceptable "skiness" of a skin pixel, one for controlling smooth changes of the pixels "skiness" between neighbors, and a third one for determining the seeds of the diffusion process. In this approach, a pixel belongs to the skin class only if it has a probability of belonging to the skin class over a certain threshold, and if some of its neighbors, were previously classified as belonging to the skin class.

Using the histogram as a probability density function, as mentioned in Ruiz-del-Solar and Verschae [115], and calculating membership probability to skin color based on neighborhood pixels, can reduce the number of morphological operations that are required for noise removal. However, the assumption that the skin color of the current user has the highest probability in the probability density function is not always reliable. Figure 4.1 presents a filtered image based on probability density of pixels. Lighter pixels have highest probability in the training data and darker pixels have the lowest probability. In this condition, the segment that belongs to the skin color of the user is slightly darker than the segment that belongs to the surface of the table. Therefore using the work of Ruiz-del-Solar and Verschae [115], the best case scenario would be segmenting the surface of the table as well as the skin region, which means a weak skin detector.

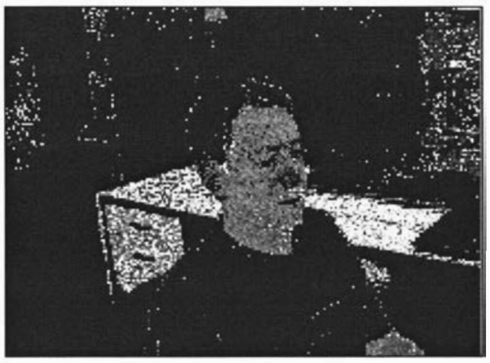


Figure 4.1. In some conditions, the probability density of the skin color of the user is lower than some of the unwanted regions like wood color.

Imagawa, Lu and Igi [75] have used a mixed approach based on locating the face using non-invariant features and estimating the color probability density function for segmenting hands. This idea because of the retraining and adapting to the color space that is used in the image, is more robust. However, in comparison to other methods, it requires a significant amount of computation for face detection. In addition, most of the fast face tracking techniques [55] are not robust to changes like rotation or to situations where a complete frontal view of the face is not presented. This weakness can be remedied by using rotated images at training time and by improving the face detection technique [23, 60, 63, 64]. However, using one of these techniques in itself requires a considerable amount of computation for the detection.

The skin color PDF can also be estimated using other techniques such as Artificial Neural Networks or Support Vector Machines. In this context, ANN is one of the techniques which has been used in research. Neural networks are parameterized non-linear models which are used for empirical regression and classification modeling. The flexibility of neural networks makes them a good tool for discovering more general relationships in the data than the traditional statistical modeling methods. Chen et al. [46] described a method of skin detection using rg color space and a back propagation neural network of five neurons with three layers. The inputs of the net are r and g color components, and a single output 1 or 0 shows whether the input pixel is skin or not. The weight of the neurons' input is randomly initiated. These weights then are re-calculated in the training stage using a set of manually segmented skin pixels. After the training stage, the neural net is used for skin color classification by scanning the image, pixel by pixel. Although they have not indicated the details of the training and the accuracy of the classification method, according to their results on images containing a single face, there is a high false detection together with correctly detected pixels. The high percentage of false detection makes this approach unreliable for skin detection.

The self organized maps (SOM) methodology which dates back to the early 1980s and has been applied to a wide variety of applications including data mining, speech processing, finance engineering, text organization and retrieval and even online game industry [127], was proposed by Brown, Craw and Lewthwaite in [121] for skin color detection. SOM is a neural network with a lattice topology of *k* neurons. Each neuron is associated with a codebook μ_i of codebook vector μ, and each is connected to a certain number of neighboring neurons. Training of the SOM involves randomly initializing the codebook vector and then sequentially presenting each training sample. Each training data *v* is presented as an input vector to all neurons in the network, and the winning neuron n_c with codebook vector μ_c is determined such that:

$$||v_c-\mu_c|| = \min ||v_i-\mu_i|| \text{ , } 1 \leq i \leq k \qquad (4.4)$$

Then the neurons in a specific neighborhood of the winning neuron have their codebook adjusted to a value closer to the input vector according to a parameterized learning function. As training progresses, the learning rate and the size of the affected neighborhood is decreased, and the lattice gradually forms a topologically ordered mapping of the training data. This map is in fact the feature map of the training data.

Brown, Craw and Lewthwaite [121] have applied the SOM for skin detection in frontal face images, and reported 95.5% accuracy of skin detection and 94% accuracy on facial images. However, according to their demonstrated results on images containing non-face skin and general applications, high false detection makes this approach unusable with the proposed configuration. The other issue on this method is the computation cost of the SOM, which is potentially expensive. The SOM in their research was implemented using SOM hardware, which of course dramatically increases the processing speed in comparison to a software based SOM.

Wu and Huang [119] introduced an alternative nonparametric approach based on SOM, called SASOM (Structure Adaptive SOM) for skin color segmentation. The color distribution of each image frame could be modeled by an SASOM, in which each neuron represents a color cluster for the image in the current time frame. SASOM is basically an SOM neural network which adapts its structure to the training data using a heuristic function. Some operators like growing, pruning and merging are applied to the network to modify the structure of the neural network. The modification of the neural network continues until the SASOM reaches its stationary status. This classifier may not be good for the next image frame because of the non-stationary nature of color distributions. Therefore a new SASOM is needed for the new image frame. Their solution to this problem is called SASOM transduction, which updates the weights and structure of the trained SASOM according to a new training dataset. According to their article, the confidence level of the SASOM for 50% labeled data is close to the confidence level of 100% labeled data. This means that this neural network using a semi-supervised training can be successful for the intended application. Although, providing 50% correct training data for a real-world application is not always feasible, the method shows improvement in comparison to SOM in providing the same result with a smaller number of neurons.

4.2.4 Adaptive learning of the skin color

Parametric and non-parametric skin color classification methods introduced in the previous section are based on two distinctive steps of training and classification. The first step is identifying a PDF and the second step is applying the PDF for skin detection. The PDF estimation is therefore static and we call these methods *static skin detection*. There are several scenarios in which a static skin detector will fail in accurately detecting the skin color. Furthermore, the role of the color model becomes significant in presence of the varying features of the skin color. Change of illumination is a typical feature of some applications. For instance, the skin color of a person walking along a corridor passing the light source would have some significant changes in intensity factor. Sigal, Sclaroff and Athitos [128] have reported the results of a study on skin color-based video segmentation under varying illumination. They proposed an adaptive approach for tracking a moving-skin color distribution as defined by an adaptive color histogram in HSV color space of size 32x32x32. Their approach is based on the Storring et al. [129] research that has shown that skin reflectance locus is closely and directly related to the locus of the illuminant, and also based on their observation that skin-color distribution changes smoothly in time varying lighting.

They have used an explicit second order Markov model to predict evolution of the skin color distribution over time. The evolution of the distribution is implicitly defined in terms of translation, rotation, and scaling of the samples in color space. Skin-color Histograms are dynamically updated based on the feedback from current segmentation and prediction of the Markov model. Parameters of the discrete-time dynamic Markov model are estimated using Maximum Likelihood Estimation, and evolve over time. They have measured the accuracy of skin and non-skin detection and the trace of confusion matrix which on average are 86.84%, 93.55% and 1.8 respectively. The numbers showing the accuracy of their skin classifier are slightly better than results reported from other research so far (2004). However, it has to be highlighted that their ground truth data contains three values: true, false and "don't care" which makes their results to not be comparable to those researchers who have used true/false ground truth data.

Zhu et al. [109] proposed a method for adaptive learning of the skin color in still images using HS color space. Their approach is based on a two step process. In the first step a general skin classifier performs a rough skin classification. In the second step a Gaussian Mixture Model (GMM) is derived using the standard Expectation-Maximization algorithm. Then by incorporating the spatial and shape information of the skin pixels to a Support Vector Machine (SVM), one of the Gaussians from the GMM is selected as the skin color model. They have reported accuracy of 88.72% to 96.57% using different features of the Gaussian kernels. The best accuracy achieved has been through applying the "Spatial" and "Shape" related features of the kernel. They have also reported a false positive rate of 30%-37% which is considerable and makes the overall accuracy questionable. The processing speed was not reported in their article. However, it is obvious that the processing time is bound by two passes through the input image and one pass on the HS histogram in addition to the required computation for finding the Gaussian distributions and SVM's classification time. The process explained might be expensive for a personal computer and in turn inappropriate for real-time processing of video sequences.

4.3 The global skin detection algorithm

Based on the discussion of the limitations and the results of other research work discussed in section 4.2, in this research we have used *hue thresholding* for classifying skin pixels and filtering the image. This representation is a one dimensional color space and requires a small color space and therefore a smaller number of samples is required for training[12]. Moreover, the skin color region can be specified by two thresholds. The pixel-based skin detection implementation requires only a few CPU instructions per pixel [110], that makes it a good candidate for real-time skin detection. In addition, it is reliable for ideal conditions with no background noise (e.g. special applications, or using a blue background - Figure 4.2).

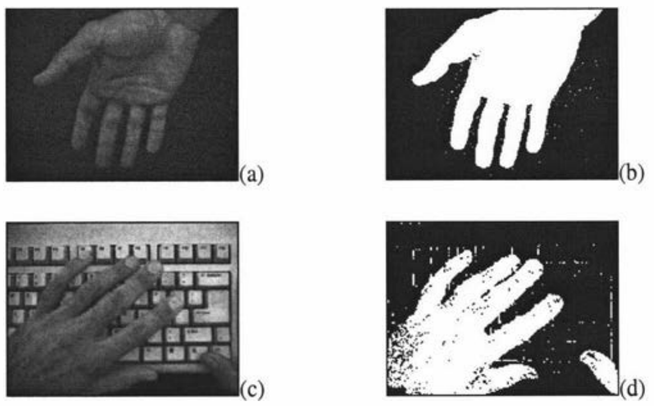


Figure 4.2. (a, c) Original image, (b, d) Filtered image using Hue thresholding.

[12] In fact each pixel requires just two integer comparison

Based on the above description, we developed a static skin detector using hue thresholding. We called this system *Global Skin Detector* or *GSD* (Figure 4.3).

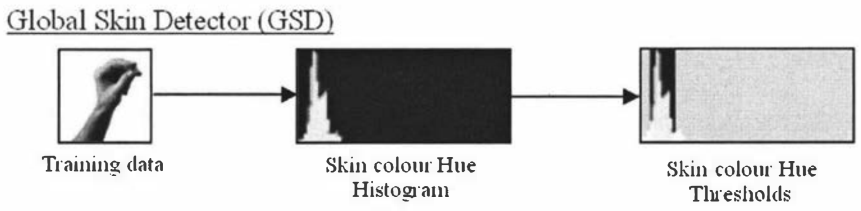

Global Skin Detector (GSD)
Training data
Skin colour Hue Histogram
Skin colour Hue Thresholds

Figure 4.3. Overview of the global skin detector

Although, a GSD can detect the actual skin pixels with a reasonable success rate, it also falsely detects some non-skin pixels. In addition, it cannot distinguish those objects that have a hue factor similar to skin color (e.g. wood). In some situations even the amount of falsely detected pixels is more that the actual skin pixels, which makes it impractical for real-world applications (Figure 4.4).

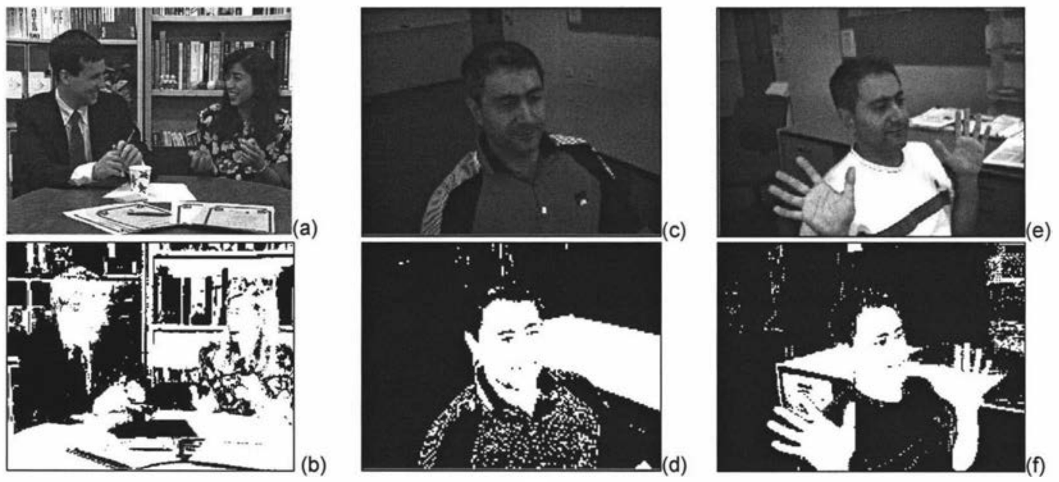


Figure 4.4. (a, c, e) Original Images, (b, d, f) Filtered image based on thresholding Hue factor of skin color extracted from training data (Hue factor of the color of the some background objects are similar to skin color).

4.4 The adaptive skin detection algorithm

Our observations show that:

1. The peak, including position and height, and width of the training histogram are dependent on the image grabbing hardware, and therefore the best results for a static (separate training and detection) algorithm can only be achieved by using the same hardware.
2. Manually changing the lower and upper bounds of the thresholds can significantly improve the results, and the new thresholds are always inside of the boundaries of Global Skin Detector.

This observation, and the fact that in a video sequence people normally move their head or hands, lead us to the idea of adapting the Global Skin Detector's thresholds. The information of the skin in the image can be re-evaluated locally using motion features of the image. That means improving the detection of the skin color through time, as described in the following.

1) Training the Global Skin Detector using a set of training data and specifying the thresholds of skin color in hue color space. The detailed description of this step is introduced in section 4.6.1.

2) Detection of in-motion skin pixels

The next step is detecting the in-motion pixels of the image, using a motion detection algorithm, and filtering the detected pixels using the Global Skin Detector. The output of this step, are those pixels that with a higher probability belong to the skin regions of the image.

3) Recalculating the thresholds

In the next step, the pixels that were considered as moving pixels belonging to the user's skin are used for retraining the detector. In this dissertation we have used a histogram of Hue factor as the base for calculating low (T_L) and high (T_U) thresholds for filtering the image. From the in-motion skin pixels (Figure 4.8c), another histogram is extracted, and the second histogram is merged with the original histogram using the following equation:

$$H_{n+1} = (1-A)*H_n + A*H_M$$

- o H_{n+1} is the new histogram for skin detection (for the next frame)
- o H_n is the histogram for skin detection in the current frame
- o H_M is the histogram of the in-motion pixels of the skin color (Figure 4.8c)
- o and A, is the weight for merging two histograms.

Empirical results show that a value between 0.02 - 0.05 gives the best output for the final skin detector (Please refer to Figure 4.10). The graph presenting the relationship between the merging factor, correct detection and false detection in Section 4.6.3, also validates our empirical results.

4) Filtering using adaptive skin detector

For each frame, thresholds of the Hue factor are recalculated such that they cover 90% of the area of the new histogram. The filter for each frame could be described as follows:

$$f(I) = \begin{cases} true & if \quad T_L(H_n) \le I \le T_U(H_n) \\ false & else \end{cases} \tag{4.5}$$

- I is the Hue factor for each pixel
- H_n is the Hue histogram for the skin color
- T_L is the calculated lower threshold for histogram H_n
- T_U is the calculated upper threshold for the histogram H_n.

4.4.1 The algorithm in summary

Figure 4.5 presents the block diagram of the adaptive skin detector.

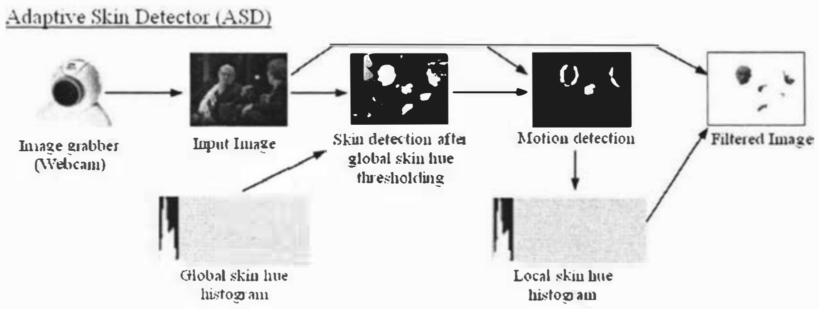

Adaptive Skin Detector (ASD)
Image grabber (Webcam)
Input Image
Skin detection after global skin hue thresholding
Motion detection
Filtered Image
Global skin hue histogram
Local skin hue histogram

Figure 4.5. Overview of the adaptive skin detector

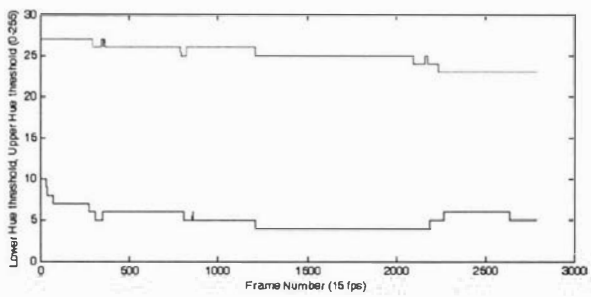


Figure 4.6. Changes in the lower threshold and upper threshold between frames 0 and 2791.

4.5 Motion detection for adaptive skin detection

Motion tracking and detection has attracted considerable research interest in the computer vision community. As a result, there are many different solutions proposed for this problem. Some of the proposed methods are based on the idea of frame subtraction [88, 130, 131]. Alternately some others use more complex features for tracking motion [62, 132, 133]. Hence, there are multiple choices available to be used together with the adaptive skin detection algorithm. Choosing an appropriate method is however an open research question. We employed two motion tracking methods and measured their effect on the adaptive skin detection algorithm. The first method used was the basic frame subtraction method, and the second was the optical-flow motion tracking proposed by Lucas and Kanade [134].

4.5.1 Underlying assumptions

We should note that in this study, our primary assumption has been that the background is not moving, the camera is in a fixed position, and the person's body is the only moving object in front of the camera (e.g. Figure 4.7).

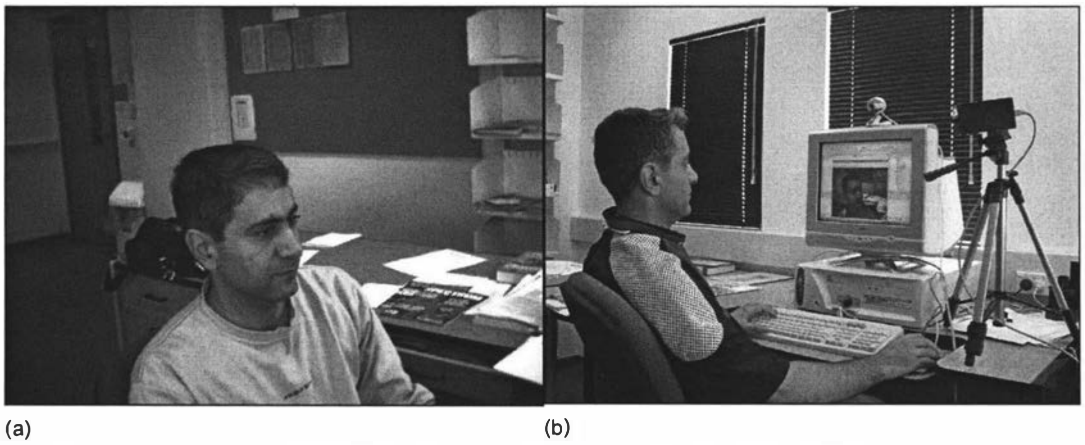

(a) (b)

Figure 4.7. Testing environment, (a) Camera view, (b) Configuration settings.

4.5.2 Frame subtraction motion tracking

Frame subtraction is one of the basic methods for motion detection in video sequences. This method is based on comparing corresponding pixels of two frames and considering those pixels with a difference larger than a certain threshold, as changed pixels. In ideal conditions, changed pixels can potentially belong to a moving object. Although this technique is not reliable for recognizing the moving object itself, it requires a small memory space (one frame) and a small number of operations per pixel (integer subtraction). These characteristics make it suitable for real-time image processing applications.

For comparing two pixels, we tested different components of the pixel for comparison, including RGB, CrCgCb, Hue and Intensity. Our observations show that the most reliable results are reached from RGB and Intensity comparison, and results from those components that did not carry intensity information were not reliable for motion detection. In the final implementation of the frame subtractor, we used Intensity component of the pixels for comparison.

In a real world application, the frame subtraction technique has two main problems. The first problem is that the noise in CCD cameras can cause some sparse falsely-detected pixels. This effect can effectively be eliminated, using an Erode morphological operator. Another problem is that a moving object in a 2D image fills the space that was the background in the previous frame, and the background fills the pixels that were previously the object. Thus, we have two sets of pixels. One set belongs to the object, and the other set belongs to the background. We solved this problem by ignoring those pixels that could not pass through the primary filter for skin color detection. Considering the fact that the body of the subject in the video sequence is the only moving object and the camera is in a fixed position, the probability of detecting some parts of the skin will be higher than non-skin (Figure 4.8c).

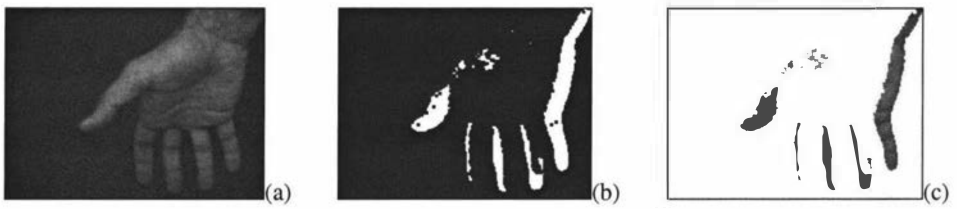


Figure 4.8. A moving hand: (a) Original image, (b) In-motion pixels of the frame, filtered using Hue thresholding (c) Mapping the result to the original image.

4.6 The experiment

4.6.1 Specifying the thresholds of the Global Skin Detector

For calculating initial hue thresholds for skin color, a set of training data of 20 colored images (approximately 3200000 pixels) of hand in which the skin region had been manually segmented were used. Half of these images were recorded in-house using a Dragonfly digital camera, and the other half collected from the Internet. Using this data, a histogram for hue factor was calculated and the lower and upper bounds of the threshold specified such that 90% of the pixels inside the histogram were covered. The values of 3 and 43 were calculated for lower and upper hue thresholds respectively. The range of hue factor was 0 to 255 calculated by RGB values of the input pixels.

4.6.2 Ground-truth data

For evaluating the algorithm, we used three video sequences. Two of the videos are from a human-human interaction and the third is from a human-computer interaction. The length of all of the sequences was 625 frames. The camera was in a fixed position in all of the sequences, there was no movement in the background, and the lighting condition was constant. Table 4.3, presents the detailed information of the image datasets.

The ground-truth data was manually prepared for frames 1 to 625 by equal distances of 25 frames. Each ground-truth frame was an image containing the skin regions (Figure 4.9b).

Table 4.3. The characteristics of the image datasets used for the experiments

	Dataset 1	**Dataset 2**	**Dataset 3**
Video recording device	Unknown	Unknown	Dragonfly digital camera[13]
Resolution	384x288	352x288	320x240
Source	Anvil – Gesture analysis toolkit[14]	CMU Test Images – Paris dataset[15]	Prepared by the author
Frame rate	25fps	25fps	15fps
Output image	RGB	RGB	RGB
Number of people in the scene	2	2	1

[13] http://www.ptgrey.com/products/dragonfly/

[14] http://www.dfki.de/~kipp/anvil/

[15] http://www-2.cs.cmu.edu/%7Ecil/v-images.html

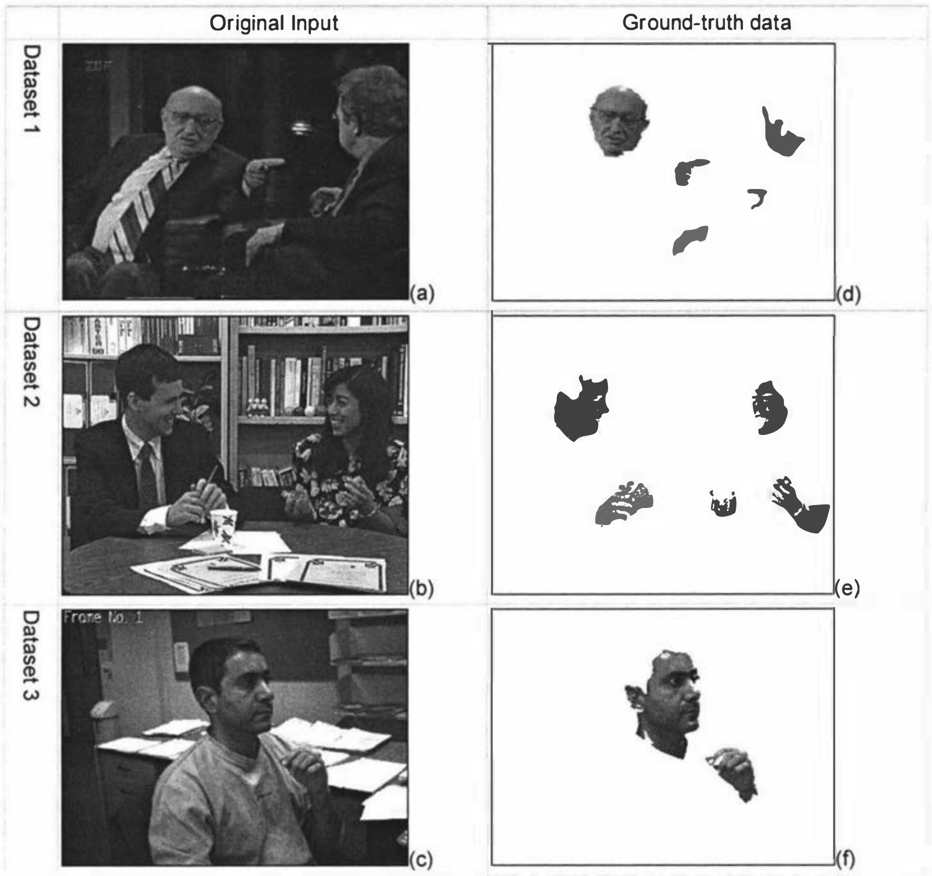


Figure 4.9. Samples of the input images and their ground-truth data.

The first dataset was a video clip of a news interview in which most of the hand and face movements were made by the person on the left side of the image (Figure 4.9a). The second dataset was a chat session between two people (Figure 4.9c). The man on the left side of the image played with a pen using both hands, and the woman on the right side of the image played with a billiard-sized ball. Both subjects have hand movements and slight head movements while talking. The third dataset was a side view of a computer user interacting with computer. He had slight head movements to look at the keyboard and a few hand movements during the interaction.

For evaluating the algorithm we run the adaptive skin detector on all three datasets. The adaptive skin detector was running continuously to the end of the experiment. The evaluation procedure was performed based on the output of the adaptive skin detector compared to the ground-truth data.

4.6.3 Measured parameters

We measured four parameters for each output based on its ground-truth data and for each dataset separately:

1) *Correct detection ratio* (R_c) is the number of correctly detected skin pixels to the actual number of skin pixels in the image. In the ideal system this factor would have a value of 1.

2) *False detection ratio* (R_f) is the number of falsely detected pixels to the actual number of non-skin pixels. In the ideal system this factor would have a value of 0.

3) *Performance* (P) of the algorithm is defined as follows:

$$p = \frac{R_c - R_f}{1 + R_f} \tag{4.6}$$

As indicated by this formula, the *performance* is equal to 1, when R_c=1, and R_f=0. A bigger value for R_f makes the *performance* a smaller number. The ideal system has a performance equal to 1, which means there is no false detection, and 100% correct detection.

4) and finally the *unreliability rate* (UR) that is the number of falsely detected pixels to the number of actual skin pixels. This factor represents the extent of the inaccuracy

(or accuracy) of the output. For instance a value of 2 means the number of falsely detected pixels is 2 times more than the total number of actual skin pixels. Obviously a smaller number for this factor indicates a better output. The ideal value for this factor is zero, which means there is no falsely detected pixel, and 100% correct detection. The reason for introducing this parameter is that the *false detection ratio* cannot represent how inaccurate the output is. For instance it is possible to have an R_f=0.1, which is a small number, but the number of falsely detected pixels to be more than the number of correctly detected pixels (Figure 4.11-f), and therefore R_f alone, may give a false impression of the accuracy or usability of the output.

4.6.4 Choosing the merging factor

For indicating the best value for the merging factor of the adaptive algorithm, our empirical results showed that a value between 0.01 and 0.05 produces the best outputs. The experiment (Figure 4.10) also confirmed this observation. Increasing the merging factor parameters causes a drop in both correct and false detections. Obviously we are interested in keeping the correct detection high and making the false detection lower. As indicated in the Figure 4.10, to have a reasonable correct detection ratio, the merging factor should be between 0.01 and 0.06.

Choosing a merging factor of 0, in fact causes the algorithm to be non-adaptive, and to act the same as a global skin detector. Although that causes the highest number of correct detections but it also produces a high number of false detections. On the other hand, choosing a very small number for the merging factor makes the convergence speed of the algorithm very low. That means reaching a reasonable output and adapting to the new conditions of the input (e.g. turning on a light) will also take more time. Also choosing a very small value, because of the round-up problem may have the same effect of a zero value. In the present research we have used a merging factor equal to 0.05 for the experiments which shows reasonable adaptive speed and stable output.

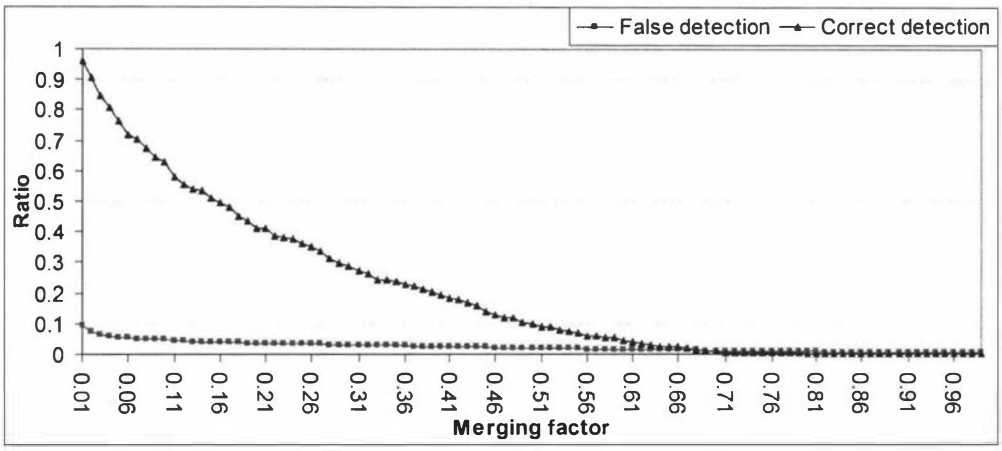


Figure 4.10. The accuracy of the algorithm based on merging factor

4.6.5 Results

Filtering the input image using the described method significantly improved the performance of the skin detector, by decreasing the ratio of false detections and keeping the ratio of positive detections high. In the initial frames, the performance of the adaptive and non-adaptive filter was the same. In Figure 4.11g, the surface of the table caused false detection, as the number of its pixels is approximately bigger than positive recognition. The rigid blobs of falsely detected pixels cannot be removed using morphological operations. The output of last frame shows significant improvement in false detection ratio (Figure 4.11h).

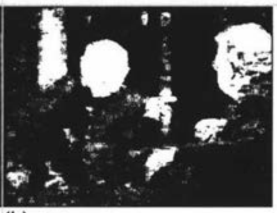
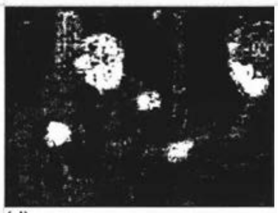
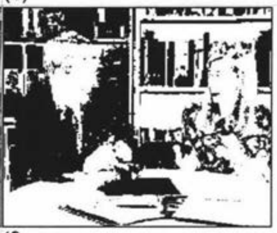
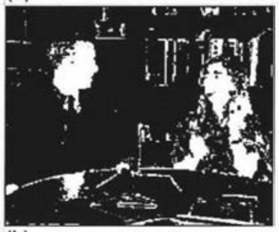
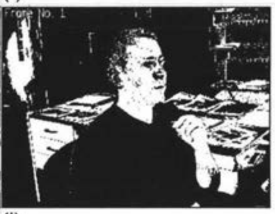
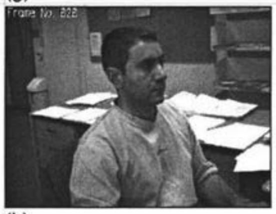
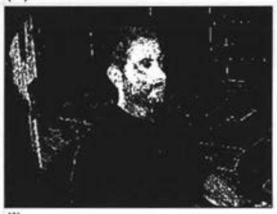
Dataset	First frame	First frame output	Last frame	Last frame output
1	(a)	 (b)	(c)	 (d)
2	(e)	 (f)	(g)	 (h)
3	(i)	 (j)	 (k)	 (l)

Figure 4.11. The changes in the output over the time.

Figure 4.17 presents the behavior of the algorithm in *correct detection ratio.* An average ratio of 0.8 is gained after frame 351, which remained approximately constant. The false detection ratio indicated in Figure 4.17, after frame 351, a false detection ratio approximately equal to 0.05 has resulted and remained constant to the end of the video sequence. The performance of the algorithm, after frame 251, was relatively stable between 67% to 72% which combined with the low ratio of false detection, represents the stability of the algorithm over time.

The unreliability rate significantly changed over time. For instance, the ratio for dataset 2, that was initially 7, dropped to 1 in frame 476, and slowly climbed to 1.8 in frame 601. As represented in Figure 4.17d, the output for dataset 2 is initially very unreliable, but in the last frame output, the detection quality is much better. However there are still some small falsely detected regions. The reliability rate for dataset 1 and 3 was much better, and after frame 176, it was relatively constant. The output for all three datasets is presented in Figure 4.17a, and shows improvement.

4.6.6 Sparseness of the correctly and falsely detected pixels

For measuring the sparseness of the detected pixels, we used four different morphological operators on the output images of the datasets. Obviously in the case of sparse false detections, the falsely detected pixels can be eliminated using Erode morphological operator. As indicated in Figure 4.18, Figure 4.19, and Figure 4.20, there is no specific pattern on sparseness of the correctly detected pixels. However, there is a vertical shift in measured parameters, based on using different combinations of Erode and Dilate morphological operators. It is also shown that employing Erode operator in all cases dramatically reduces the false detection ratio which means falsely detected pixels are sparser than correctly detected pixels.

4.6.7 The comparison of the behavior of the algorithm using optical-flow motion tracking

We run another experiment on the algorithm to verify the effect of another motion tracking method. The alternative method used here was optical-flow motion tracking, and the experimental platform was the same as we discussed in the previous section.

4.6.7.1 Optical flow motion tracking

The optical-flow motion tracking technique is a more accurate method for object tracking in video sequences. The idea is based on the fact that lighting condition in two sequential frames is relatively constant, and therefore the intensity of each point of an object would be the same. If the position of a specific intensity changes in the next frame, then we can assume that the point has moved to the new place, in the time in between two frames. Tracking of the segments in optical-flow algorithm is based on the fact that the intensity of a moving segment changes slightly in two sequential frames. Therefore, the next position of a track point will be around its previous position.

Considering the possible movements that the object may have (e.g. moving left, right, up and down, rotation, moving toward or away of the camera), there are several implementations for this technique. In this research we have used an implementation proposed by Lucas and Kanade [134] which is called Lucas-Kanade optical-flow tracking, and is available in OpenCV [135], as measurement for motion tracking. The implementation is based on choosing a number of tracking points, a small block size to track and a neighborhood distance to search. These parameters may affect the processing speed of the algorithm. To make it suitable for real-time applications, we chose 100 track points and block size 5. The proposed optical-flow algorithm has two main parts: Initialization track points and Tracking.

4.6.7.2 Initialization track points

Normally, for initializing the optical-flow algorithm and setting the track points, a set of small segments that have intensity different than neighborhood segments are chosen. Based on this approach most of the track points will belong to the edges in the image. Our observations show that using this approach for tracking skin segments produces poor results. Because of smooth intensity changes of the skin color of the face, most of track points that belong to skin are located around eyes, nose and mouth, and the number of track points belonging to the skin segments of the face is very small (Figure 4.12).

To improve the results, we initialized the track points on the skin segments (based on global skin detector) with equal distances (Figure 4.13). This method has two advantages for this application. First of all, we will have a reasonable number of track points on the region of interest, which is the skin color here. Secondly, by tracking motion, we can find bigger parts of skin segments.

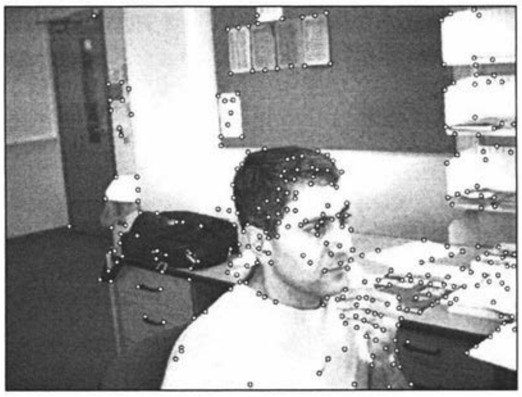


Figure 4.12. Track points based on traditional approach (good features to track).

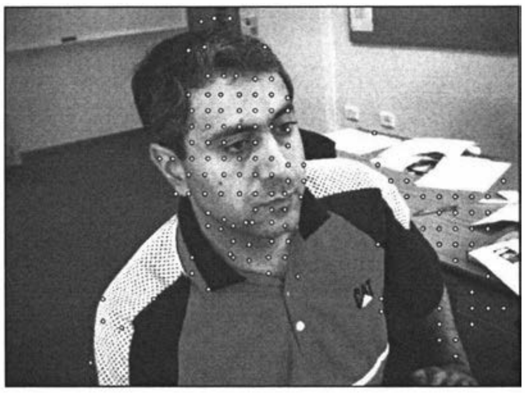


Figure 4.13. Initializing track points, based on primary skin color filter and equal distances.

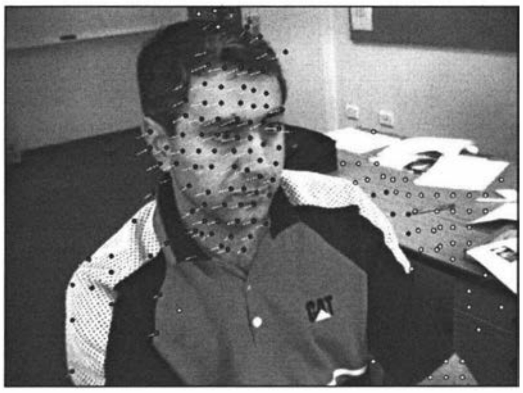


Figure 4.14. Tracking skin color segments using LK optical -flow motion tracking and selected features to track.

4.6.7.3 Tracking

In the initialization section, we put the track points on those regions that are potentially skin. Therefore, recognizing motion in these points reveals user's skin segments with a higher probability. In Figure 4.14, darker points have been recognized as moving skin segments. The small lines connected to these points, represents direction of the movement.

Figure 4.15 represents the in-motion skin detected by the optical-flow tracking method.

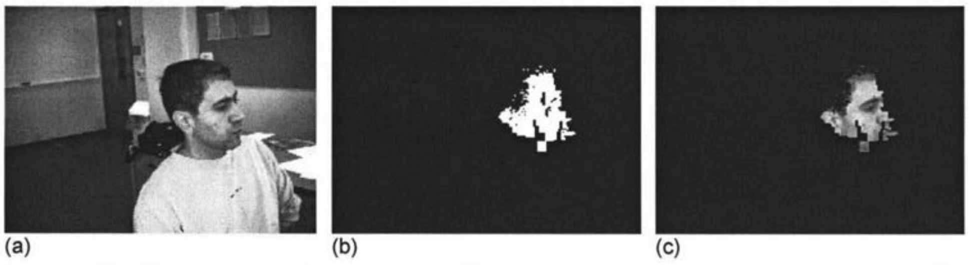


Figure 4.15. Nodding head: (a) Original image, (b) In-motion pixels of the frame, filtered using Global Skin Detector (c) Mapping the result to the original image.

4.6.7.4 Optical-flow vs. frame subtraction

The results of this comparison are presented in Figure 4.21. The average correct detection using optical-flow shows about 20% decrease against frame-subtraction. The average false detection of frame-subtraction is slightly better than optical-flow. The performance of the frame subtraction is slightly better than optical-flow and the unreliability rate (UR) is significantly better for frame subtraction. The comparison shows that optical flow motion tracking, as a measure of motion, is not as good as frame-subtraction for this particular algorithm and application.

4.7 Chapter summary

In this chapter, we introduced the adaptive skin detection algorithm. This algorithm has two main components: i) the static skin detection which is a skin detection method based on the hue factor of the skin color, and ii) the adaptive skin detector which retrains itself based on new data gathered from movement of the user.

This algorithm was designed for human-computer interaction applications. While the user is interacting with the computer, the system captures the movements and retrains itself based on the user's skin color. The application is robust against environmental noise like lighting condition, noise caused by fluorescent lights, changes in intensity level caused by automatic gain control in digital cameras, and background noise. The performance of the system is less than 20 milliseconds per frame, on a P4 2.2GHz PC, for a 24bit RGB image size 640x480 with non-optimized C++ code (Figure 4.16).

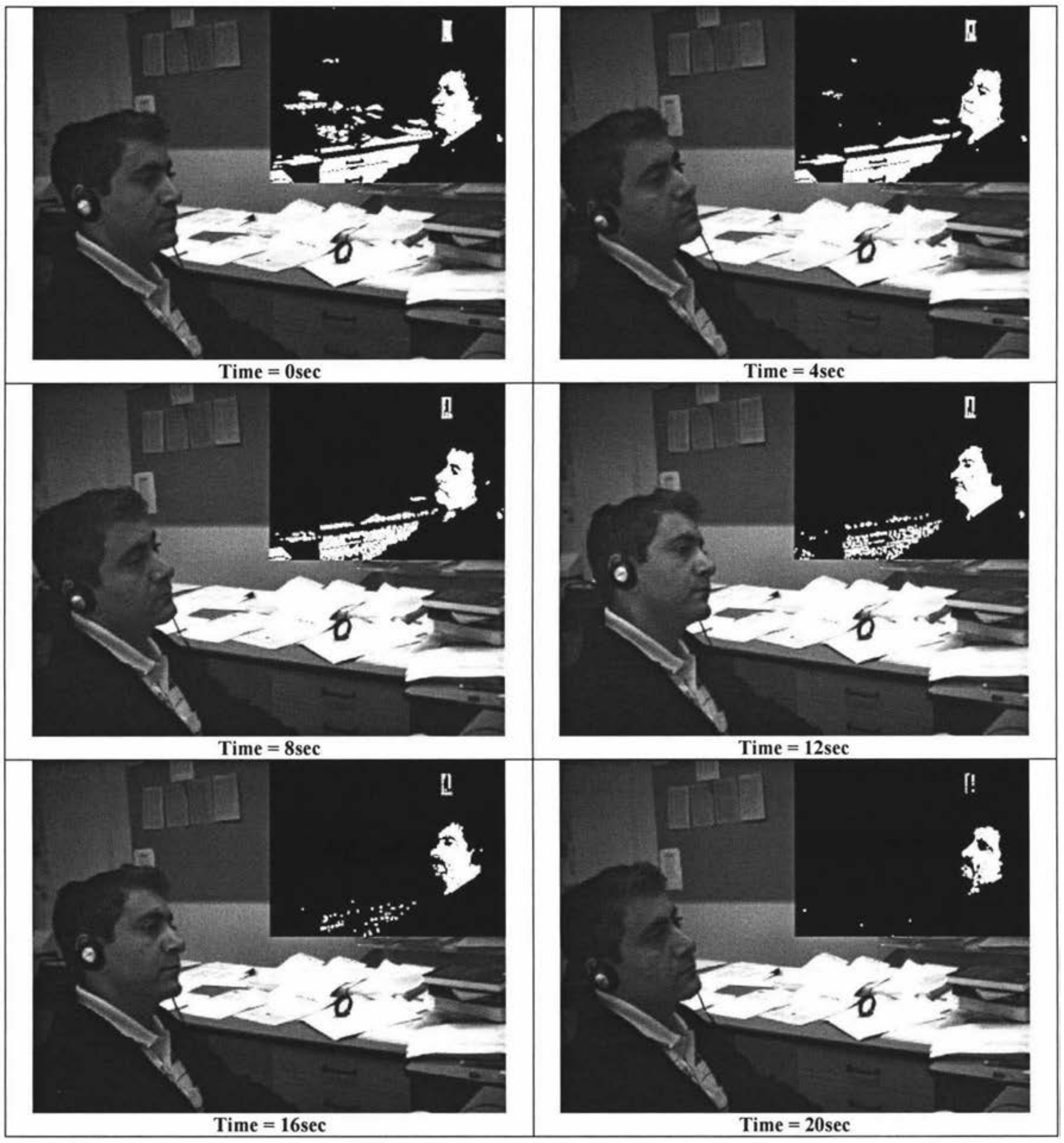
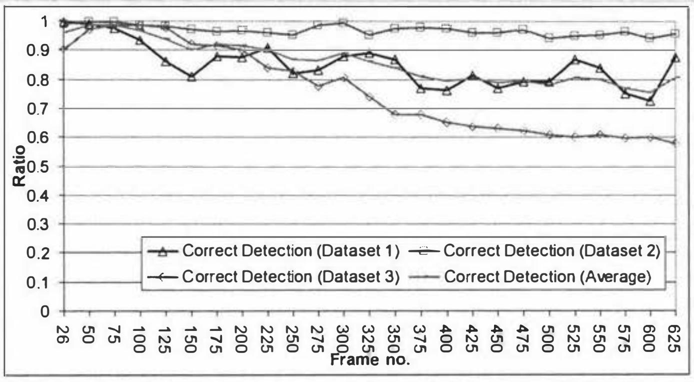
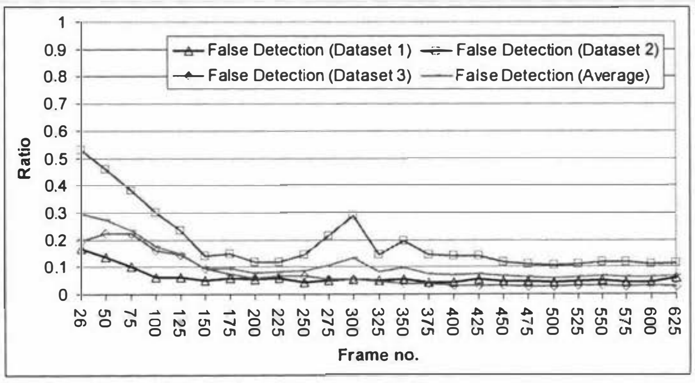


Figure 4.16. A short video sequence presenting the behavior of the adaptive skin detection algorithm over time. On the top-right corner of each image, the detected silhouette of the skin is presented. The falsely detected areas were gradually eliminated over time [16].

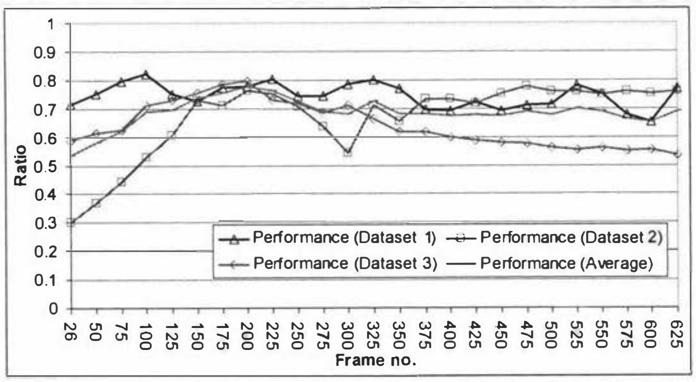
[16] Videos are available in the enclosed CD. Also available online at: www.massey.ac.nz/~fdadgost and ngits.massey.ac.nz



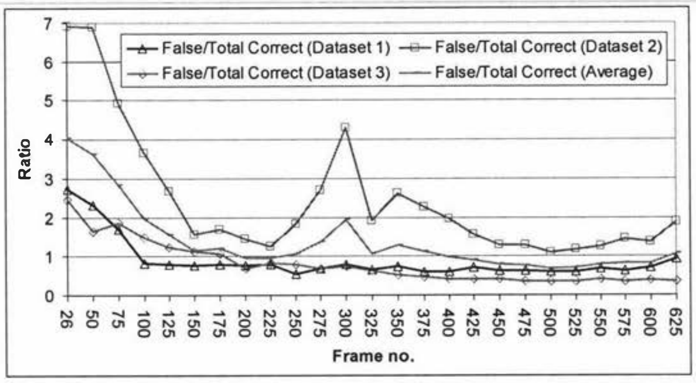
(a)



(b)



(c)



(d)

Figure 4.17. The behavior of the adaptive skin detector using frame subtraction motion detection

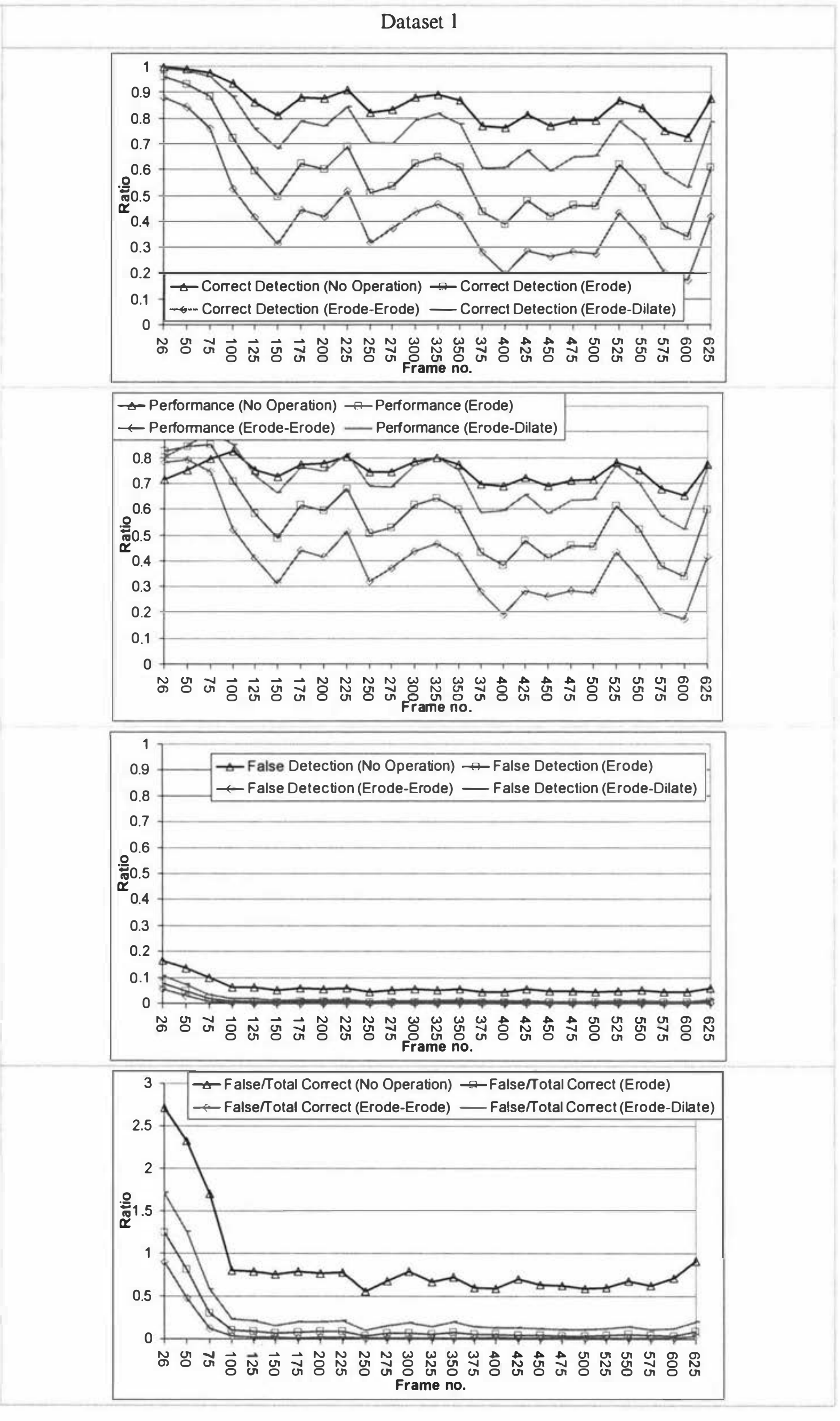


Figure 4.18. The results on using morphological operators on dataset 1.

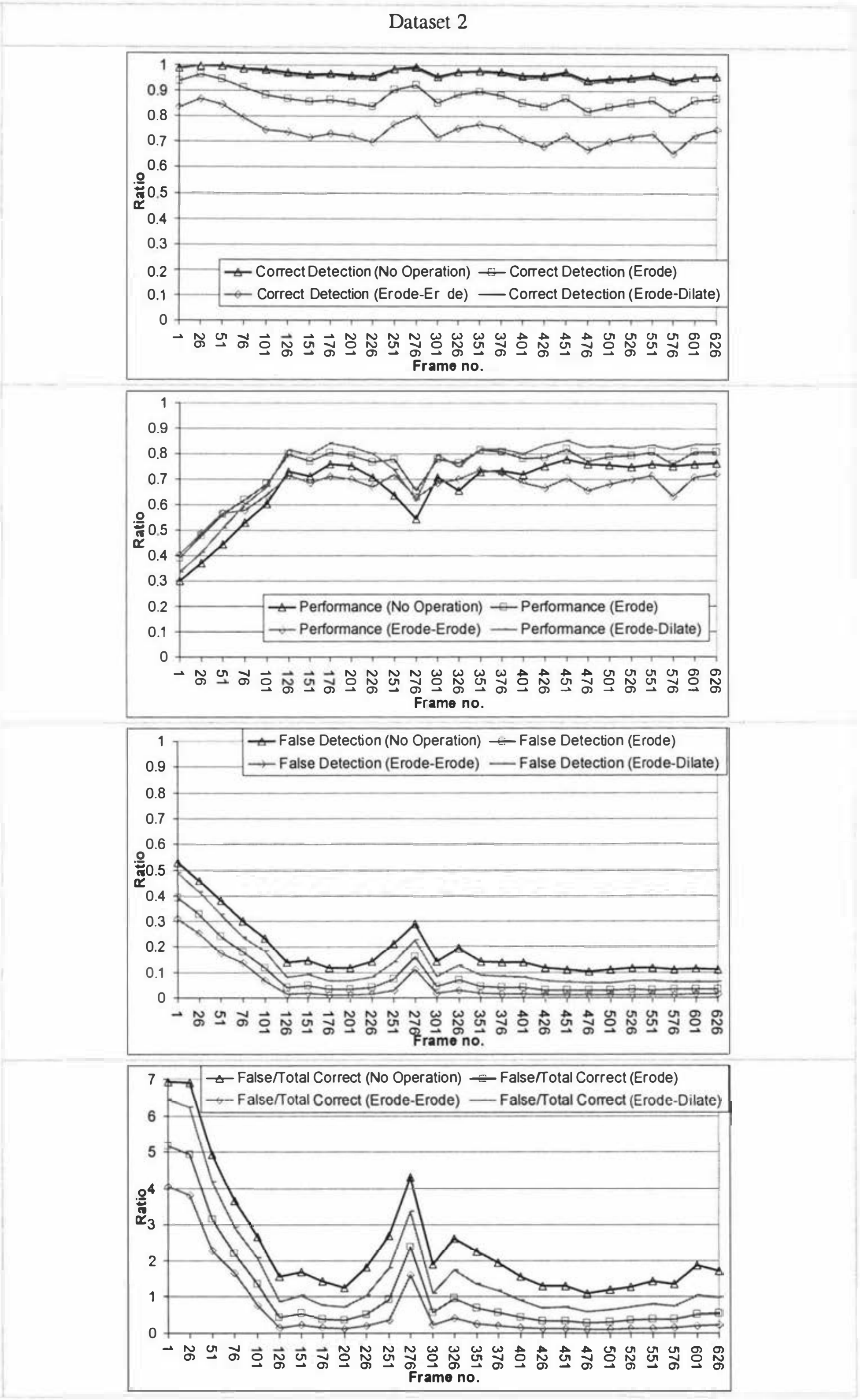


Figure 4.19. The results on using morphological operators on dataset 2.

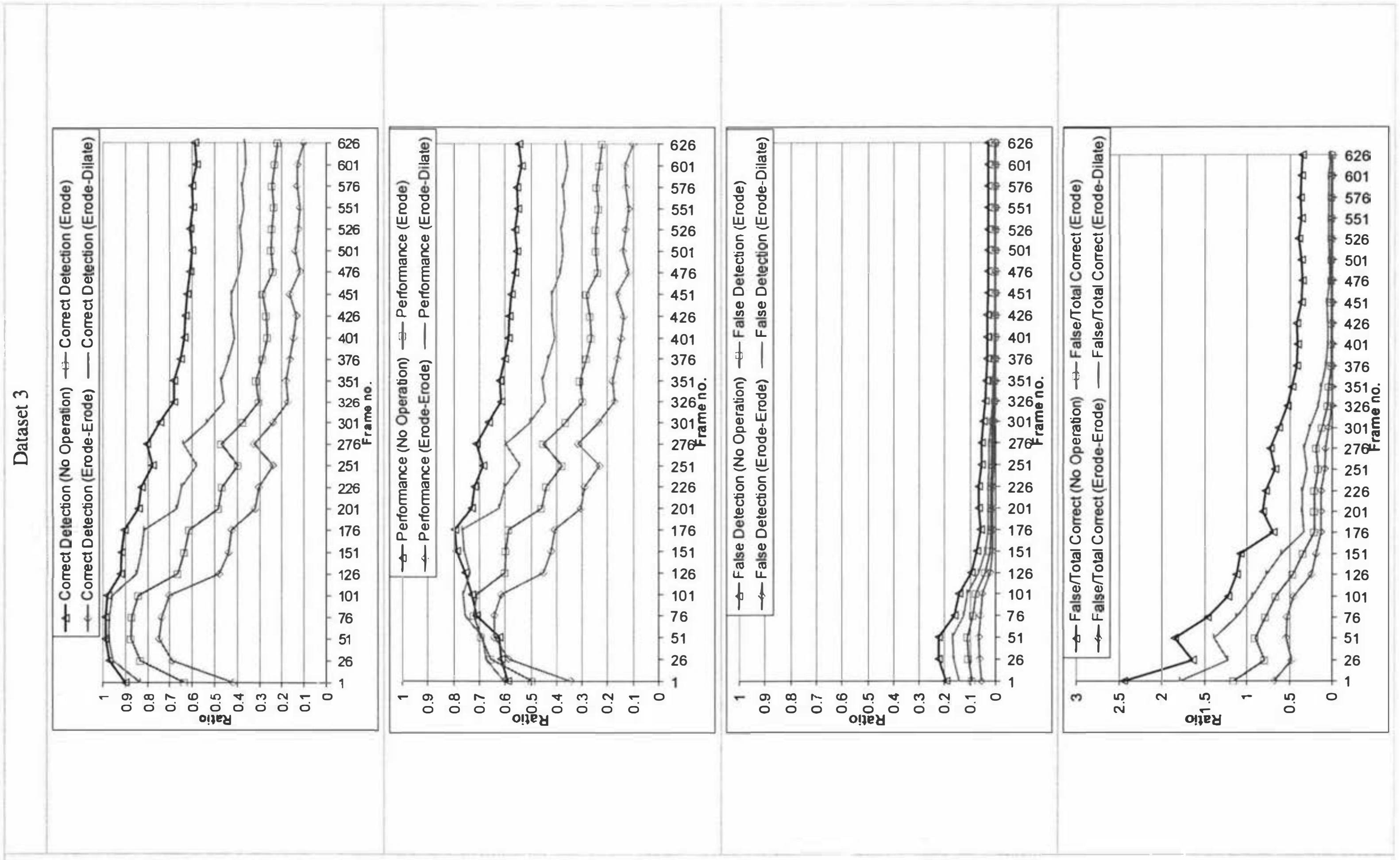


Figure 4.20. The results on using morphological operators on dataset 3.

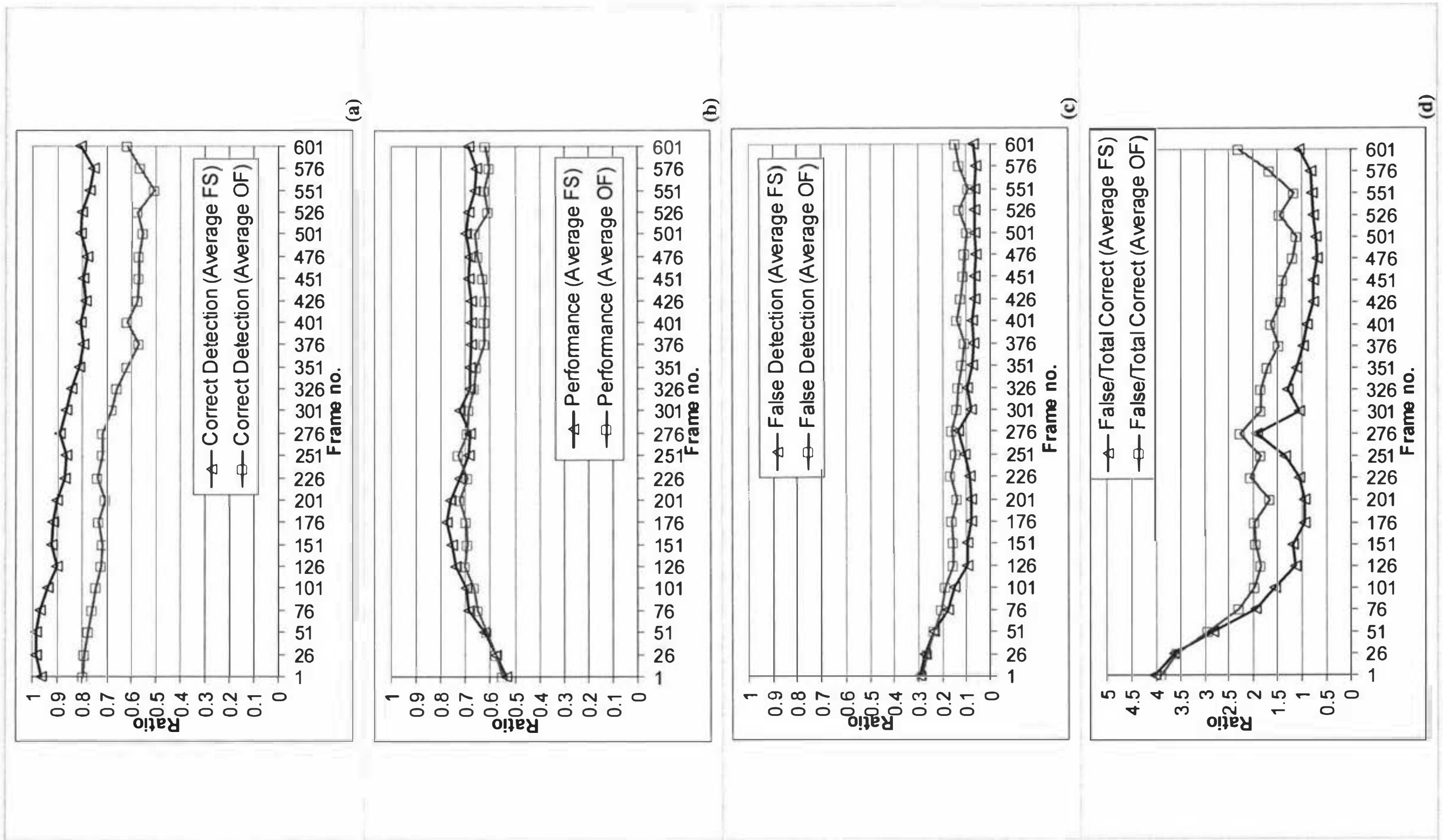


Figure 4.21. Comparison of measured parameters of the algorithm, using different motion tracking methods (FS: Frame Subtraction, OF: Optical Flow). The values are the average of the values measured for all three datasets. However, calculating the average for a small number of datasets (3 in this case) may bias the results toward one of the datasets (e.g. dataset 2 dominates the results shown in graph d).

Chapter 5. A novel approach for robust tracking, based on the Mean-shift algorithm

5.1 Introduction

Vision-based gesture recognition systems require identifying and tracking the boundaries of the skin segments which in the case of this study specifically be hand and face. Color information provides an efficient feature for this purpose because of its robustness to partial occlusion, geometry invariance and computational efficiency. The output of the skin detection algorithms on the other hand is a disperse set of detected skin pixels. The quality of the detection is dependent on several parameters including the background noise and lighting condition. These parameters may in some applications not be controllable. Considering that providing the ideal conditions for a real-world application is impractical, improving the quality of the output using enhancements techniques is considered a desired solution in skin detection systems. These techniques may themselves require considerable amounts of computation. Moreover, as we showed in the previous chapter, even using simple morphological operations (e.g. Erode-Dilate) will not fully eliminate the sparse falsely detected pixels.

Using a tracking algorithm which does not require a rigid segment and can handle ambiguous boundaries is another approach for tracking a disperse set of points. In this chapter, we introduce the Mean-shift algorithm and its application in hand and face tracking by tracking the disperse set of detected skin pixels detected by the skin detection algorithm. Among the different tracking algorithms, the Mean-shift tracking algorithm has recently become popular due to its simplicity and robustness. The Mean-shift algorithm is a nonparametric statistical method for seeking the nearest mode of a point sample distribution.

The Mean-shift algorithm and its applications in pattern recognition was originally introduced by Fukunaga and Hostetler [136] in 1975 for data clustering. They referred to their algorithm as a "valley-seeking procedure". The first reported application of the Mean-shift algorithm in image processing is probably the one introduced by Cheng [137] in 1995 for *mode seeking* and *clustering.* Following some successful applications including Cheng's work, the Mean-shift algorithm attracted the interest of the image processing community in different application areas including feature tracking.

5.1.1 The Mean-shift algorithm

The basic idea of object tracking using the Means-shift algorithm is finding the highest density of features within the image using a search window. The search window has been called the "kernel" by some in the literature, and we will use the same term throughout this thesis. The modeling of features and the feature space, is an implementation issue, and may vary based on the target application. Particularly for object tracking, this function can be the value of a measurement function evaluating the feature within the kernel. This value is also called the density of the kernel. The initial placement of the kernel is based on a placement strategy.

The Mean-shift algorithm is an iterative algorithm. In each iteration, the kernel moves to a place where its center of gravity matches its geometrical center. The iterations continue while the kernel value is increasing. Figure 5.1 presents the iterations that took place to find the kernel with the highest density. The detailed description of the Mean-shift algorithm for blob tracking is introduced in section 5.1.2.

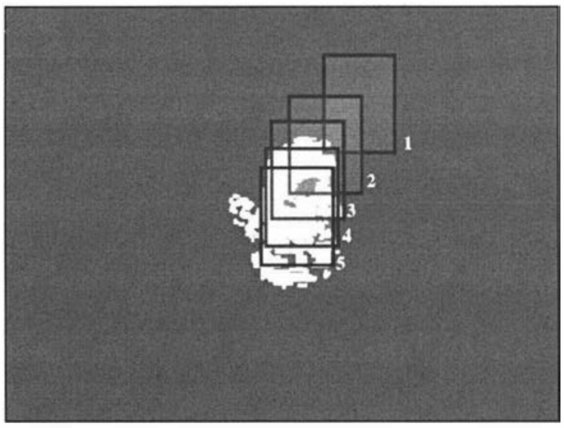


Figure 5.1. Iterations, to find the highest dense kernel using the Mean-shift algorithm. The initial position of the kernel is [1] which has a small overlap with the face blob. 2, 3, 4, 5, are the new positions of the kernel to match the center of gravity of the kernel to the geometrical center of the kernel.

5.1.2 Automatic resizing of the kernel

Choosing the proper size and the initial placement of the kernel are other problems which are not yet addressed in the literature and have remain as open research questions. Random placement and max-fit placement of the kernel are two possible strategies for placement of the kernel which may be superior to one another in terms of required amount of computation and specifications of the application.

Improper initial placement of the kernel may result in finding a background object instead of the object of interest. The kernel size, on the other hand is a crucial parameter in the performance of the Mean-shift algorithm. If the kernel size is chosen too large, it will contain too many background pixels. Kernels that are too small not only determine an incorrect position for the object of interest, but may also "roam" around on the object in a video sequence leading to poor object location (Figure 5.2). Changing the size of the kernel may also be necessary to cope with changes in the shape of the tracked object (e.g. rotation, moving toward or away from the camera).

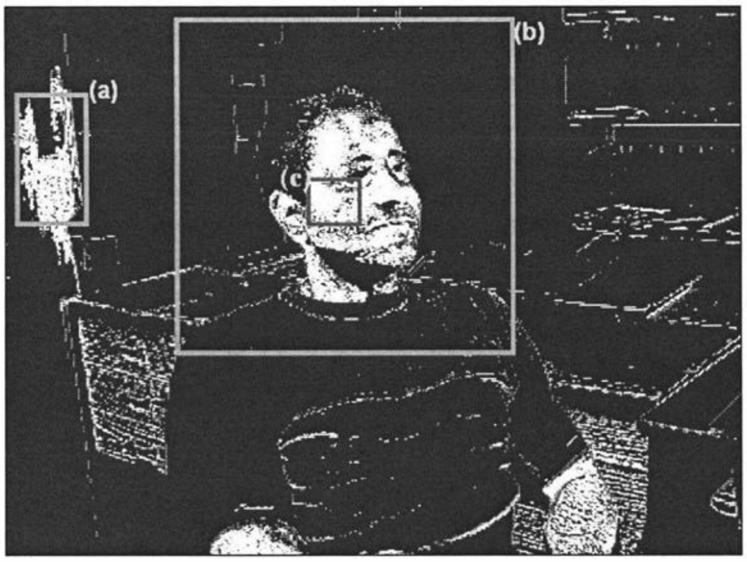


Figure 5.2. Choosing the size and initial placement of the kernel. (a) incorrect placement of the kernel, (b) choosing a kernel that is too large, (c) choosing a too small kernel.

The general approach to single object tracking using the Mean-shift algorithm is referred to as the *"Continuously Adaptive Mean-shift"* or the *CAM-Shift* algorithm. It was introduced by Bradski [110] in 1998. This is one of the earliest works in using the Mean-shift algorithm in object tracking. The CAM-Shift tracking algorithm which adapts the size of the search window to the object using the knowledge of the aspect-ratio of the desired object and the zeroth moment of the kernel is as follows (Algorithm 5.1).

Let ***M(p)*** represents the degree of membership of pixel ***p*** to the target object. Then for a kernel in quantized 2D space the coordinates of the centre of gravity are calculated as follows:

$$M_{00} = \sum_y \sum_x M(x, y) \qquad \textbf{(5.1)}$$

$$M_{01} = \sum_y \sum_x x \cdot M(x, y) \qquad \textbf{(5.2)}$$

$$M_{10} = \sum_y \sum_x y \cdot M(x, y) \qquad \textbf{(5.3)}$$

$$\mathbf{X_c = M_{01}/M_{00}} \qquad \textbf{(5.4)}$$

$$\mathbf{Y_c = M_{10}/M_{00}} \qquad \textbf{(5.5)}$$

$\boldsymbol{M_{00}}$, $\boldsymbol{M_{01}}$ and $\boldsymbol{M_{10}}$ are called the zeroth moment, the first moment-x and the first moment-y respectively.

```
Algorithm CAM-ShiftTracker()
Begin
   Kernel = InitializeKernel();
   For each frame Image$_i$
      Features = ExtractObjectFeatures( Image$_i$ );
      Repeat
         [$M_{00}$, $M_{01}$, $M_{10}$] = ComputeMoments( Kernel, Features );
         $C_w$ = Centre_of( Kernel );
         $C_g$ = ($M_{01}/M_{00}$, $M_{10}/M_{00}$);
         ShiftWindow( Kernel, $C_g - C_w$ );
         SetKernelSize( Kernel, w($M_{00}$), h($M_{00}$) );
      Until $|C_g - C_w| <= \varepsilon$;
   End For
End
```

Algorithm 5.1. The CAM-Shift algorithm.

5.2 Research Background

Comaniciu, Ramesh and Meer [138] in their survey on "Kernel-based Object Tracking", have indicated two major components in visual object tracking. Target Representation and Localization. Target representation is normally a bottom-up process which has to cope with changes in the appearance of the target. Filtering and Data Localization is mostly a top-down process dealing with the dynamics of the tracked object, learning of scene priors, and evaluation of different hypotheses. The way the two components are combined and weighted is application dependent and plays a decisive role in the robustness and efficiency of the tracker. Some of the applications of the Mean-shift algorithm as a general tool for analyzing the feature space are introduced in Comaniciu [139]. Another successful application of the Mean-shift algorithm is object tracking with the core idea of representing the object as a set of features which may be varying in terms of number, distance and time [140]. This assumption is realistic in real-world applications of video analysis because a 100% accuracy in detecting features is not reachable.

Comaniciu [141] used the Mean-shift algorithm for object tracking with a moving camera. This meant that the feature extraction of the object is more difficult because of the changing of the background, but the basic idea of the tracking is the same as Bradski's [110] work. Allen [142] used the CAM-Shift algorithm for tracking of multiple color patches. Wang, Chen and Huang [143] used the features extracted from the wavelet to track the object.

5.2.1 The Mean-shift algorithm and feature tracking

In general, the Mean-shift algorithm can be applied as a feature tracking algorithm for video sequences. One of the features of this approach is continuous evaluation of the kernel which can be done using a distance evaluation function. Different distance functions have been introduced and applied in research. Depending on the features and the target application, one may be superior to the other. These functions may require a significant amount of computation.

For color-based object tracking also different similarity measurements have been introduced in the literature. The Euclidean distance, Mahalanobis distance [110], the Bhattacharayya coefficient [144] and the Kullback-Liebler divergence are the most typically used similarity measurements.

Yang et al. [145] proposed a new similarity measurement for object tracking using the Mean-shift algorithm. They have shown that Bhattacharayya coefficient and Kullback-Liebler divergence are inaccurate in higher dimensions on Gaussian synthesized data. Alternatively they proposed another similarity measurement for two Gaussian distributions which is more accurate and reliable in higher dimensions. Instead of evaluating the information-theoretic measures from the estimated PDF (probability density function), they defined the similarity between two distributions as the expectation of the density estimates over the model or target image. Given two distributions with samples $I_x = \{x_i, u_i\}$. $i = 1..N$, and $I_y = \{y_j, v_j\}$, $j = 1..M$, where the center of sample points in the model is x, and the current centre of the target points is y, the similarity between I_x, I_y in the joint feature-spatial space is as follows:

$$J(I_x, I_y) = \frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M} w\left(\left|\frac{x_i - y_j}{\sigma}\right|^2\right) k\left(\left|\frac{u_i - v_j}{h}\right|^2\right) \quad (5.6)$$

k(x) is a RBF (Radial Basis Function) kernel function, and *w*, *h* are descriptors of the size of the kernel. They have shown that their approach for tracking of objects with known size performs better than Bhattacharayya coefficient and Kullback-Liebler divergence distance measurements.

Xu et al. [146] proposed a color-based object tracking method using the Mean-shift algorithm based on a user selected window in the initial frame which should contain the object of interest. They used the Bhattacharayya coefficient as the convergence measurement of the Mean-shift algorithm. Their color model is a bitwise method for YUV 32 bit color space which contains 32 color clusters. To tolerate initial incorrect manual selections, they have used the Epanichnekiv kernel smoothing function to give higher value to the pixels located in the centre of the selected area and lower value to the pixels of the background that may be located on the edges of the selected area. However, they have not shown that their color model is better than other color models.

Finally the Yang et al. [145] approach introduced the size and shape of the kernel as one of the features. This is in fact not suitable for some movement directions e.g. moving toward or away from the camera, which result a significant change in size, or for tracking an articulate object like hand where the shape is changing over time. These shortcomings make this approach not suitable for hand tracking and gesture recognition.

Using a higher number of dimensions which is the result of using different features at the same time for tracking is one of the current requirements of tracking. Research shows that more feature dimensions results in more accurate classification [39]. For instance, considering motions features, wavelets and color, together may improve the tracking accuracy. On the other hand, the feature selection itself and the amount of computation are the disadvantages of multiple feature space.

5.2.2 The Mean-shift algorithm and variable sized kernel

Choosing the size of the kernel is one of the challenges with the Mean-shift algorithm. As mentioned in Section 5.1.2, kernel resizing in the CAM-shift algorithm is based on the density of the features within the kernel and the known aspect-ratio of the object. This approach is not robust, and therefore alternative methods are still being studied.

Setting the kernel size in the CAM-Shift algorithm [110] is based on several assumptions: i) the quality of the detection together with the solid shape of the object of interest provides a certain density of the pixels within the kernel, ii) the aspect ratio of the size of the object of interest (e.g. face) is always constant, and iii) the face silhouette is the only silhouette in the image. Based on these assumptions, Bradski has proposed the following equation for calculating the kernel size in each frame:

$$w=2\sqrt{\frac{M_{00}}{256}} \qquad \textbf{(5.7)}$$

$$h=1.2\times w \qquad \textbf{(5.8)}$$

where w and h are the width and the height of the new kernel and M_{00} is the zeroth moment of the kernel. We refer to this method as the "*kernel density-based*" method or for the sake of simplicity, *sqrt(m00)*, because the kernel size is based on the square root of the kernel density. Sherrah and Gong [147] proposed a similar method for tracking discontinuous motion of multiple occluding body parts of an individual from a single 2D view. They used the following width and height for estimating the kernel size for face tracking which is basically the same as Bradski's approach:

$$w = \sqrt{n_{skin}} \qquad \textbf{(5.9)}$$

$$h = 1.2 \times w \qquad \textbf{(5.10)}$$

where n is the number of non-zero pixels inside the kernel.

The resizing method of the kernel in Comaniciu, Ramesh and Meer [141] is based on measuring the kernel density by three different kernels simultaneously applied on the image. The three kernels consist of one kernel with the current size and the other two with window sizes of plus and minus 10 percent of the current size. For each, the color distribution of the kernel is compared to the color distribution of the target using the Bhattacharayya coefficient and the most similar distribution is chosen as the new current scale. One of the disadvantages of the Bhattacharayya coefficient is that in a uniformly colored region, any location of a window that is too small will yield the same value of the window as would be obtained by shrinking the window even more [148]. Therefore, decision making about shrinking or enlarging the kernel is impossible.

KaewTraKulPong and Bowden [76] proposed a method for tracking low-resolution moving objects in an image using color, shape and motion information. The height and width of the kernel were modeled by a white noise velocity model based on the assumption that the bounding box does not change dramatically. It should be noted that this assumption cannot be made in some applications including face tracking in HCI.

Yang, Duraiswami and Davis [145] proposed using Gaussian transform to model the kernel to decrease the number of iterations in the Mean-shift algorithm for tracking a moving blob. This enhancement requires some extra computation for pre-processing which degrades the overall performance of the algorithm. Collins [148] points out that setting the window size based on the CAM-Shift algorithm with negative weights does not produce a reliable estimation of the object boundaries. Alternatively he proposes a method for resizing the search window based on the weight of the samples and the scale of the Mean-shift kernel. The scale space is generated by convolving a filter bank of spatial *Difference Of Gaussian* (DOG) filters with a sample weight image. The results are then convolved with an Epanichikov kernel in the scale dimension. Although this method produces more reliable results for a wider range of applications, the computational expense of calculating convolutions is very high. This makes the approach unfavorable for real-time applications.

Shan et al. [149] proposed a method for skin tracking using the Mean-shift algorithm based on particle filtering called the Mean-shift Embedded Particle Filter (MSEPF). They have introduced the in-motion skin pixels as an extra weigh for the tracking using the Mean-shift algorithm. Introducing the motion feature as another weight factor makes the tracking algorithm more sensitive to moving hand or face. On the other hand due to the motion tracking method the detected centre of gravity may be varying after applying the Mean-shift algorithm. For instance, using frame subtraction, the in-motion pixels are mostly located on the edges of the hand and face. This makes the tracked rectangle to move biased to the edges of the hand silhouette. They have reported that this algorithm is being used for giving simple hand commands to an intelligent wheelchair as visual input.

5.3 Fuzzy-based kernel resizing

Our observations show that selecting a large kernel size equal to the input image together with resizing it with a proper algorithm can find the biggest blob in the image which would normally be the face or the hand region in an HCI application.

5.3.1 Initialization and boundary detection of the kernel

Based on the output of the skin color segmentation algorithm, we observed the following limitations of the skin detection:

1) The quality of detection is not deterministic and may vary over time.
2) The quality of detection may be inhomogeneous on the same blob. For instance the density of the detected pixels in the forehead area may be different from the density of the detected pixels in the chin area (Figure 5.3).

3) The density of the falsely detected pixels for some colors (e.g. wood color) may be considerable in comparison to the density of the pixels of the object of interest and therefore cause incorrect tracking.

4) The shape of the silhouette of the object of interest (e.g. hand) is non-deterministic and therefore the size of the kernel cannot be interpreted using the zeroth moment information.

5) The blob of the object of interest may be disconnected due to different lighting condition.

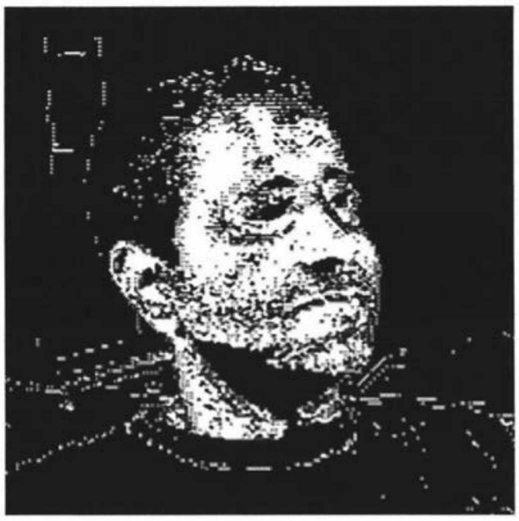


Figure 5.3. Inhomogeneous detection of the skin pixels in the image. The density of the pixels are varying in the face area which makes the detection and tracking of the face blob more difficult.

To overcome the limitations discussed, we used a kernel initially equal to the size of the input image and resized it based on the density of the edges of the kernel. The results of applying this method in different scenarios using Algorithm 5.2 were superior to the CAM-Shift algorithm.

```
Algorithm SkinBlobTracker()
Begin
   Kernel = MaximumSize();

   For each frame Image$_i$
      For each pixel P$_{x,y}$ in Image$_i$
         If LowerSkinThreshold ≤ Hue(P$_{x,y}$) ≤ UpperSkinThreshold
             Feature[x, y] = 1;
         Else
             Feature[x, y] = 0;
         End if
      End For

      For each D = boundary B$_i$ of the Kernel
          If (D < 10%) then
               Shrink the kernel from boundary B$_i$
          Elseif (D > 15%) then
               Enlarge the kernel from boundary B$_i$
          End if
      End For

      Repeat
         [M$_{00}$, M$_{01}$, M$_{10}$] = ComputeMoments( Kernel, Features );

         C$_w$ = Centre_of( Kernel );
         C$_g$ = (M$_{01}$/M$_{00}$, M$_{10}$/M$_{00}$);
         ShiftWindow( Kernel, C$_g$ - C$_w$ );
      Until |C$_g$ - C$_w$| <= $\varepsilon$;
   End For
End
```

Algorithm 5.2. Blob tracker based on the Mean-shift algorithm and evaluating edges of the kernel.

The primary comparisons of the proposed algorithm to the CAM-Shift algorithm show that in tracking of a single object both algorithms have advantages and disadvantages. The convergence speed of the CAM-Shift algorithm is faster than the proposed algorithm because the kernel size changes exponentially to the inner density of the kernel in comparison to the proposed algorithm in which the kernel size changes linearly. To improve the convergence speed of the proposed algorithm we have proposed a fuzzy approach to resizing the kernel based on the edge density of the kernel which is described in the following section.

5.3.2 Fuzzy boundary detector

The fuzzy function for resizing the kernel includes three functions as described in the following paragraphs.

Fuzzifier: The fuzzifier, changes the input values to fuzzy values. We have considered 3 input fuzzy values. The boundaries of these values are based on the empirical results that we found by trial and error in face tracking using a Dragonfly digital camera in separate experiments. Empirical results show that the proposed number of levels is sufficient for face and hand tracking (Figure 5.4).

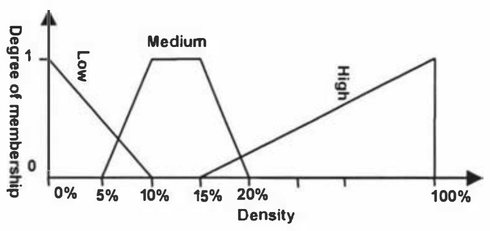


Figure 5.4. Fuzzy values for the inputs of the fuzzy boundary detector

Inference Engine. The inference procedure for the proposed fuzzy controller is presented in Table 1. We have considered three fuzzy values for changing the output as indicated in Figure 5.5. The values -5 and 5 are arbitrary and determine the shrinking and enlarging speed of the algorithm.

Table 5.1. Fuzzy controller for the fuzzy boundary detector

Edge density (input)	Resize Rate (output)
Low	Shrink
Medium	No Change
High	Enlarge

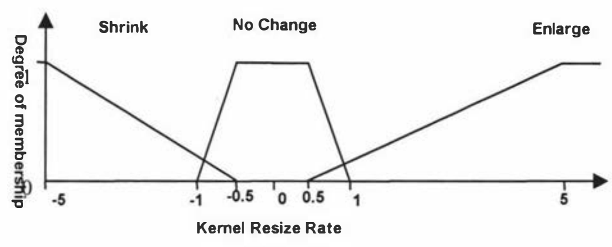


Figure 5.5. Fuzzy outputs for the fuzzy boundary detector

Defuzzifier: For converting the fuzzy outputs to numerical values, the centre of gravity method was used.

The algorithm for the Mean-shift face tracker with fuzzy boundary detection is as follows:

```
Algorithm FuzzyBlobTracker()
Begin
   Kernel = MaximumSize();

   For each frame Image$_i$
      For each pixel P$_{x,y}$ in Image$_i$
         If LowerSkinThreshold ≤ Hue(P$_{x,y}$) ≤ UpperSkinThreshold
             Feature[x, y] = 1;
         Else
             Feature[x, y] = 0;
      End For

      For each boundary B$_i$ of the Kernel
         Delta = FuzzyBoundaryDecision(B$_i$);
         ChangeBoundarySize( Kernel, B$_i$, Delta );
      End For

      Repeat
         [M$_{00}$, M$_{01}$, M$_{10}$] = ComputeMoments( Kernel, Features );

         C$_w$ = Centre_of( Kernel );
         C$_g$ = (M$_{01}$/M$_{00}$, M$_{10}$/M$_{00}$);
         ShiftWindow( Kernel, C$_g$ - C$_w$ );
      Until |C$_g$ - C$_w$| <= $\varepsilon$;
   End For
End
```

Algorithm 5.3. Fuzzy blob tracker based on the Mean-shift algorithm.

5.4 Experiments and Results

In this section, we present the experiments and the results showing the behavior of our proposed algorithm in comparison to the CAM-Shift algorithm in noiseless and noisy environment. The implementation of the CAM-Shift algorithm is based on estimating the size of the kernel as a function of the zeroth moment as described in Bradski [110] and Sherrah [147]. We call this method sqrt(m00), and compare it to our method which we have called the "fuzzy method". We measured (a) the top-left corner position, (b) the position of the centre of gravity, (c) the area of the kernel and (d) the error in positioning the kernel, which collectively give an overview of the stability and correctness of the kernel.

We also applied preset levels of white noise in each experiment. The noise might change the value of a skin pixel causing it to not appear as a skin pixel in the silhouette image. Seven levels of noise (0 to 30% in 5% steps) were tested on the input sequence and all of the experiment's parameters were measured in each experiment. The video sequences were recorded using a Dragonfly video camera equipped with a Sony CCD. Each frame was a 640x480 RGB color image recorded in 15 fps.

The noise was applied before skin color detection which means the number of missing skin pixels were increased by performing the Erode-Dilate morphological operators inside the skin detector. We believe this model is more realistic and more similar to the real world conditions.

To compare the algorithms we used an ideal tracker as the ground-truth data for evaluation. The ideal tracker is a tracker which holds at least 10% of the filled pixels in each edge, and tracks the object from an ideal (no noise) skin detection algorithm. The ideal tracker finds this characteristic for the edges in each frame, which makes its behavior more like the inner density based algorithm which has faster convergence speed. In addition, it makes it to be flexible in resizing and to match properly around the shape of the desired object. Results from four experiments are presented in the following sections[17].

5.4.1 Experiment 1: Detecting a blob with no movement

The platform for the first experiment was a manually segmented face image used as the input video sequence. The purpose of using a fixed frame instead of a real video sequence of an in motion blob was to test the algorithms with ideal input. We also wanted to have the ability to control the noise level without having to deal with other variables such as camera noise and change in lighting condition. The results of tracking in a noiseless environment in both of the algorithms were almost the same (Figure 5.6).

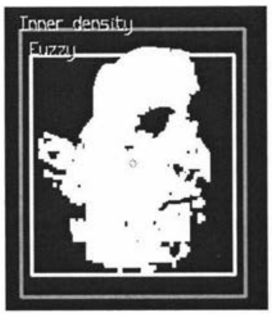


Figure 5.6. Detected boundary by the two algorithms, surrounding a static image silhouette

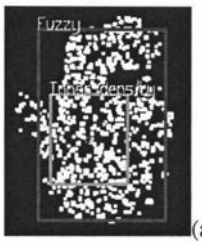
[17] The video sequence is available online at http://www.massey.ac.nz/~fdadgost

5.4.1.1 Behavior of the kernel density-based algorithm in noisy environment

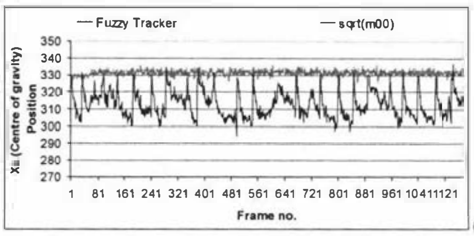
By increasing the noise, the reliability of the *sqrt(m00)* method decreased and it finally couldn't properly specify the boundaries of the face. However, there is a logical explanation for this behavior. This method depends on the density of the kernel to specify the boundaries. Adding noise to the kernel causes a decrease in the density by a power of two, and therefore smaller width and height for the kernel are computed. By increasing the noise level, this method starts to loose the track more frequently (Figure 5.7b).

5.4.1.2 Behavior of the fuzzy boundary detector algorithm in noisy environment

The fuzzy boundary detection algorithm proposed here shows more robustness against white noise. After locating the approximate position of the object, it examines the density of the edge of the kernel instead of the inner density. This feature makes it more robust against change of the density inside the kernel. In addition, the fuzzy behavior applied on this algorithm, makes resizing smoother and less sensitive to noise. With the highest noise level, the fuzzy-based approach demonstrates significant stability in comparison to the inner density algorithm. Figure 5.7a shows the boundary detected by the fuzzy-based approach and the density based approach. The position of the kernel using fuzzy-based approach was correct and stable in almost the whole period of tracking, while position of the kernel using the density-based approach was unstable. Figure 5.7b shows the occurrence of the "roaming effect" for sqrt(m00) while the fuzzy edge density method is stable.



(a)



(b)

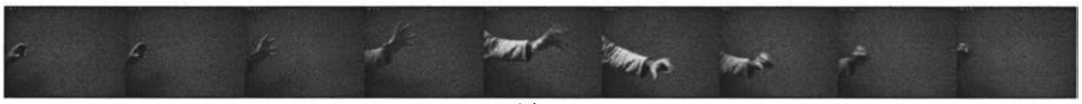
Figure 5.7. a) The correct detection determined by Edge density-fuzzy. The smaller rectangle is the result of kernel density-based – sqrt(m00) – method. b) Behavior of the algorithms with a noise level of 15%.

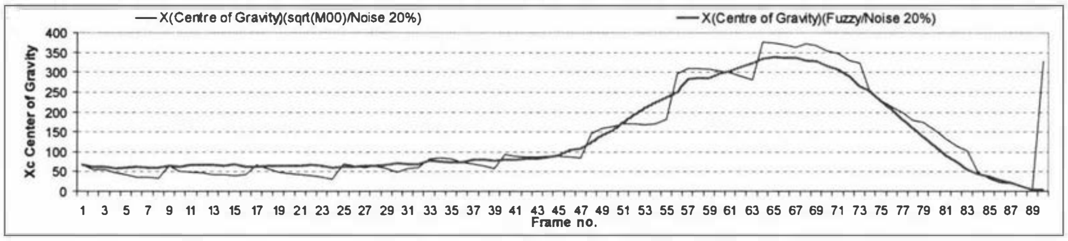
In the following experiments we present the behavior of these algorithms in different scenarios, like occluded blobs, changing shape and zoom effect.

5.4.2 Experiment 2: Tracking the blob of a moving hand

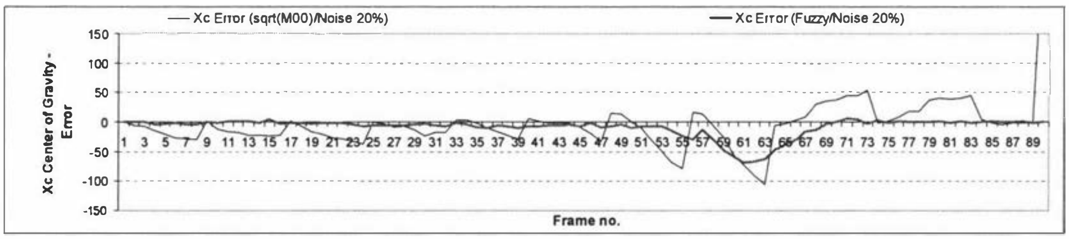
The second experiment was performed on the video sequence of a hand, demonstrating a grabbing gesture. The characteristic of this gesture is a continuous change of the shape, and fast movement of the position of the blob as shown in Figure 5.8.

Figure 5.8b and c, present position of the centre of gravity of the trackers and the measured error, respectively. Although both trackers are able to follow the object of interest, the tracker based on inner density shows fluctuating error in boundary detection, while the fuzzy tracker is more stable and more accurate in detection of boundaries.


(a)



(b)



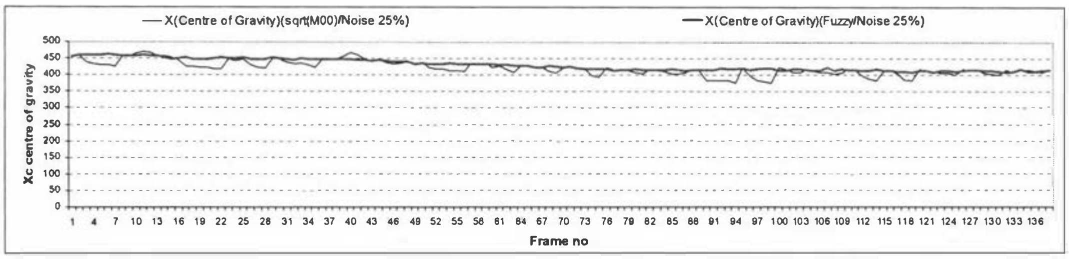
(c)

Figure 5.8. Tracking the hand in a "grabbing" hand gesture with white noise 20%, a) Original image sequence, b) Centre of gravity, c) Error of displacement in comparison to an ideal tracker (X_c-$X_{c\ ideal}$).

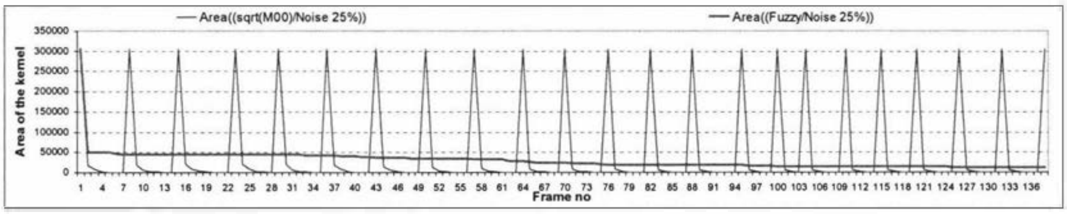
5.4.3 Experiment 3: Tracking an object moving away from the camera

The third experiment was performed on the video sequence of a face, moving away from the camera. This experiment was designed to analyze the behavior of the tracking algorithm on determining the boundaries of an object continuously shrinking. Both the size and the centre of gravity of the kernel change during the sequence (Figure 5.9). The roaming effect for the inner density tracker is observable in Figure 5.9b & c, representing fluctuation of the tracker in tracking the object of interest.

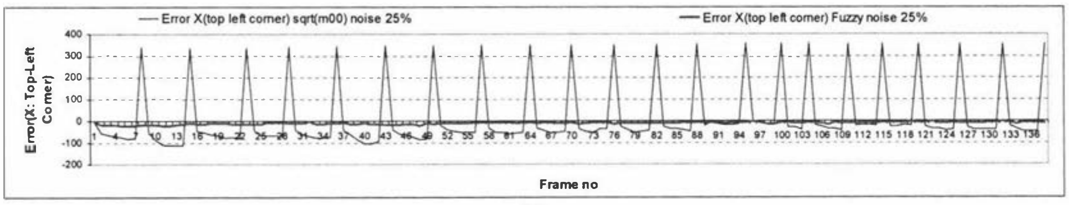
(a)



(b)



(c)



(d)

Figure 5.9. Tracking the centre of gravity – zoom out with 25% white noise, a) Original image sequence, b) Xc centre of gravity, c) Area of the kernel, and d) Error of placement of the kernel in comparison to an ideal tracker (X-X_{ideal}).

5.4.4 Experiment 4: Tracking a moving hand in occluded situation

The forth experiment was a simple hand movement in presence of another hand, as shown in Figure 5.10. This video sequence has two main characteristics. Firstly the presence of the "other hand" causes occlusion for the tracking algorithm, especially while the hands overlap. The tracking algorithms cannot distinguish which one is the hand in front of the camera. Secondly, the movement is a rotation of the hand around the elbow, which causes the rectangular boundaries of the hand to change in time.

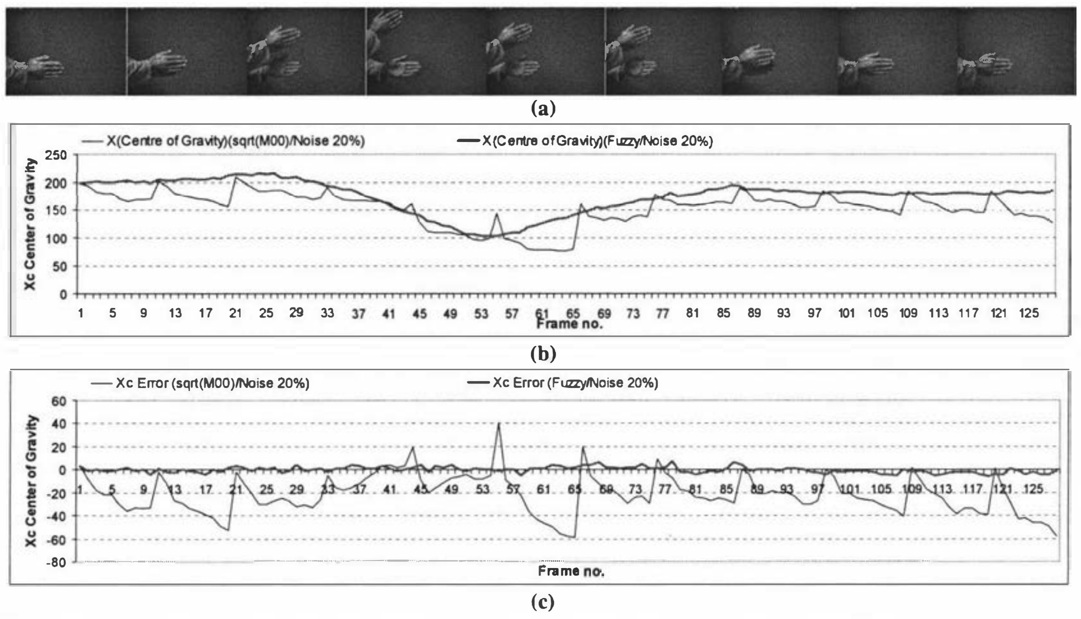


Figure 5.10. Moving hand, in occluded situation, a) the original image sequence, b) Xc of the centre of gravity in noise 20%, d) Error in Xc of the kernel in comparison to an ideal tracker (X-X$_{ideal}$).

5.4.5 Accuracy of tracking

Evaluating a general purpose tracker is not a straight forward procedure because the acceptable accuracy and error tolerance are dependent on the application. For instance, considering an acceptable error tolerance for the position of the kernel in experiment 3, the accuracy of the sqrt(M00) dramatically decreases by limiting the error tolerance, while the fuzzy algorithm shows robustness against white noise and high accuracy as shown in Table 5.2.

Table 5.2. Comparison of accuracy of the algorithms

Experiment 3	**Accuracy**	
Acceptable error (pixel)	**Sqrt(M00) Noise 20%**	**Fuzzy Noise 20%**
1	**0.094**	**0.608**
2	**0.123**	**0.992**
5	**0.181**	**1**
10	**0.246**	**1**
15	**0.304**	**1**
20	**0.405**	**1**
25	**0.507**	**1**

Another distance measurement between two segments introduced by Gevers [113] is based the sum of distances of all of the points in one shape to another shape called the Mean-distance.

We used the concept of the Mean-distance to evaluate the distance between an arbitrary tracker and an ideal tracker, as discussed in the following paragraphs.

Let A denote an arbitrary shape in the 2D space in discrete Cartesian system, containing *n* points. Then the shape *A* can be defined as:

$$A = \{u_i\}_{i=1..n} \qquad \textbf{(5.11)}$$

Then, the distance between an arbitrary vector *x* and shape *A* is defined as:

$$d(\vec{x}, A) = \min[d(\vec{x}, \vec{u})] \qquad \textbf{(5.12)}$$

Now, using the definition of the distance between two points in the 2D Cartesian system, the distance between x and u, is defined as:

$$d(\vec{x}, \vec{u}) = \sqrt{(v_x - u_x)^2 + (v_y - u_y)^2} \qquad \textbf{(5.13)}$$

Finally the distance between two shapes A and B in 2D space is defined as:

$$d(A,B) = \sum_{\vec{u} \in A} d(\vec{u}, B) + \sum_{\vec{v} \in B} d(\vec{v}, A) \qquad \textbf{(5.14)}$$

Based on the definition and above equation, the distance between an inner point of a shape and the shape itself is zero. This means that the distance between a tracker and another which surrounds that tracker is zero as well. To penalize these cases, we used both of the distances as shown in the above equation. The result of the Mean-distance is presented in Figure 5.11.

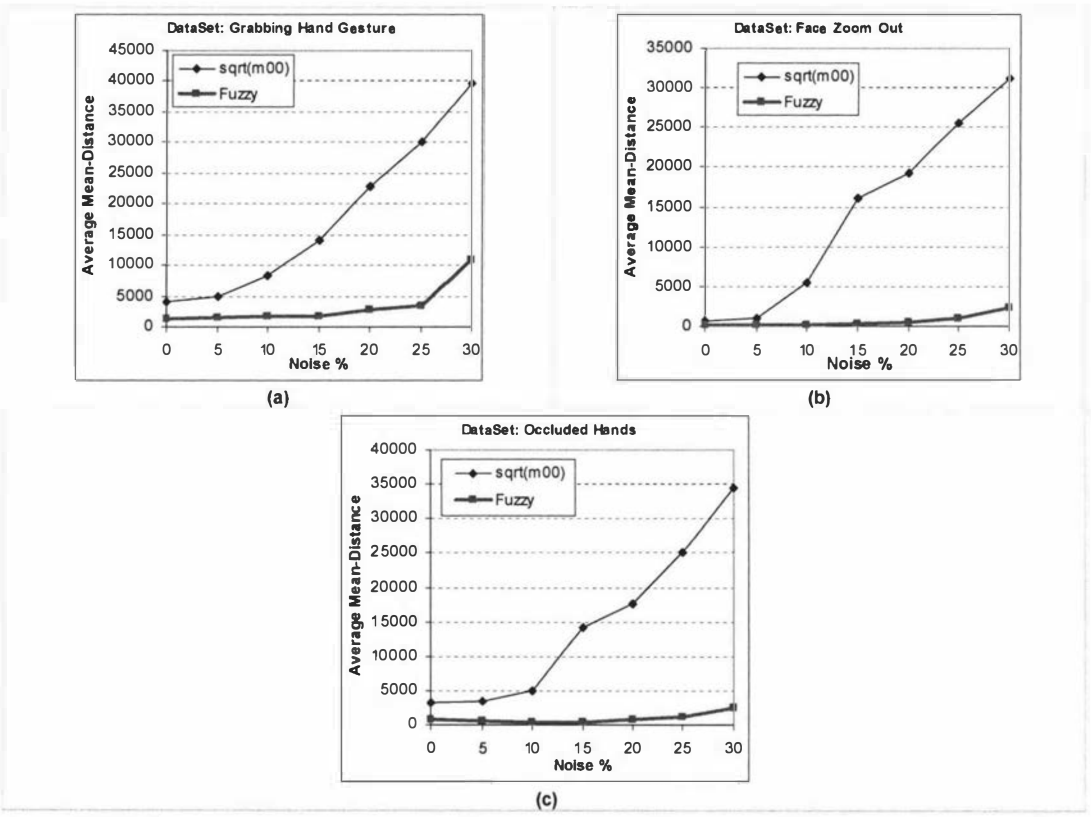


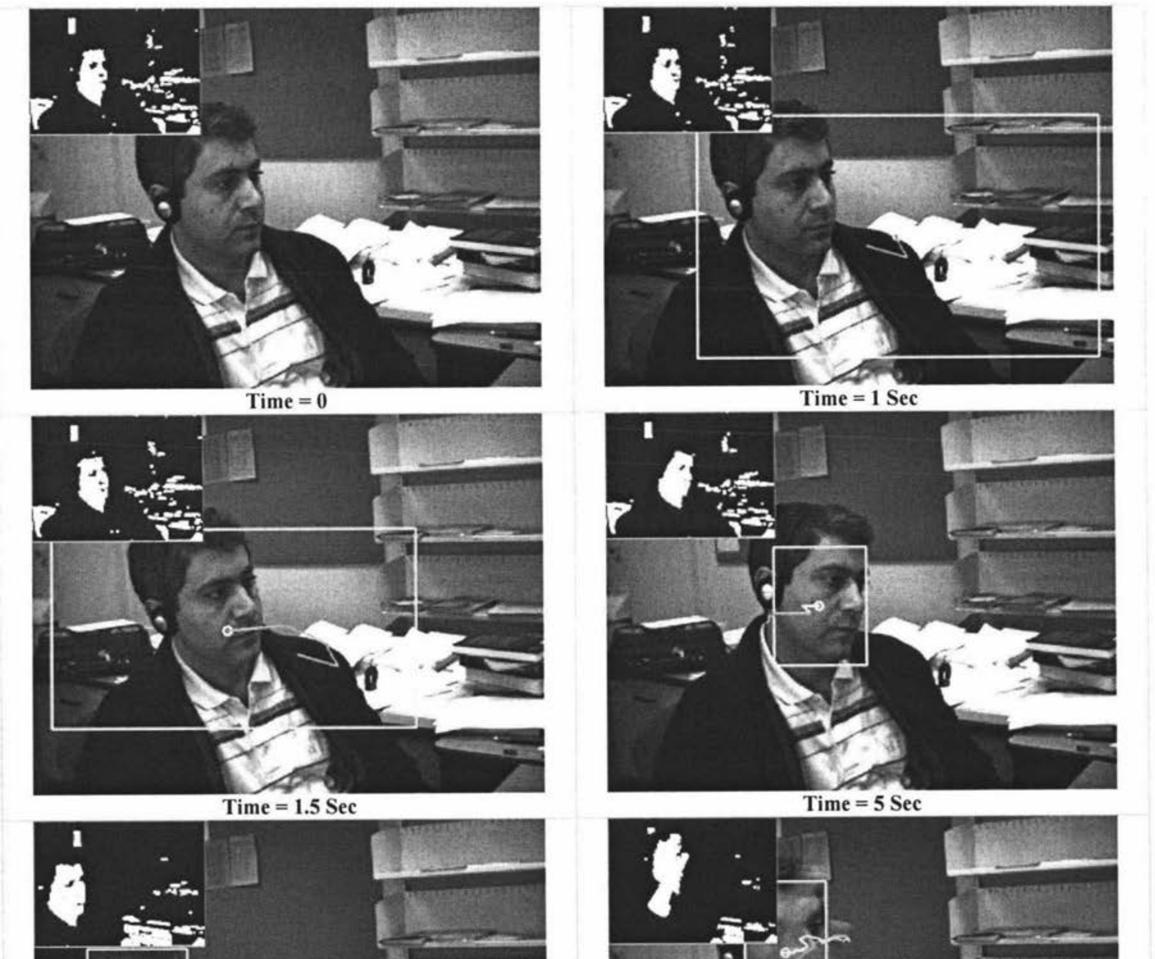
Figure 5.11. The average Mean-distance between the trackers and the ideal tracker in noise level 0% to 30%. a) "Grabbing hand gesture dataset" (experiment 2) , b) "Face zoom out" dataset (experiment 3) , and c) "Occluded hands" dataset (experiment 4).

5.5 Implementation issues for hand and face tracking

Based on the proposed algorithm, we implemented an application for tracking multiple objects including face and hands. In the following sections we describe the details of the implementation and demonstrate the results.

5.5.1 The multi-tracker implementation

The Mean-shift tracker introduced in this chapter was designed for tracking a single object silhouette within the image (Figure 5.12). While for HCI applications, tracking the hands and face of the user is required, capability of tracking multiple silhouettes is an implementation challenge using this algorithm.



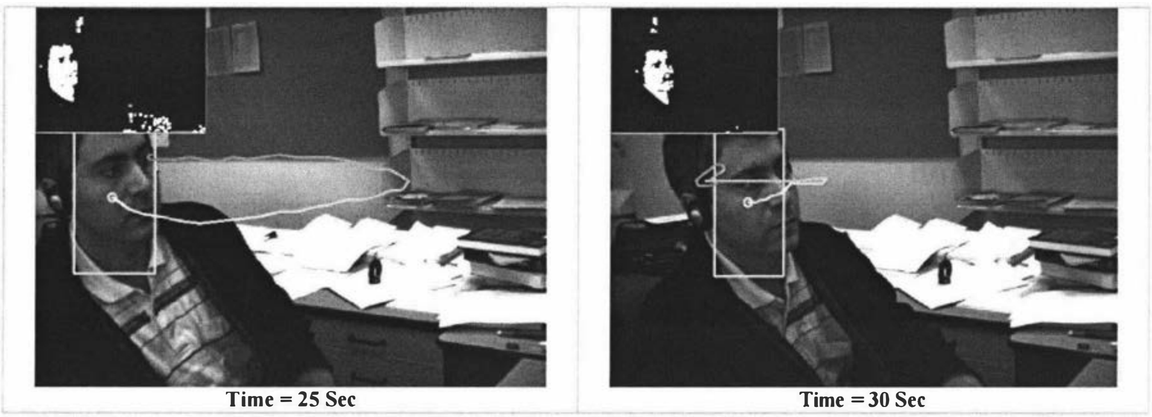


Figure 5.12. Face tracking using the proposed algorithm for skin detection, and the Mean-shift algorithm for blob tracking[18].

Since running multiple instances of the same algorithm on the same silhouette will produce the same tracking result, a strategy for distinguishing the different instances should be considered. For this purpose, we implemented an algorithm for running multiple instances of the tracker algorithm. This implementation has two main components: a) *tracker manager* and b) *tracker scheduler.*

The *tracker manager* scans the image and searches for the blocks containing a certain density of the skin pixels within the frame. For each block satisfying this constraint, it assigns a tracker to it. In the next stage the tracker scheduler runs the available instances of the tracker, and erases the content of the image silhouette after each individual tracker detected its boundaries. This strategy prevents the trackers to move to the detected area of each other. If the tracker scheduler recognized that a certain tracker does not carry a certain number of skin pixels, marks it for deletion in the next scan.

[18] Please refer to video#3 in the enclosed CD to view the full video

The result of running this algorithm on an image sequence is presented in Figure 5.13. In Figure 5.13, the green rectangles represent the trackers which just initialized by the tracker manager, and the yellow rectangles represent those trackers which lasted more than one frame. As seen in Figure 5.13d, there is no tracker on some of the non-skin areas, as initially were. This is the effect of the adaptive skin algorithm described in Chapter 1.

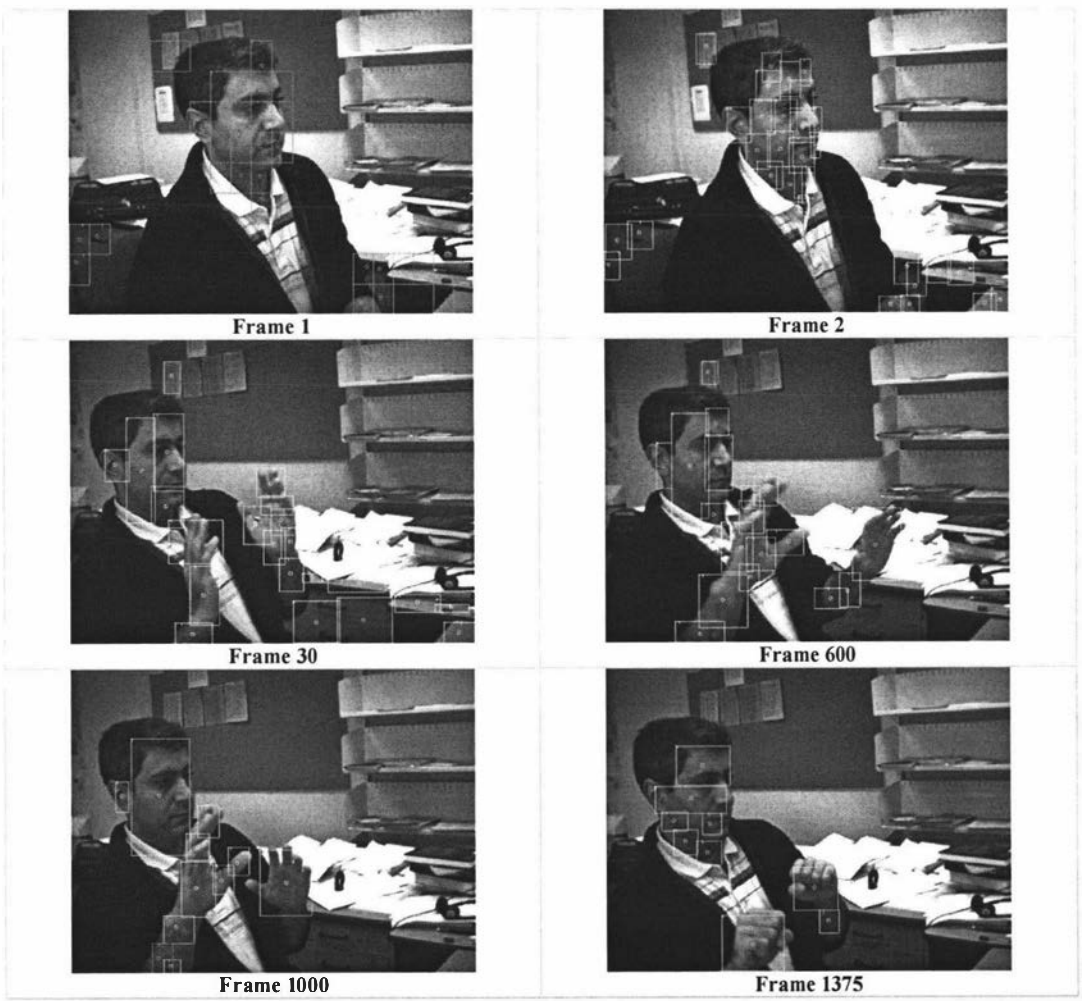


Figure 5.13. Running multiple trackers on the image sequence of hand movement based on the algorithm described in Section 5.5.1.

5.5.2 Tracker-tracker implementation

The result of the implementation in the previous section is a set of trackers, tracking the skin area. Although using a good quality silhouette there will be one tracker per visible object in the image, in some scenarios there will be more than one tracker per object. The silhouette of the hand palm with open fingers near the camera is one of those scenarios. To overcome this limitation, we implemented an extension of the introduced multi-tracker algorithm. This implementation is based on two sets of trackers. The first set consists of the trackers which track the blobs of the skin pixels. We call this set "the micro trackers". The second set is the trackers which track groups of trackers in the first set. We call this set "the macro trackers". The more precise membership function and definition are as follows:

Let $M = \{M_1,\ldots, M_n\}$ represent the set of macro trackers, and $m = \{m_1,\ldots,m_k\}$, represent the set of micro trackers. Each macro tracker carries a collection of micro trackers as $M_i = \{m_p \mid m_p \in m\}$, such that for each pair of macro trackers (i, j): $M_i \cap M_j = \emptyset$.

```
Algorithm TrackerOfTrackers()
        $M = \{\}$,  $m = \{m_1 \ldots m_k\}$
        For each $m_i$ in m do
                Found := false
                For each $M_j$ in M do
                        If distance($m_i$, $M_j$) < delta then
                                $M_j = M_j + \{m_i\}$
                                Found := true; break
                        End If
                End for each
                If not Found then
                        $M = M + \{\{m_i\}\}$
                End If
        End for each
End Algorithm
```

Algorithm 5.4. Tracker of trackers algorithm.

The result of running this algorithm on an image sequence containing different hand movements is presented in Figure 5.14.

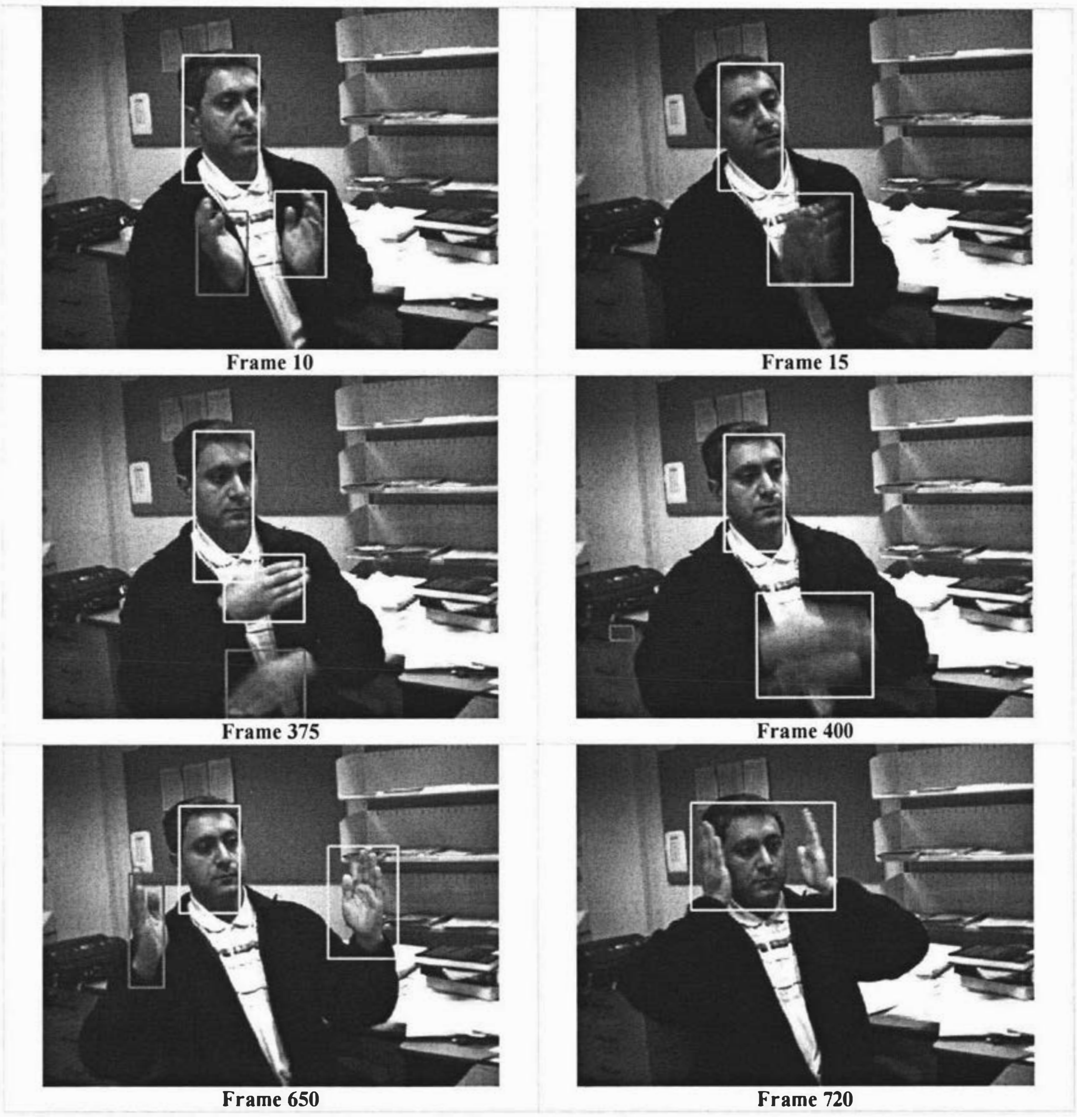


Figure 5.14. Robust hand and face tracking using the tracker-tracker algorithm on an image sequence of different bi-hand movements.

5.5.3 Using depth information for hand and face tracking

One of the limitations of the Mean-shift tracking algorithm together with the skin detection is occlusion handling. Because there is no significant information about the boundaries of the hand and face in the skin blobs, using an extra cue to distinguish the boundaries of the hand and face may be useful. In this section we describe the technique of using the depth information together with the Mean-shift algorithm. The depth information is extractable using a stereo-vision system.

A stereo-vision system can provide the depth information of the scene in terms of distance to the camera. This information can be used for enhancing the algorithms which we introduced in Chapters 3 and 4.

The simplest configuration of a stereo-vision system is bi-camera. However, three or more cameras also may be used in a stereo-vision system. Stereo-vision can produce a dense disparity map which can be translated to the depth information map. The resultant disparity map should be smooth, detailed and continuous. Moreover, surfaces should produce a region of smooth disparity values with their boundaries precisely delineated while small surface elements should be detected as separately distinguished regions. Unfortunately, satisfying all of these requirements simultaneously is not achievable. Algorithms that can produce a smooth disparity map tend to miss the details and those that can produce a detailed map tend to be noisy. The depth maps obtained by bi-camera stereo systems are not very accurate and reliable. A higher number of cameras may gain better quality depth information [150].

While the luxury of stereo-machines with more than two cameras is not yet commonly available, the normal bi-camera stereo-vision systems are the most available choice. However, the depth estimation ability is somewhat limited. We used a Bumblebee[19] bi-camera stereo-vision system for applying the techniques proposed in this section. Connected to the fire-wire port, this camera is able to record two 1024x768 color or gray scale images in 25 fps. The test platform was Windows XP and the programming platform was C++ in Visual Studio.Net 2003. Although the depth map obtained using this camera is not highly accurate (Figure 5.15), it can provide an estimation of the distance of different objects to the camera which was sufficient for evaluation of our ideas.

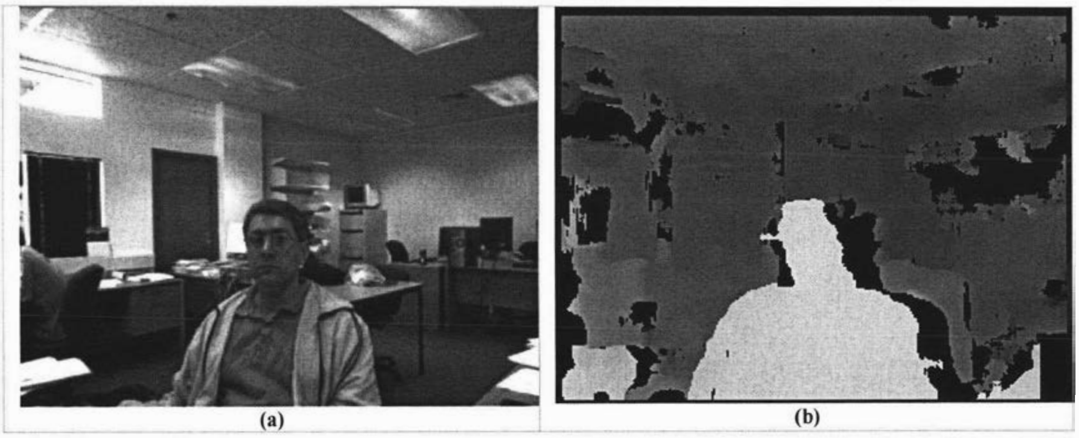


Figure 5.15. a) A sample image, b) the depth information of the sample image. The light area is the object closer to the camera, the black patches are of unknown depth.

5.5.3.1 The depth information and the adaptive skin detection algorithm

Let's take a step back and review the concept of motion detection for our adaptive skin detection algorithm. In fact, one of the reasons for the motion detection is separating the user and the background. Considering the fact that in an HCI environment the user is closer to the camera than the background, the depth information can significantly improve the accuracy of background elimination from

[19] Bumblebee stereo-vision system is manufactured by PointGrey research labs.

the image. This idea can be implemented using depth thresholding. Figure 5.16b represents the eliminated background and the remaining foreground after applying the depth thresholding technique.

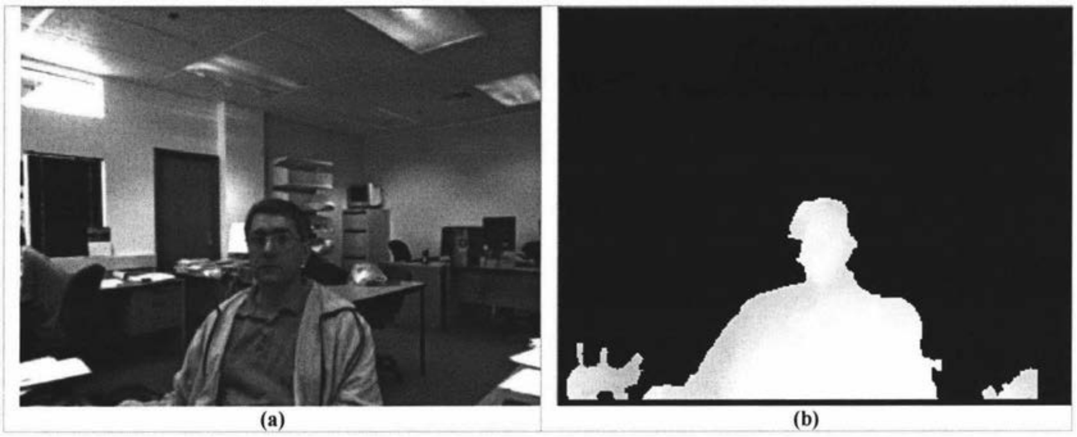


Figure 5.16. Background elimination using the depth thresholding technique.

The detected skin pixels within the separated foreground are more likely to be the actual skin color and therefore the adaptive skin detection algorithm will provide more accurate detection. Some of the unwanted areas like the surface of the table behind the user will also automatically be eliminated.

5.5.3.2 The depth information and the Fuzzy Mean-shift blob tracker

The depth information can be used for occlusion resolution in some scenarios. Figure 5.17 presents two cases in which the distance of the hand and face to the camera are different. Obviously, without the depth information the hand and face blobs in these images are being considered as one single blob. In this section, we describe how the depth information can be applied for enhancing the Fuzzy Mean-shift blob tracker algorithm for occlusion prevention.

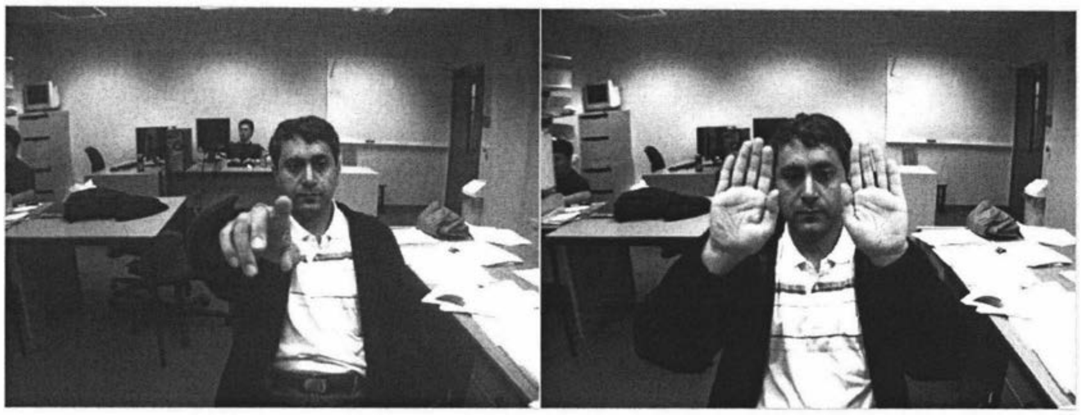


Figure 5.17. Some of the scenarios in which depth information can be useful for occlusion resolution.

Based on the idea of applying the depth information as an extra cue for blob tracking within the Mean-shift algorithm, we implemented another variant of the Fuzzy Mean-shift algorithm. The basics of this variant in Kernel shifting is similar to its ancestor. However the kernel management is slightly different.

The kernel in this version of the algorithm in addition to the vertical and horizontal boundaries of the tracking blob also carries the minimum and the maximum depth of the points within. The shrinking and enlarging procedure on the boundaries of the kernel are based on the pixels with depths within ±10% of the kernel's min-max depth. The pixels which do not satisfy this constraint or their depth is unknown will be ignored. Therefore, blob trackers which are tracking blobs in different depth levels will not interfere with each other. Figure 5.18 represents the result of this algorithm on an image. We should note that without the depth information, the hand and the face blobs are considered as one connected blob. Using the depth information, the Mean-shift algorithm is able to stay around the blob which has the homogeneous depth.

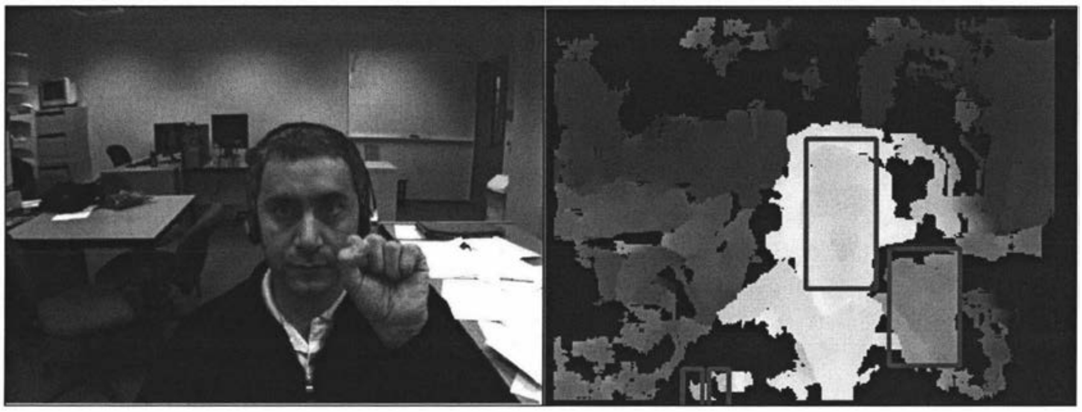


Figure 5.18. Occlusion resolution using the depth information and the Mean-shift algorithm. The tracked blobs are displayed on the disparity image (right).

5.6 Chapter summary

In this chapter, we presented a new approach for boundary detection in blob tracking based on the Mean-shift algorithm. Our approach was based on continuous sampling of the boundaries of the kernel and changing the size of the kernel using our novel algorithm. We also showed that the proposed method is superior in terms of robustness and stability compared to the density-based tracking method known as the CAM-Shift algorithm.

The robustness of our method against noise makes it a good candidate for use with cheap cameras and real-world vision-based HCI applications. This method is to be applied in conjunction with a fast pixel-based skin color segmentation algorithm as the level of noise and the quality of the skin detection are not deterministic.

Chapter 6. Modeling and recognition of gesture signals in 2D space

6.1 Introduction

A gesture is defined as "*the use of movements (especially of the hands) to communicate familiar or prearranged signals*"[20]. McNeill [95] one of the pioneers in behavioral science and gesture meaning, defines the term *gesture phrase* as "*the period of time between successive rests of the limbs*". He categorized gestures into three types: metaphoric, iconic and deictic which were discussed in detail in Chapter 3. Metaphoric gestures are more representational, but the concept they represent has no physical form; instead the form of the gesture comes from a common metaphor. An example is "the meeting went on and on" accompanied by a hand indicating rolling motion [96]. Iconic gestures, on the other hand, can convey meaning out of the context. These gestures represent information about such things as object attributes, actions, and spatial relations. Finally deictic gestures, also called pointing gestures, highlight objects, events, and locations in the environment. Deictic gestures have no

[20] http://wordnet.princeton.edu

particular meaning on their own, and convey information solely by connecting a communicator to a context.

A pointing gesture can be represented by the movement trajectory of hand or head within a certain period of time. The movement trajectory itself carries a large amount of information, including speed, direction, position and acceleration. A gesture recognition system in this context accepts the gesture information from some form of a data acquisition device and recognizes a specific gesture pattern within the data. From this point of view, gesture recognition requires analysis of a sequence of information over time, to detect a gesture signal. Although there are some techniques which are more suitable for pattern recognition in time series, there is no commonly accepted approach for gesture recognition. The methods and techniques for gesture recognition are still evolving together with introduction of new hardware and user requirements. Hence, gesture recognition which requires feature analysis of time series is still an ongoing research.

In this context, adequate feature selection from gesture signal is an essential part of a gesture recognition system. Although a higher number of features and multiple cues, may result in a more accurate classifier for gesture recognition, a larger number of features to process makes a classifier more complex and also typically more computation intensive. Moreover, the data preparation for test and training could be an extensively hard and time consuming task. Therefore selecting a smaller set of features which are more descriptive is highly desirable.

In this chapter we have introduced a technique for modeling and representing gesture signals in 2D space. This technique is based on measuring the gradient of the movement trajectory in certain time distances, to represent the gesture signal. Using this technique, each gesture signal is represented as a time series of gradient values. These features can be classified by applying a supervised learning method.

6.2 Research background

Dietterich [151] asserted five different categories of methods for sequential supervised learning, 1) The sliding window method, 2) Recurrent sliding windows, 3) Hidden Markov Models (HMM) and related methods, 4) Conditional random fields, and 5) Graph transformer networks. Some of these methods have been successfully applied for specific applications such as hand written character recognition or automatic dictation correction. However, the advantages and disadvantages of these methods in gesture recognition are yet unknown. We presented a detailed research background on gesture recognition and HMM in Chapter 2. Here, in this section, our concentration is on feature selection from the gesture signal, and other gesture recognition techniques.

6.2.1 Time series analysis

Yang and Xu [1] present a method for developing a gesture based system using a multi-dimensional HMM and model gesture as sequential symbols. They have reported 99.7% accuracy in a prototype for recognition of hand written digits from 1 to 9. They have applied short time Fourier transform (STFT) for feature selection from the input [x, y] coordinates of the gesture movement.

Watnabe and Yachida [152, 153] proposed a method of gesture recognition from image sequences. The input image is segmented using maskable templates and then gesture space is constituted by Karhunen-Loeve (KL) expansion using the segment. They applied Eigen vector-based matching for gesture detection. Mantyla et al. [154] show that using acceleration sensors in a mobile device, some useful information for detecting static and dynamic gestures are extractable. They used an HMM for dynamic gesture detection and the self-organizing mapping scheme of Kohonen for static gesture detection.

Oka, Satio and Kioke [77, 78] proposed a gesture recognition based on measured finger trajectories for an augmented desk interface system. They used Kalman-Filter for predicting the location of multiple fingertips and HMM for gesture detection. They have reported average accuracy of 99.2% for single finger gestures produced by one person. Ogawara et al. [155] proposed a method of constructing a human task model by attention point (AP) analysis. Attention points relate and integrate multiple observations and construct a locally enhanced task model of human demonstration. Their AP analysis system consists of two steps. In the first step, action segments and APs are extracted. Then in the second step, by closely examining human demonstration only around APs, the system extracts the attribute values and improves the model. Their target application was gesture recognition for human-robot interaction.

New et al. [156] proposed a gesture recognition system for hand tracking and detecting the number of fingers being held up to control an external device, based on hand-shape template matching. Perrin et al. [157] described a finger tracking gesture recognition system based on a laser tracking mechanism which can be used in hand-held devices. They have used HMM for their gesture recognition system with an accuracy of 95% for 5 gesture symbols at a distance of 30cm to their device.

Because of the difficulty of data collection for training an HMM for temporal gesture recognition, the vocabularies are very limited. Furthermore, to reach to an acceptable accuracy, a great amount of data is required for training and a lot of time should be spent to estimate the parameters of HMM. Hence, some researchers have suggested to use a better approach for more complex systems [157]. However, this is still an open research issue.

6.2.2 Gesture identification through pattern recognition

The approaches to gesture recognition are not limited to gesture modeling as time series. Other approaches based on static pattern recognition such as Neural Networks and template matching are also being used for gesture-signal recognition.

Darrell and Pentland [158] in one of the first papers on detecting gestures through space-time, described their system for matching gesture templates rather than the feature sets. In this context, a gesture pattern is described as a set of views observed over time. Since each view is characterized by the outputs of the view models used in tracking, a gesture can be modeled as a set of model outputs. This sequence of model outputs is also referred to as a gesture signal. To match the gesture signals they have applied a signal-template which is determined by the training data and DTW (dynamic time warping) to resize the input samples to the normalized size of the gesture signal.

Zho, Ren, Xu and Lin [84] described a method for developing a real-time gesture controller which includes visual modeling, analysis, and recognition of continuous dynamic hand gestures. They used visual spatio-temporal features of gesture and DTW technique to match the gesture signals. For detecting each gesture, they have used a gesture template created from the min-max values of the training gesture signals. They have reported an average accuracy of 89.6% for 12 simple hand gestures.

Eisenstein, Ghandeharizadeh and Golubchik [159, 160] proposed a multi-layered framework for hand posture recognition. The lowest layer is hardware dependent, and the higher level layers are built on top of the lowest layer and are independent of the physical device. The lowest layer accepts a continuous stream of the raw data from the sensors on a glove input device. The second layer contains a set of predicates that describe hand postures. They have considered 22 hand postures in their models, to address the ASL alphabet which is the intended application. The third layer contains a set of templates, each which corresponds to a whole hand gesture. This layer is basically the description of a sequence of hand postures which form a simple hand gesture. They have defined a gesture as "A description of the changing posture and position of the hand over time". Finally, a letter of the alphabet is made from a set of hand gestures which were defined in the previous level. To detect hand postures from the input sensors, they have used thirty-seven neural networks using each network for a postural predicate. For each hand posture, output of a set of 4 to 10 sensors are fed to the neural net. To match the posture sequence to the gesture templates, they have used the Euclidean distance between the ANNs' output set and each template. The best match is chosen as the recognized template. They have reported that their approach achieved an average accuracy of 72.9% for ASL letters considering the context, and 62.2% without considering the context. They have used a 22 sensor CyberGlove. They have also tested their approach using lower number of sensors. Based on their report a smaller number of sensors dramatically decreases the accuracy of the gesture detector.

One of the difficulties in modeling of gesture signals is handling the variable length data which is not suitable for this category of classifiers. For instance, features such as speed, position or acceleration of a "circle drawing gesture signal" could be highly variable from one person to another. This makes applying some of the techniques like template matching and ANN even more difficult.

Applying a resizing technique for different input sizes such as DTW is one of the possible solutions. However, these techniques can be applied only for known boundaries of input signals. This limitation dictates that the user explicitly indicate "when" s/he has started and finished a gesture signal. Although satisfying this constraint for some applications like a digital pen or mouse is quite easy, for some others like vision-based gesture signal recognition it is difficult. This is because there is no alternative signal (such as mouse click or touching a surface) to specify the start and the end of the signal.

6.3 The gesture trajectory recognition technique

In the previous section the research background on gesture recognition system was presented. In this section, we describe our gesture recognition technique which is meant to be used as a component of a gesture-based user interface. The input to this component is the movement trajectory information of the gesture signal. The assumption is that the movement trajectory is captured by an intermediate device and presented to the system. Particularly for vision based hand tracking, the trajectory of the hand is provided by the vision-based hand tracker which can virtually be any method of tracking. The method for skin blobs tracking which we introduced in the previous chapter is one of the possible inputs. The gesture signal recognition can be represented as a classification problem in the feature space. This approach typically has two major components: Feature selection and classification.

6.3.1 Feature selection

Feature selection can be a challenging task. The number of the selected features should be as small as possible to be efficiently used by a classifier. There are also other issues such as sensitivity to input noise which may be caused by vibrations of the hand, small rotation and scale which may vary from person to person or with different input devices.

The movement trajectory of each gesture signal in two dimensional space can be represented as a set of (x_t, y_t) coordinates over time. Therefore, a gesture trajectory signal can be defined as a set of lines connecting the two sequential coordinates.

Although this representation makes the reconstruction of the input accurate, it is not invariant to position which makes it not suitable for a general gesture recognizer.

There are other alternatives for representing the shape of a gesture signal. The gesture signal can also be represented as a set of angles over time. The value of the angle sequence can be extracted directly from the set of coordinates using the gradient of the motion trajectory over time as represented in the following formula.

$$G_m = \left\{ \alpha_t \mid \alpha_t = \tan^{-1}\left(\frac{y_t - y_{t-1}}{x_t - x_{t-1}}\right) \right\}_{t=1..m} \quad \textbf{(6.1)}$$

- where G_m is a gesture signal recorded between time steps *0* to ***m***.
- α_t is the angle at time step *t*
- (x_t, y_t) represents the coordinate of the gesture movement trajectory at time-step *t*.

To reduce the effect of vibration and also having a smaller number of feature-values, the calculated angle is quantized to values of 10^o. Thus, each sample after quantization will have a value between 0 to 35 (Figure 6.1a). Hence, the input gestures can be described as a finite set of integer values.

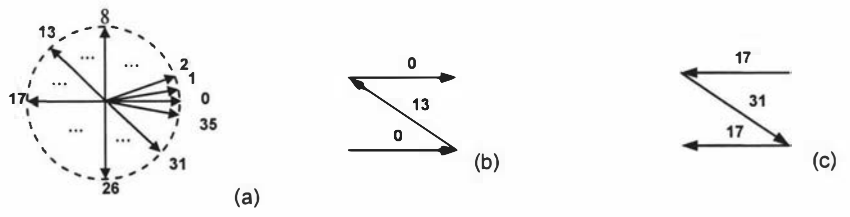


Figure 6.1. a) Quantized input vectors, b) Gesture vector (0, 13, 0), c) Gesture vector (17, 31, 17)

The advantage of this representation is its invariance to translation. Using this model, a gesture like Figure 6.1b will be different from Figure 6.1c, and the data, implicitly includes the time and the direction of the movement trajectory. Figure 6.2a shows a simple hand movement. The density of the arrows in different parts of the movement represents the speed of the hand in those parts. A higher density of arrows represents slower movement. It is observed that the hand has vibrations in some parts and the number of samples (arrows) in Figure 6.2a is considerably more than Figure 6.2b, which is the quantized version of the original movement trajectory. With this approach, a *gesture* is translated to a *gesture signal* (Figure 6.2c), which reduces the gesture recognition problem to a signal matching problem. Figure 6.3 shows that the proposed modeling of the gesture is invariant against position variation. The interesting feature of this model is that it transforms rotation in gesture space to a vertical shift in the angle space.

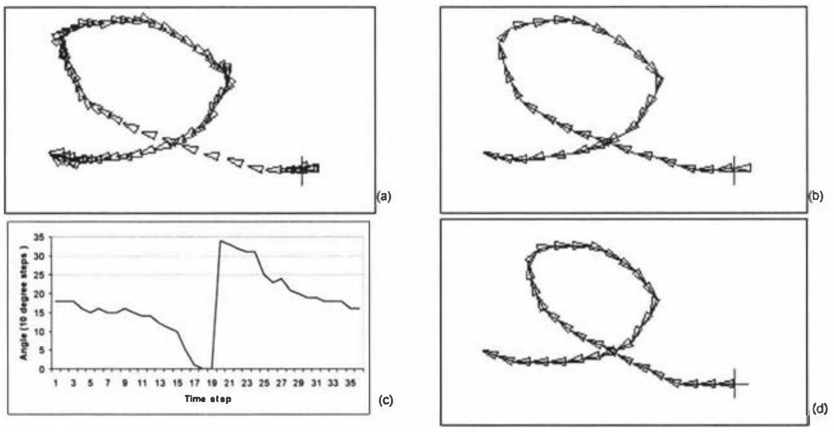
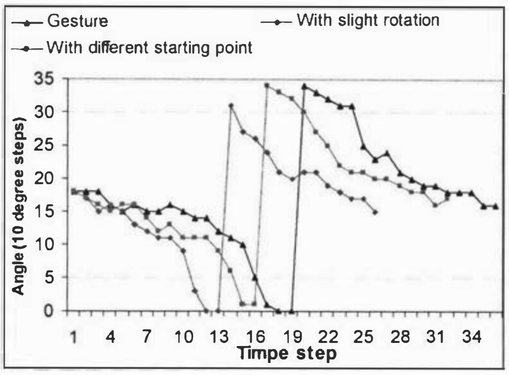


Figure 6.2. a) Original gesture trajectory, b) sampling from the gesture trajectory, c) gesture signal of the collected data over time, d) reconstructed gesture using [c].



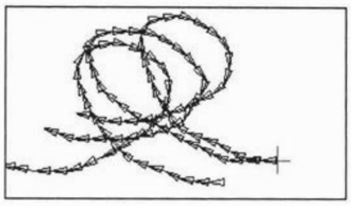
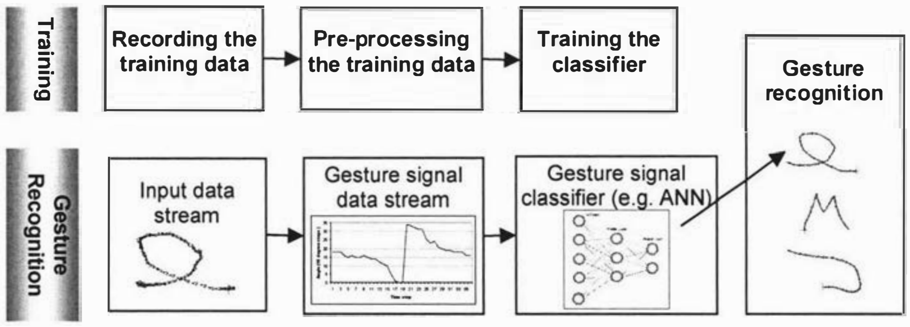


Figure 6.3. Left: Gesture signal, Right: gesture movement trajectories

6.3.2 Gesture Classification

After feature selection, introduced in the previous section, the next step is to implement a tool for recognition of a gesture signal. In this study, we applied a feed-forward neural network for gesture classification. Applying this classifier requires a primary training step which we describe in detail in the following sections. Figure 6.4 summarizes the steps involved in gesture recognition based on our technique. Obviously, the gesture set is application dependent. Hence, some factors like user's physical and ergonomical limitations in employing gesture signals should be considered.

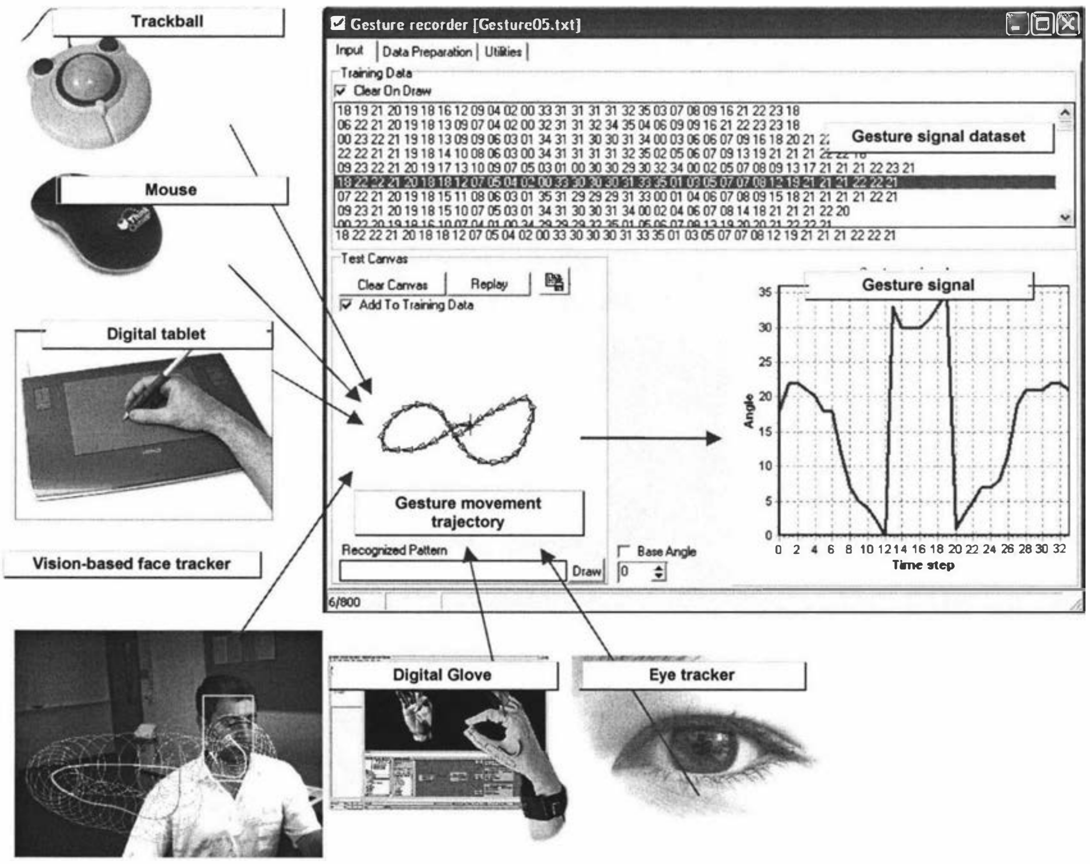


Training
Recording the training data
Pre-processing the training data
Training the classifier
Gesture recognition
Gesture Recognition
Input data stream
Gesture signal data stream
Gesture signal classifier (e.g. ANN)

Figure 6.4. Gesture recognition based on input gesture signal to a classifier.

6.3.3 Recording the training data

The data collection for training the classifier was done using the trajectory recoding software which we implemented for this purpose (Figure 6.5). The program records the movements of a mouse cursor on the "Test canvas" and translates it to a gesture signal. The start and end of each signal is indicated by the user by clicking and releasing the left mouse button. Hence, input devices such as optical mouse, trackball or digital tablet, which can control the mouse cursor are useable together with this application. Finally, the recorded gesture signals were then stored in a file for further processing.



```
18 19 21 20 19 18 16 12 09 04 02 00 33 31 31 31 31 32 35 03 07 08 09 16 21 22 23 18
06 22 21 20 19 18 13 09 07 04 02 00 32 31 31 32 34 35 04 06 09 09 16 21 22 23 23 18
00 23 22 21 19 18 13 09 09 06 03 01 34 31 31 30 30 31 34 00 03 06 06 07 09 16 18 20 21 2
22 22 21 21 19 18 14 10 08 06 03 00 34 31 31 31 31 32 35 02 05 06 07 09 13 19 21 21 21 2
09 23 22 21 20 19 17 13 10 09 07 05 03 01 00 30 30 29 30 32 34 00 02 05 07 08 09 13 17 21 21 21 22 23 21
18 22 22 21 20 18 18 12 07 05 04 02 00 33 30 30 30 31 33 35 01 03 05 07 07 08 12 19 21 21 21 22 22 21
07 22 21 20 19 18 15 11 08 06 03 01 35 31 29 29 29 31 33 00 01 04 06 07 08 09 15 18 21 21 21 21 22 21
09 23 21 20 19 18 15 10 07 05 03 01 34 31 30 30 31 34 00 02 04 06 07 08 14 18 21 21 21 22 20
18 22 22 21 20 18 18 12 07 05 04 02 00 33 30 30 30 31 33 35 01 03 05 07 07 08 12 19 21 21 21 22 22 21
```

Figure 6.5. The gesture signal recording software.

6.3.4 Normalizing the data

Although our modeling approach has the mentioned attractive features, because of the nature of the gesture which is the movement trajectory information, the number of samples may vary for each user or even at each time of the gesture input. The varying number of samples on the other hand is not suitable for classifiers. Most classifiers such as Neural-Networks, Support Vector Machines, and Eigen Vector Regression require a certain number of inputs which represents the number of dimensions in the feature space. Therefore, a pre-process is required to equate the size of the input data. Since the effects of the different normalization approaches using this technique are unknown, we designed a set of experiments to study the results of the different normalization techniques to find a realistic approach providing a reasonable accuracy. The details of the experiments and the results of their evaluation are drawn in the following sections.

6.4 The experiments

The training and evaluation platform of our gesture recognition technique was the Neural Network library of MatLab, Version 6.5, with sigmoid activation function for all neurons in all layers.

6.4.1 The first experiment - Using one ANN

In this step, we used a multi-layer feed forward neural-network with 28 inputs, two inner layers containing 20 and 9 neurons respectively, and three output neurons each representing the detection of one gesture signal. The outputs translated to yes/no, using a HardLimit transform (Figure 6.6).

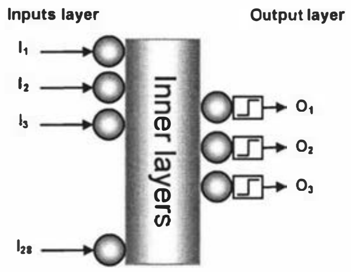
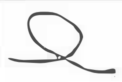
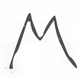
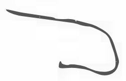


Figure 6.6. The structure of the ANN for gesture classification

The challenge of this model was the different size of the average vectors for each gesture. Instead of using the average size for the normal vector, we used a constant size of 28 (the maximum of the average size of the gesture classes) as the general vector size, and then applied the trimming/replication of the tail of the vector to set the training data size to 28. The results of the first experiment are presented in Table 6.1. Figure 6.7 presents the performance of the ANN in training within 4000 iterations.

Table 6.1. First experiment, detecting multiple gestures using a single ANN

Gesture	 (1)	 (2)	 (3)
Size of the training data	235	221	254
Average size	31	29	23
Size of the normal vector	28	28	28
Size of the test data	126	113	96
Accuracy of the classifier for each gesture	100%	98.23%	97.917%

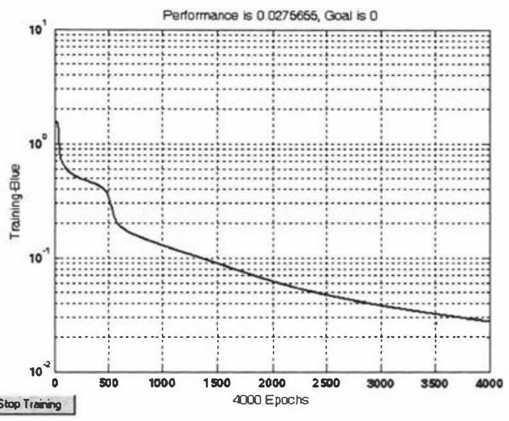


Figure 6.7. Performance of the ANN over time while training (Goal was 0).

- **Gesture detection from continuous input**

In this experiment, we also applied a continuous gesture signal to the trained ANN. The inputs of the ANN were initially set to 0 and each input angle was pushed to the input queue at the time of observation. The sequence of the three gestures was correctly detected. However, the boundaries of the gesture signal was not determinable because of the multiple detections of each gesture.

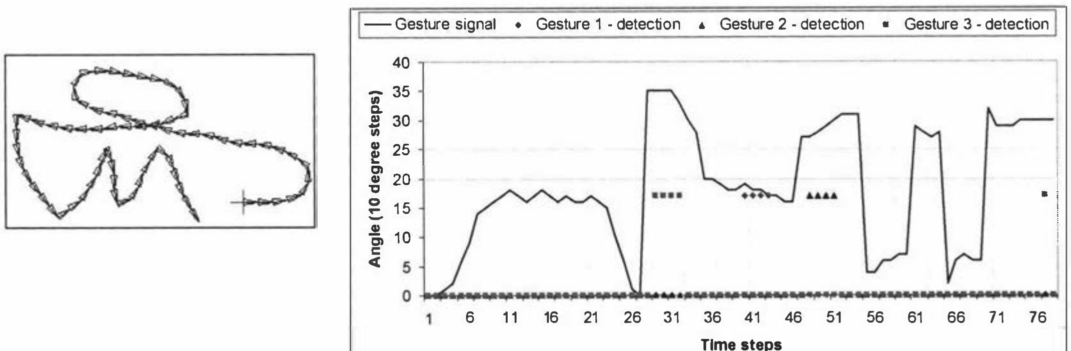


Figure 6.8. Continues gesture detection using one ANN

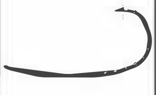
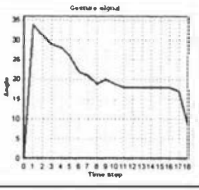
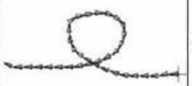
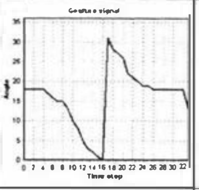
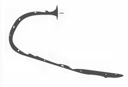
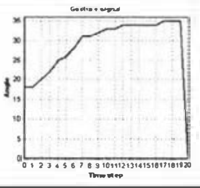
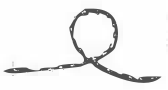
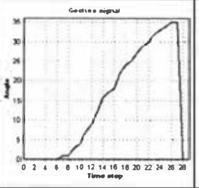
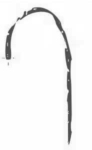
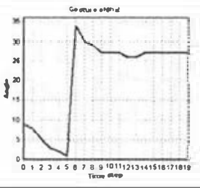
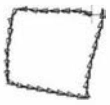
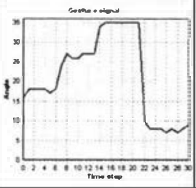
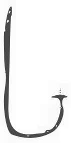
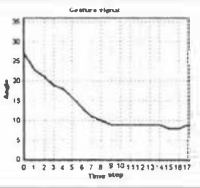
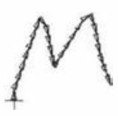
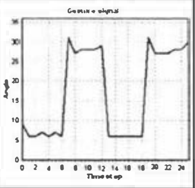
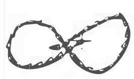
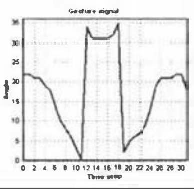
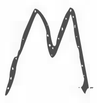
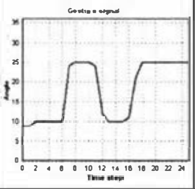
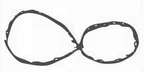
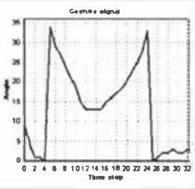
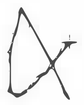
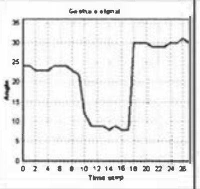
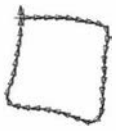
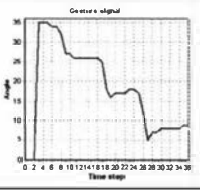
6.4.2 The second experiment

The promising results from applying this technique in the previous experiment encouraged us to apply this approach on a more comprehensive set of gesture signals with more freedom in terms of the size of gesture signals. For this purpose, using the gesture recording software and a digital tablet, a total of 7392 gesture signals of 13 gesture classes were recorded by three individuals (two males and one female). The detailed information of the recorded samples is presented in Table 6.2.

In this experiment, we used one ANN for detecting each gesture with a constant input size of 45 (the maximum size) as the input vector to an ANN. The method of size adjustment in this prototype was based on padding the input vectors with random values. Hence, the test and the training data both carry two types of noise. The first one is what was recorded together with the user's input (e.g. vibrating hand). The second one is the random values which were added to the gesture signal. It is expected that by using this strategy, the ANN adjusted its weight such that the lower inputs to be less important in classification and at the same time to be able to classify the real input data robustly.

To train the ANN, 80% of each gesture signal was used as positive samples. Another dataset equal to the size of the positive training data was used as the negative samples containing 50% other gesture signals and 50% random values. For evaluating the ANNs, we used the remaining 20% of the gesture signals as positive test data and a dataset with equal size containing 50% of other gesture signals and 50% of random signals as negative test data.

Table 6.2. The data recorded for the second experiment, using the gesture recorder and a digital tablet.

Gesture no.	Gesture trajectory	Gesture signal	Number of samples	Size Min Max Avg	Gesture no.	Gesture trajectory	Gesture signal	Number of samples	Size Min Max Avg
1			500	15 22 17.91	8			504	27 38 30.34
2			515	14 24 19.93	9			518	24 38 29.31
3			517	16 26 20.75	10			513	27 43 34.11
4			524	11 23 16.49	11			507	17 36 26.96
5			800	20 42 30.65	12			536	19 38 27.9
6			829	21 42 31.56	13			576	16 33 25.03
7			553	29 45 36.08	14	Random signal	-	-	-

We used the collected training datasets for training 13 feed forward ANNs each representing one of the gesture signals. Each ANN had two inner layers of 45 and 30 neurons respectively and one output. The result of this experiment is presented in Table 6.3. The output of the ANN was interpreted by a HardLimit transform. Considering that the reliability of applying the HardLimit transform may be in question, we analyzed the output values of the ANNs. The histogram in Figure 6.9 shows that most of the output values are located at -1 and +1 thresholds. Therefore, shifting the threshold which is zero in the HardLimit transform causes the interpretation for a minority of the outputs to be changed. We evaluated the output values of the ANNs based on thresholds from -0.9 to +0.9 with step 0.1. The effects of changing the threshold are presented in Figure 6.10. It is seen that by changing the threshold the correct and false detection ratio change together. The distance between the correct detection ratio and the false detection ratio at position zero is maximal. Therefore, the hard-limit transform is the best interpretation of the output.

Table 6.3. The results concluded from the experiment 2 (one ANN for each gesture signal)

Gesture No.	Correctly detected	Total correct	Correct positive detection%	Falsely detected	Total false	False positive detection%	False positive to correct positive detection
1.	80	80	100	23	402	5.72	0.0572
2.	110	112	98.21	0	397	0	0
3.	101	103	98.06	35	397	8.81	0.0899
4.	96	103	93.20	29	401	7.23	0.0775
5.	175	176	99.43	15	602	2.49	0.0251
6.	176	182	96.70	23	619	3.72	0.0384
7.	93	93	100	15	440	3.41	0.0341
8.	82	84	97.62	27	407	6.63	0.0679
9.	96	96	100	0	409	0	0
10.	89	90	98.89	13	417	3.12	0.0315
11.	103	105	98.09	6	387	1.55	0.0158
12.	105	105	100	18	411	4.38	0.0438
13.	116	118	98.30	12	436	2.75	0.0280
Summary			98.35			3.83	0.0392

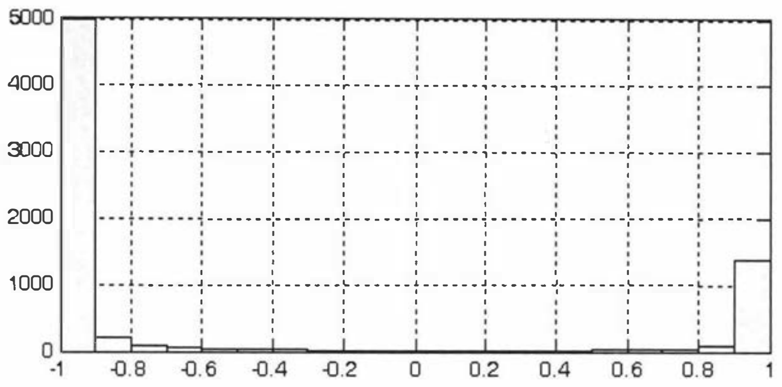


Figure 6.9. The histogram of the output values of the ANNs for the test data.

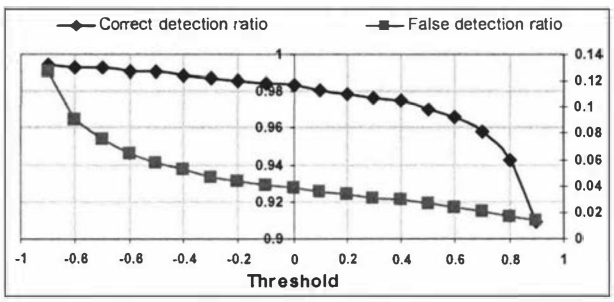


Figure 6.10. The effect of the change of the threshold on average correct detection ratio (middle vertical axis) and false detection ratio (right vertical axis).

To analyze the accuracy of this method of classification more appropriately, the measured values are shown in a confusion matrix in Table 6.4. For this purpose, the entire test dataset was labeled from 1 to 14 representing the 13 gesture alphabet and randomly generated gesture signals. Each ANN was evaluated using this data and false positive detection of each gesture signal measured amongst other ANN gesture classifiers. Since the test dataset is not the same as used before (measurements in Table 6.3), the values here in Table 6.4 are different. The values in the diagonal cells represent the accuracy of each ANN in classifying its peer gesture and other cells represent the false positive detections of each ANN in dealing with non-peer gesture signals.

Table 6.4. Analyzing the accuracy/inaccuracy of multiple ANNs for gesture classification using confusion matrix (all values are in percent).

NN	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6	Gesture 7	Gesture 8	Gesture 9	Gesture 10	Gesture 11	Gesture 12	Gesture 13	Random Gesture Signals - 14
N1	99.64	0.06	-	0.02	0.01	-	0.29	-	0.01	-	-	-	-	3.16
N2	-	99.06	0.01	-	-	-	0.04	-	-	0.03	-	-	-	0.33
N3	-	0.87	97.5	-	0.08	1.74	2.66	-	0.9	0.33	0.49	0.32	0.01	1.59
N4	0.38	0.01	-	94.13	1.61	0.01	0.01	1.05	0.6	-	-	0.01	0.76	3.98
N5	0.01	0.01	-	0.28	98.98	0.29	-	0.29	0.12	0.01	0.14	-	0.35	0.87
N6	-	-	0.25	0.01	0.03	97.39	0.15	0.01	0.32	0.03	0.88	0.13	0.06	1.71
N7	0.04	0.51	0.99	-	0.01	0.1	99.65	-	0.01	0.29	-	-	0.01	0.57
N8	-	0.01	-	2.05	0.97	0.03	-	98.57	0.21	-	0.26	0.01	1.1	2.16
N9	-	-	-	0.04	-	-	0.04	-	100	-	0.01	-	0.01	0.1
N10	0.04	1.0	0.07	-	0.21	0.09	0.32	-	-	98.61	0.01	0.01	0.01	1.35
N11	-	-	-	-	0.29	1.13	-	-	0.06	-	98.08	0.46	-	0.31
N12	-	-	0.25	-	0.07	1.12	0.06	-	0.13	0.01	3.0	100	-	2.65
N13	0.15	1.16	-	0.17	0.83	-	-	0.25	0.32	0.07	0.03	-	98.27	0.82

6.4.3 Comparing the ANN classifier with a SVM classifiers

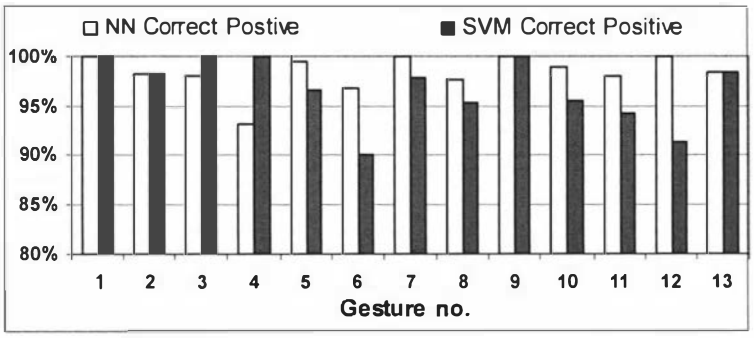
It is an interesting question that whether or not other methods of classification would be superior to the ANN method for gesture signal classification. To answer this question we trained a support vector machine (SVM) using the same training data. The kernel we used in the SVM was a radial basis function (RFB) kernel based on the equation below, with $\sigma = 0.5$.

$$K(x_i, x_i) = \exp(\frac{\|x_i - x_j\|^2}{2\sigma^2}) \qquad \textbf{(6.2)}$$

The results of the classification using the SVM on the test data is presented in Table 6.5. Figure 6.11 represents the result of the classification using SVMs on the test data in comparison to ANNs.

Table 6.5. The evaluation of a support vector machine in classifying the 13 gesture signals.

Gesture No.	Correctly detected	Total correct	Correct positive detection%	Falsely detected	Total false	False positive detection%	False positive to correct positive detection
1.	80	80	100.00	31	402	7.71	0.0771
2.	110	112	98.21	6	397	1.51	0.0154
3.	103	103	100.00	28	397	7.05	0.0705
4.	103	103	100.00	51	401	12.72	0.1272
5.	170	176	96.59	27	602	4.49	0.0464
6.	164	182	90.11	64	619	10.34	0.1147
7.	91	93	97.85	32	440	7.27	0.0743
8.	80	84	95.24	69	407	16.95	0.1780
9.	96	96	100.00	0	409	0.00	0.0000
10.	86	90	95.56	36	417	8.63	0.0903
11.	99	105	94.29	44	387	11.37	0.1206
12.	96	105	91.43	62	411	15.09	0.1650
13.	116	118	98.31	16	436	3.67	0.0373
Summary			96.34			8.14	0.0845



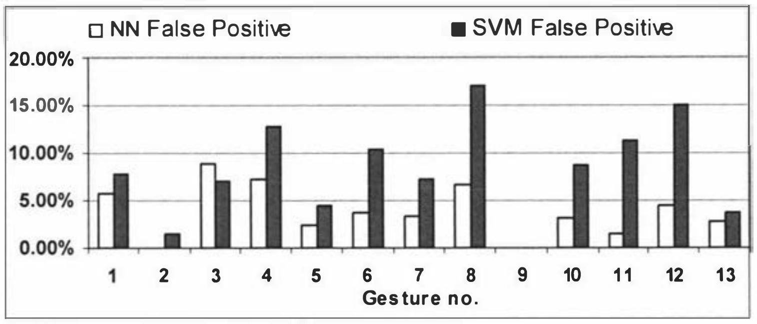


Figure 6.11. The evaluation of a support vector machine in classifying the 13 gesture signals in comparison to the ANN classifiers.

The average accuracy of the SVM classifier was 96.34% and 8.14% for correct positive and false positive detections, respectively. The high accuracy in correct-detection makes the SVM classifier comparable to the feed forward ANN, however the 8.14% average false detection ratio makes the output of the SVM less reliable in comparison to ANN which was 3.83%.

We should note that in this experiment the performance of the SVM in terms of the number of floating point multiplications and summations was better than ANN. However, this comparison can not be generalized, because the performance is dependent on the number of neurons in the ANN and the size of the support vector in the SVM.

6.4.4 Gesture classification using one ANN with 14 outputs

We also implemented the gesture classifier based on what was introduced in Section 6.4.2 using a single ANN. This structure is preferable because of the following advantages. Firstly, the misclassification ratio is automatically decreased at the time of training and therefore the output would be more reliable. Secondly, in case of using the same number of hidden layers, the computation cost of a single recognizer would be less than 13 individual recognizers.

Hence, a single feed forward ANN with 45 inputs and 14 outputs was designed for classification (Figure 6.12). For training the ANN, each element of the training dataset was labeled by the relevant gesture numbers from 1 to 13 and random gesture signals also labeled to 14. The results indicating the accuracy of the ANN for each gesture signal are presented in Table 6.6.

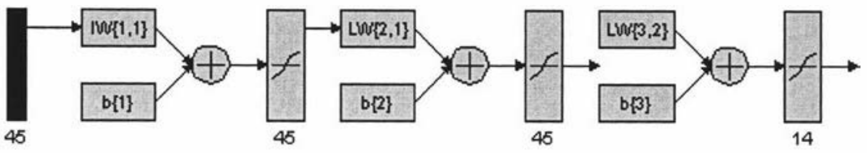


Figure 6.12. The structure of ANN for detecting multiple gesture signals.

Table 6.6. The evaluation of a single ANN in classifying 14 gesture classes using 2 hidden layers.

Gesture No.	Correctly detected	Total correct	Correct positive detection%	Falsely detected	Total false	False positive detection%	False positive to correct positive detection
1.	164	165	99.39	6	2424	0.247	0.0024
2.	166	169	98.22	1	2420	0.041	0.0004
3.	151	153	98.69	3	2436	0.123	0.0012
4.	137	139	98.56	7	2450	0.288	0.0029
5.	160	166	96.38	3	2423	0.123	0.0012
6.	160	164	97.56	2	2425	0.082	0.0008
7.	146	159	91.82	1	2430	0.041	0.0004
8.	159	166	95.78	2	2423	0.082	0.0008
9.	156	156	100	0	2433	0	0
10.	130	140	92.85	3	2449	0.123	0.0013
11.	156	163	95.71	0	2426	0	0
12.	175	179	97.76	4	2410	0.165	0.0016
13.	176	185	95.14	3	2404	0.124	0.0013
14.	419	485	86.39	14	2104	0.577	0.0066
Summary			96.02			0.144	0.0015

6.4.4.1 The optimum design for gesture classifier

Choosing the right number of hidden layers is a critical factor in design and training an ANN classifier. Although a larger number of hidden layers may increase the accuracy of the ANN in training, it is not preferred due to some side effects. Firstly, the number of hidden layers may be significantly effective in the required computation time to process the input which is important in the case of using a software-based ANN. Secondly, it may also cause the ANN to simply memorize the training data rather than identifying patterns in it. Therefore, choosing the optimum number of hidden layers is one of the design issues.

To identify the optimum number of hidden layers, we analyzed the ANN using 0 to 7 hidden layers, each layer containing 45 neurons. Table 6.7 represents the evaluated and measured values from each ANN. From this result, it is concluded that 1 or more hidden layers are required to gain acceptable output accuracy. However, with more than 2 hidden layers the output accuracy would not significantly improve (Figure 6.13).

Table 6.7. Comparison of the accuracy of a single ANN containing 0 to 7 hidden layers in recognizing the 14 gesture signal classes.

Number of hidden layers	Correct positive detection (Avg%)	False positive detection (Avg%)	Epochs passed	Accuracy reached (Goal was 0.0001)
0	26.83	0.05	66	0.1013
1	88.11	0.18	423	0.0536
2	96.02	0.14	593	0.0285
3	96.67	0.16	446	0.0199
4	96.69	0.18	339	0.0157
5	96.42	0.19	369	0.0142
6	96.47	0.21	411	0.0116
7	96.51	0.21	457	0.0098

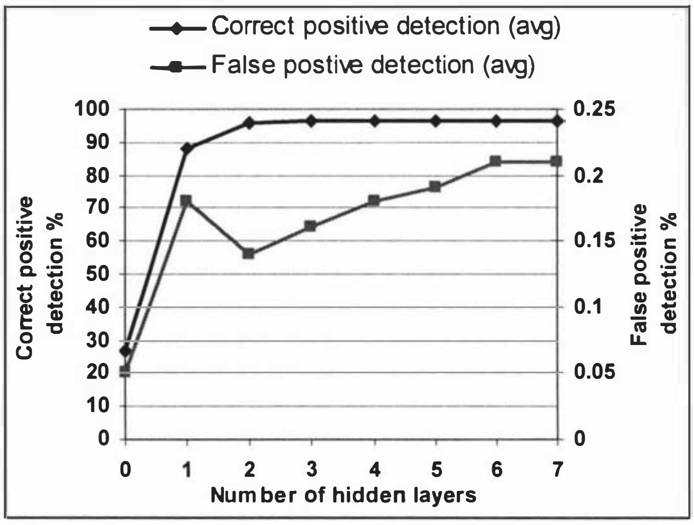


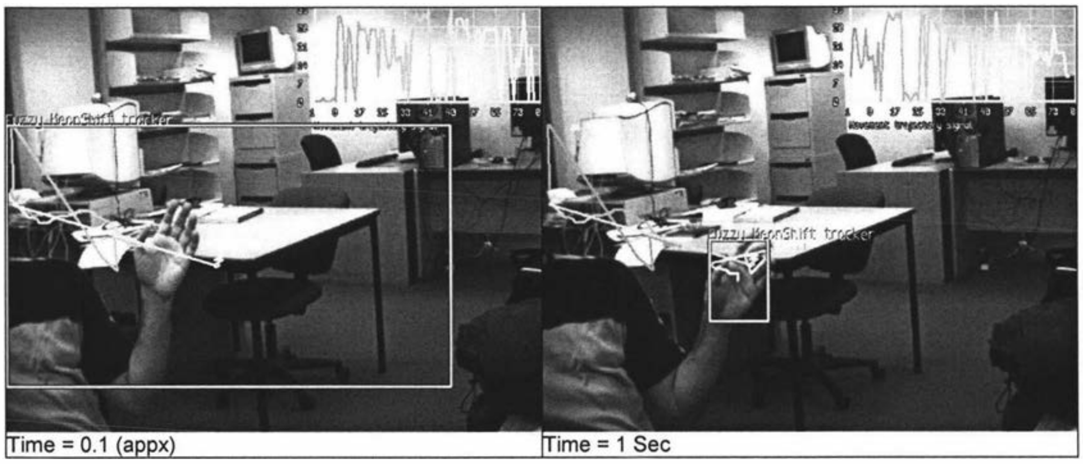
Figure 6.13. Comparison of correct positive and false positive recognition accuracy of a single ANN in gesture signal classification employing different numbers of hidden layers.

6.5 Implementation of the gesture tracking system

6.5.1 System inputs

The inputs to the system were provided by an implementation of the Fuzzy-based Mean-shift tracker as described in Chapter 5. The tracker, tracked the biggest skin blob present in the video sequence and translated its movement trajectory to a sequence of angles as required for the gesture recognition technique which was introduced in this chapter.

Figure 6.14 represents the movement trajectory of gesture number 6 and its gesture signal. The gesture signal is presented on right-top corner of each frame. Hence, the gesture signal can visually be matched to the known gesture signals. The movement trajectory was extracted from the centre of gravity of the hand blob as displayed in each frame. The frame grabber was a Dragonfly digital camera (15 fps).



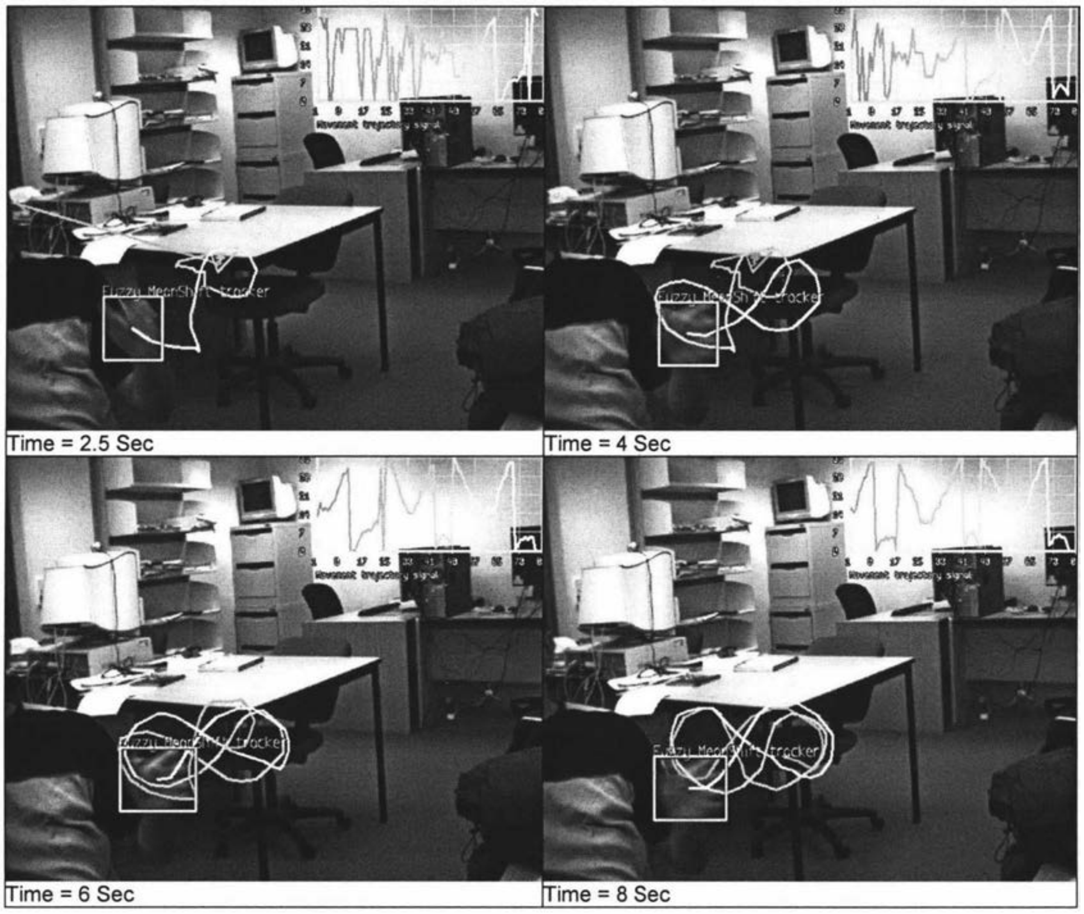


Figure 6.14. Recording the hand movement trajectory using the Fuzzy-based Mean-shift tracker. The gesture movement trajectory signal is presented on the right-top corner of each frame.

6.5.2 Gesture movement trajectory recognition

The implementation was carried out using one ANN with 45 inputs, 14 outputs and two hidden layers each with 45 neurons. For this purpose, the ANN discussed in Section 6.4.4 was trained in Matlab. Then, the parameters of the ANN including weights and bias values were exported to a data file and used by a feed-forward ANN implementation in C++. The processing power which the ANN required to process its inputs was also measured. It was on average equal to 68 micro seconds on a PC with a Pentium 4-2.8GHz processor. This also indicates that the ANN itself requires just 0.2% of the CPU processing time which means it can be used in real-time.

The inputs of the ANN were provided through an input buffer with size 45 containing the gesture's movement trajectory signal. In each time-step the detected trajectory angle was pushed into the input buffer. Then, the input buffer was delivered to the ANN for processing and the outputs of the ANN were recorded for demonstration. Figure 6.15 presents the output of the described implementation. The diagram on the top-right corner of the image represents the gesture signal. The other diagram on the bottom-right corner of the image represents the history of detected gesture signals. The horizontal axis of this diagram is frame number and the vertical axis is gesture number. Each marker against each gesture number indicates one positive detection of that gesture per frame. As it is observable in this figure, gestures number 5 and 14 were detected. It is worth mentioning that detection of gesture number 14 has no particular meaning here, however, its presence indicates the existence of a noisy input signal. Figure 6.16 presents the recognition of gestures number 9 and 13 using the vision-based gesture recognition system introduced in this section[21].

[21] The original videos are available at: http://www.massey.ac.nz/~fdadgost/xview.php?page=videos

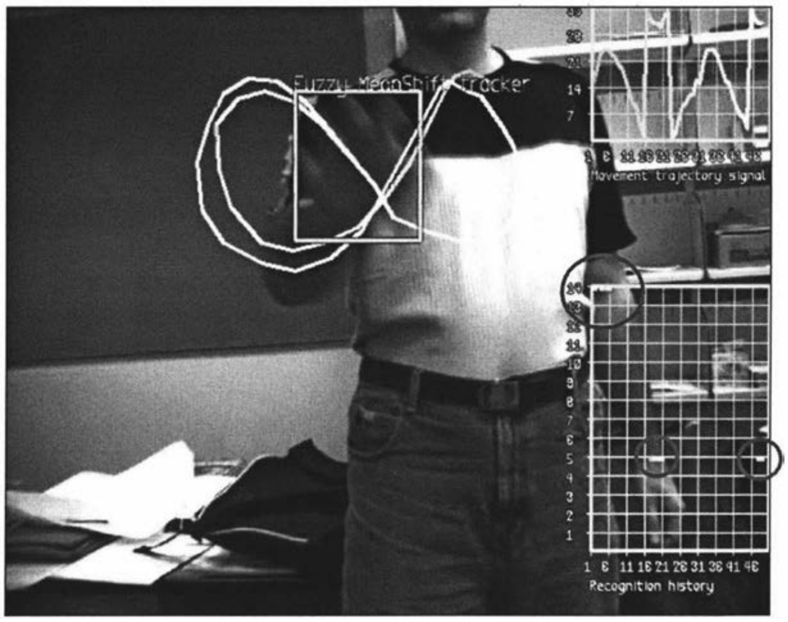


Figure 6.15. Implementation of the vision-based gesture recognition using one ANN and the Fuzzy-based Mean-shift tracker. The diagram on the right-bottom corner of the image represents the detected gestures over time[22].

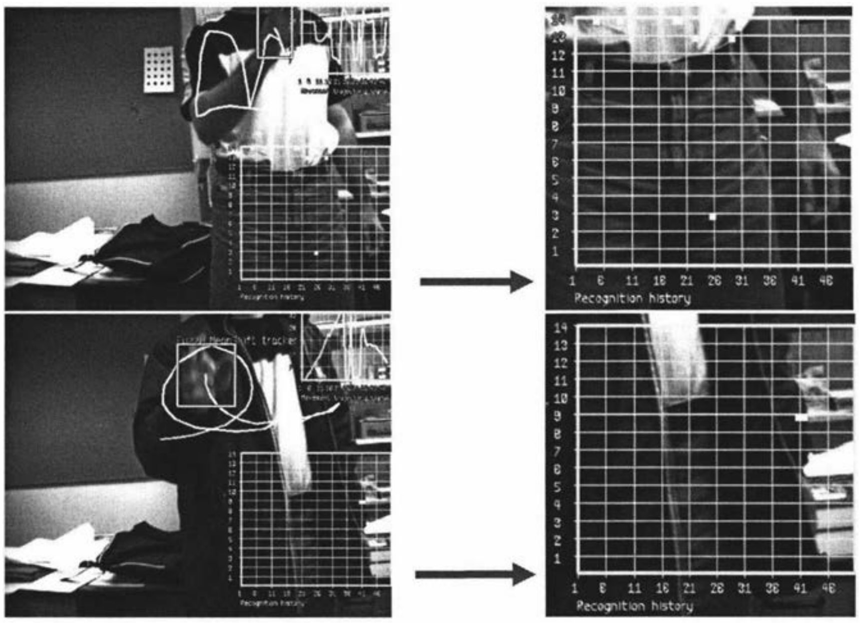


Figure 6.16. Recognition of gesture signals number 13 and 9 using the vision-based gesture recognition system[23].

[22] Please refer to video#9 in the enclosed CD to view the full video

6.6 Chapter summary

In this chapter, we presented a novel approach for gesture recognition. This approach has two main steps: i) gesture modeling, and ii) gesture recognition. For gesture recognition, we used a multi-layer feed-forward neural-network. The results of our first experiment show 99.72% average accuracy in gesture recognition. In the second experiment we used another ANN for gesture classification which shows 98.71% average accuracy for gesture recognition. Based on the high accuracy of the gesture classification, the number of ANN layers seems to be enough for detecting a limited number of gestures. However, more accurate judgment requires the more number of gestures in the gesture-space to further validate this assertion. Our observation also shows that this technique is a potential approach for continuous gesture classification. Table 6.8 presents the accuracy of the proposed gesture recognition technique in comparison to other techniques introduced in the literature.

The gesture recognition technique introduced in this chapter can be used with a variety of front-end input systems such as vision-based input, hand and eye tracking, digital tablet, mouse, and digital glove. We applied the Fuzzy-based Mean-shift tracker introduced in Chapter 5 as the front-end of this system. We also showed that gesture signals recorded using our vision-based hand tracker can be used as the input to the gesture recognition system.

[23] Please refer to videos #10 & #11 in the enclosed CD to view the full video

Table 6.8. Comparison the accuracy of the proposed method with other gesture recognition systems in the literature.

Application / Description	Underlying applied technique	Accuracy		Gesture alphabet
		Correct positive	False positive	
Movement trajectory detection [78]	HMM	97.5% - 99.2%	NA	Drawing three shapes (circle, triangle, square)
Mobile sign language recognition using [83]	GT^2K – HMM	52.38% (Vision-based) 65.87% (Accelerometer)	NA	40 words vocabulary in a controlled environment
Workshop activity recognition [83]	GT^2K – HMM	93.33%	NA	10 gestures
Movement analysis based on intensity image sequence and a hand mask image sequence [84]	Spatio-temporal appearance in 2D space	89%	NA	12 gestures
Dynamic hand writing recognition [1]	HMM and STFT for feature selection	99.7%	NA	9 gestures
Gesture controller [84]	Gesture signal matching and DTW	89.6%	NA	12 gestures
Gesture movement trajectory recognition (section 6.4.1)	Single ANN and angle space	98.71%	NA	3 gestures
Gesture movement trajectory recognition (section 6.4.2)	Multiple ANN and angle space	98.34%	3.83%	13 gestures
Gesture movement trajectory recognition (section 6.4.3)	SVM and angle space	96.34%	8.14%	13 gestures
Gesture movement trajectory recognition (section 6.4.4.1)	Single ANN and angle space	96.02%	0.14%	13 gestures

Chapter 7. Summary, conclusion and future research

In this dissertation we presented the research pathway in design and implementation of a real-time vision-based gesture recognition system. This system was build based on three components, representing three layers of abstraction: i) detection of skin and localization of hand and face, ii) tracking multiple skin blobs in video sequences and finally iii) recognition of gesture movement trajectories.

The adaptive skin detection, the first component, was described in Chapter 4. This component was implemented based on our novel *adaptive skin detection algorithm* for video sequences. This algorithm has two main sub-components: i) the static skin detection which is a skin detection method based on the hue factor of the skin color, and ii) the adaptive skin detector which retrains itself based on new data gathered from movement of the user. We evaluated this algorithm using two motion detection methods.

The results based on both of these motion detection methods show that the algorithm can improve the quality of the skin detection within the video sequences. Specifically, the adaptive skin detector based on frame subtraction motion detector provides better results in terms of correct detection ratio. The effect of some of the morphological operators on this algorithm was also studied. The results shows Erode and Erode-

Erode operators significantly reduce the number of falsely detected pixels. This means that a majority of the falsely detected pixels are sparse individual pixels which can easily be eliminated.

The role of the next component is tracking hand and face in video sequences. For this purpose, as presented in Chapter 5 a new approach for boundary detection in blob tracking based on the Mean-shift algorithm was proposed. Our approach is based on continuous sampling of the boundaries of the kernel and changing the size of the kernel using our novel Fuzzy-based algorithm. We compared our approach to the kernel density-based approach which is known as the CAM-Shift algorithm, in a set of different noise levels and conditions.

We analyzed the proposed algorithm based on two main factors: correct boundary detection and stability against white noise. The results show that the proposed approach is superior in stability against white noise and provides correct boundary detection for arbitrary hand postures which is not achievable by the CAM-Shift algorithm. Based on the results, we conclude that our proposed method of boundary detection can be effective for fast and robust tracking of the face and hand.

We also presented three variances of the proposed algorithm with the applications of hand and face tracking in HCI environments. The first variance addresses tracking of multiple skin blobs in the image sequence. The blobs in the context of HCI applications are the hand and hand face. Considering the fact that the skin detection algorithms may provide disperse set of pixels with variable density over time, the second algorithm provides a high level of tracking which we called it "Macro tracker". And the third algorithm addresses applying the depth information in blob tracking. The later algorithm is particularly more robust in handling occlusion which may occur in HCI applications.

Moreover, the robustness of our method against noise makes it a good candidate for use with cheap cameras and real-world vision-based HCI applications. This method is to be applied in conjunction with a fast pixel-based skin color segmentation algorithm as the level of noise and the quality of the skin detection are not deterministic.

Finally in Chapter 6o main parts: i) gesture modeling, and ii) gesture recognition. The gesture modeling technique is based on sampling the gradient of the gesture movement trajectory and presenting the gesture trajectory as a sequence of numbers. This technique has some important features for gesture recognition including robustness against slight rotation, small number of required samples, invariance to the start position and device independence. For gesture recognition, we used a multi-layer feed-forward neural-network. The results of our experiments show that this approach provides 98.71% accuracy for gesture recognition, and provides higher accuracy rate than other methods introduced in the literature.

These components form the required framework for vision-based real-time, gesture recognition and hand and face tracking. The components, individually or as a framework can be applied in scientific and commercial extensions of either vision-based or hybrid gesture recognition systems.

7.1 Extending the work

This work can be extended in several directions. Firstly, the adaptive skin detection algorithm can be used as a framework for localizing the skin in video sequences. Hence, limiting the search space in the image can dramatically improve the performance of computationally expensive, hand and face tracking algorithms. Moreover, employing other techniques such model-based and shaped-based matching would be easier and more accurate after limiting the search space.

Secondly, the Fuzzy-based tracking algorithm can be enhanced using extra information from the image. As we demonstrated in Chapter 5, depth information can successfully be applied for this purpose. This makes the algorithm more accurate in distinguishing between objects and determining their boundaries. Particularly, this enhancement may be useful for vision-based human computer interaction applications.

Finally, the gesture recognition technique can be extended in several directions. Ability of detection of hand postures and interpreting more complex gestures will move this technique to the next level. This ability makes the final gesture recognition system suitable for other purposes such as sign language interpretation or implementing more comprehensive man-machine interfaces. Adding more channels of information is another challenge and may be necessary for some specific applications. For instance, the movement trajectory in 3D space or having the movement velocity may be required for controlling a virtual orchestra or manipulating a virtual environment by hands.

7.2 The future of gesture recognition systems

Vision-based gesture recognition technology is still in the early stages of development, but it is growing steadily. According to Gartner, in the next few years, gesture recognition will probably be used primarily in niche applications [2]. The activities of the big players in the ICT industry are witnessing to this claim. Recently, Microsoft[24], HP[25] and Sony have started investing this technology. Some of the cell phone giants, such as Nokia and Samsung have started to add simple gesture recognizers to their next generation cell phones. In addition, some of the software producers such as Electronic Arts upon availability of the game consoles equipped with camera have decided to add vision-based gesture recognition technology to the next generation of their games[26]. It is said that gesture recognition is one of the five technologies which will change the way you do business[27].

[24] http://www.microsoft.com/windowsxp/using/tabletpc/learnmore/gestures.mspx
[25] http://insight.zdnet.co.uk/software/0,39020463,39268336,00.htm
[26] www.gamespot.com/news/6148455.html
[27] http://www.inc.com/magazine/20060701/column-freedman.html

References

[1] J. Yang and Y. Xu, "Gesture Interface: Modeling and Learning," presented at IEEE International Conference on Robotics and Automation, 1994.

[2] D. Geer, "Will gesture-recognition technology point the way?," in *Computer*, vol. 37, 2004, pp. 20-23.

[3] S. Goldin-Meadow and M. W. Alibali, "Looking at the hands through time: A micro-genetic perspective on learning and instruction," in *In Micro development: Transaction processes in development and learning*. New York: Cambridge University Press, 2002, pp. 88-105.

[4] S. Goldin-Meadow and M. A. Singer, "From Children's Hands to Adult's Ears: Gesture's Role in the Learning Process," *Development Psychology*, vol. 39, pp. 509-520, 2003.

[5] S. Goldin-Meadow, *Hearing Gesture: How Our Hands Help Us Think*. Cambridge, MA: Belknap, 2003.

[6] S. Lu, G. Tsechpenakis, D. N. Metaxas, M. L. Jensen, and J. Kruse, "Blob Analysis of the Head and Hands: A Method for Deception Detection," in *Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05)*, vol. 1: IEEE Computer Society, 2005, pp. 20.3.

[7] ANSA, "Brain stimulation pinpoints gesture recognition," vol. 2006, 2006.

[8] Y. Wu and T. S. Huang, "Hand Modeling, Analysis, and Recognition For Vision-Based Human Computer Interaction," *IEEE Signal Processing Magazine*, vol. Special issue on Immersive Interactive Technology, pp. 51-60, 2001.

[9] J. Daugman, "Face and gesture recognition: Overview," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 675-676, 1997.

[10] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 677-695, 1997.

[11] Y. Wu and T. S. Huang, "Vision-based gesture recognition: A review," in *Gesture-Based Communication in Human-Computer Interaction*, vol. 1739, *Lecture Notes in Artificial Intelligence*, 1999, pp. 103-115.

[12] K. W. Wong, K. M. Lam, and W. C. Siu, "An Efficient Algorithm for Human Face Detection and Facial Feature Extraction under Different Conditions," *Pattern Recognition, U.S.A*, vol. 34, pp. 1993-2004, 2001.

[13] J. Wood, "Invariant pattern recognition: a review," vol. 29, pp. 1-17, 1996.

[14] M. Turk and A. Pentland, "Face recognition using eigenfaces," presented at Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Maui, Hawaii, 1991.

[15] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," in *Data Mining and Knowledge Discovery*, vol. 2. Boston: Kluwer Academic Publisher, 1998, pp. 121-167.

[16] J. Maydt and R. Lienhart, "Face Detection with Support Vector Machines and a Very Large Set of Linear Features," presented at ACM Multimedia, Juan Les Pins, France, 2002.

[17] G. McGlaun, F. Althoff, M. Lang, and G. Rigoll, "Robust video-based recognition of dynamic head gesture in various domains - Comparing a rule-based and a stochastic approach," presented at The 5th International Workshop on Gesture and Sign Language based Human-Computer Interaction, 2003.

[18] S. Nagaya, S. Seki, and R. Oka, "A fuzzy rule-based approach to spatio-temporal hand gesture recognition," presented at Automatic Face and Gesture Recognition, 1996., Proceedings ofthe Second International Conference on, 1996.

[19] M. C. Su, "A fuzzy rule-based approach to spatio-temporal hand gesture recognition," *IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews*, vol. 30, pp. 276-281, 2000.

[20] M. C. Su, Y. X. Zhao, H. Huang, and H. F. Chen, "A fuzzy rule-based approach to recognizing 3-D arm movements," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 9, pp. 191-201, 2001.

[21] M. Gargesha and P. Kuchi, "Facial expression recognition using artificial neural networks," presented at IEEE Artificial Neural Computation Systems, Department of Electrical Engineering Arizona State University, 2002.

[22] K. Murakami and H. Taguchi, "Gesture recognition using recurrent neural networks " in *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology* New Orleans, Louisiana, United States ACM Press, 1991 pp. 237-242

[23] H. Rowley and S. Baluja, "Rotation Invariant Neural Network-Based Face Detection," presented at IEEE Conference on Computer Vision and Pattern Recognition, 1998.

[24] N. Yoshiike and Y. Takefuji, "Object segmentation using maximum neural networks for the gesture recognition system," *Neurocomputing*, vol. 51, pp. 213-224, 2003.

[25] S. Z. Li and Z. Q. Zhang, "FloatBoost and Statistical Face Detection," *IEEE Transactions on Pattern Analysis and Machine Vision*, vol. 6, 2004.

[26] J. Martin, D. Hall, and J. L. Crowley, "Statistical gesture recognition through modelling of parameter trajectories," in *Gesture-Based Communication in Human-Computer Interaction*, vol. 1739, *Lecture Notes in Artificial Intelligence*, 1999, pp. 129-140.

[27] H. Schneiderman and T. Kanade, "A Statistical Method for 3D Object Detection Applied to Faces and Cars," presented at Conference on Computer Vision and Pattern Recognition, 2000.

[28] V. Athitsos and S. Sclaroff, "An Appearance-Based Framework for 3D Hand Shape Classification and Camera Viewpoint Estimation," presented at IEEE Conference on Automatic Face and Gesture Recognition, 2002.

[29] M. J. Black and A. D. Jepson, "EigenTracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol. 26, pp. 63-84, 1998.

[30] R. Cutler and M. Turk, "View-based interpretation of real-time optical flow for gesture recognition," presented at Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition, Nara, Japan, 1998.

[31] T. J. Darrell, I. A. Essa, and A. P. Pentland, "Task-specific gesture analysis in real-time using interpolated views," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 1236-1242, 1996.

[32] A. I. Tew and C. J. Gray, "A Real-Time Gesture Recognizer Based on Dynamic-Programming," *Journal of Biomedical Engineering*, vol. 15, pp. 181-187, 1993.

[33] Z. Yao and H. Li, "Tracking a Detected Face with Dynamic Programming," presented at Conference on Computer Vision and Pattern Recognition Workshop, 2004.

[34] Y. Q. Xu, B. C. Li, and B. Wang, "Face recognition by fast independent component analysis and genetic algorithm," presented at CIT '04, The Fourth International Conference on Computer and Information Technology, 2004.

[35] L. Gupta and S. Ma, "Gesture-based Interaction and Communication: Automated Classification of Hand Gesture Contours," *IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews*, vol. 31, pp. 114-120, 2001.

[36] N. Otsu, "A Threshold Selection Method from Gray-Level Histogram," *IEEE Transactions. Systems, Man and Cybernetics*, vol. 9, pp. 62-66, 1979.

[37] R. Lockton and A. W. Fitzgibbon, "Real-time gesture recognition using deterministic boosting," presented at British Machine Vision Conference, 2002.

[38] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Hand Pose Estimation Using Hierarchical Detection," presented at International Workshop on Human-Computer Interaction, Prague, Czech Republic, 2004.

[39] B. D. R. Stenger, "Model-Based Hand Tracking Using A Hierarchical Baysian Filter," in *St. John's Colledge*. Cambridge: University of Cambridge, 2004, pp. 119.

[40] J. M. S. Dias, P. Nande, N. Barata, and A. Correia, "OGRE - open gestures recognition engine," presented at Proceedings of the 17th Brazilian Symposium on Computer Graphics and Image Processing, 2004.

[41] A. Shamaie and A. Sutherland, "Graph-Based Matching of Occluded Hand Gestures," presented at 30th Applied Imagery Pattern Recognition Workshop, (AIPR'01), 2001.

[42] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand Gesture Recognition using Multi-Scale Colour Features, Hierarchical Models and Particle Filtering," presented at Fifth IEEE International Conference on Automatic Face and Gesture Recognition, 2002.

[43] E. Ardizzone, A. Chella, and R. Pirrone, "An architecture for automatic gesture analysis " in *Proceedings of the working conference on Advanced visual interfaces* Palermo, Italy ACM Press, 2000 pp. 205-210

[44] M. A. Akbari and M. Nakajima, "A novel color region homogenization and its application in improving skin detection accuracy," in *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia.* Dunedin, New Zealand: ACM Press, 2005, pp. 269-272.

[45] F. Boussaid, D. Chai, and A. Bouzerdoum, "On-chip skin detection for color CMOS imagers," presented at MEMS, NANO and Smart Systems, 2003. Proceedings. International Conference on, 2003.

[46] L. Chen, J. Zhou, Z. Liu, W. Chen, and G. Xiong, "A skin detector based on neural network," presented at IEEE International Conference on Communications, Circuits and Systems, 2002.

[47] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," *International Journal of Computer Vision*, vol. 47, pp. 81-96, 2002.

[48] R. Kjeldsen and J. Kender, "Finding skin in color images " in *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition (FG '96)* IEEE Computer Society, 1996 pp. 312

[49] L. Sigal, S. Sclaroff, and V. Athitsos, "Skin Color-Based Video Segmentation under Time-Varying Illumination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 863-877, 2004.

[50] V. Vezhnevets, V. Sazonov, and A. Andreeva, "A Survey on Pixel-Based Skin Color Detection Techniques," presented at GraphiCon: The intenational conference on computer graphics, Moscow, Russia, 2003.

[51] J. Brand and J. S. Mason, "A comparative assessment of three approaches to pixel-level human skin-detection," presented at Proceedings of the15th International Conference on Pattern Recognition, 2000.

[52] F. Dadgostar and A. Sarrafzadeh, "An adaptive real-time skin detector based on Hue thresholding: A comparison on two motion tracking methods," *Pattern Recognition Letters*, vol. 27, pp. 1342-1352, 2006.

[53] J. Ruiz-del-Solar, A. Shats, and R. Verschae, "Real-time tracking of multiple persons," presented at Proceedings of the 12th International Conference on Image Analysis and Processing, 2003.

[54] L. Brethes, P. Menezes, F. Lerasle, and J. Hayet, "Face tracking and hand gesture recognition for human-robot interaction.," presented at Proc. of the International Conference on Robotics and Automation, 2004.

[55] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, vol. 57, pp. 137-154, 2004.

[56] K. W. Wong, K. M. Lam, and W. C. Siu, "A Robust Scheme for Live Detection of Human Face in Color Images," *Signal Processing: Image Communication*, vol. 18, pp. 103-114, 2003.

[57] R. E. Schapire, "The Boosting Approach to Machine Learning: An Overview," presented at MSRI Workshop on Nonlinear Estimation and Classification,, 2002.

[58] P. Viola and M. J. Jones, "Robust Real-time Object Detection," Cambridge Research Laboratory February 2001 2001.

[59] P. Viola and M. J. Jones, "Rapid Object Detection Using a Boosted Cascade of Simple features," presented at IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2001.

[60] M. J. Jones and P. Viola, "Fast Multi-View Face Detection," Mitsubishi Electronic Research Laboratories, Technical Report TR2003-96, July 2003 2003.

[61] S. Sheng, "A Study of AdaBoost in 3D Gesture Recognition," in *University of Toronto*. Toronto, 2003.

[62] P. Viola, M. J. Jones, and D. Snow, "Detecting Pedestrians Using Patterns of Motion and Appearance," presented at Ninth IEEE International Conference on Computer Vision, Nice, France, 2003.

[63] B. Wu, H. Ai, C. Huang, and S. Lao, "Fast rotation invariant multi-view face detection based on real Adaboost," presented at Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on, 2004.

[64] Z. Zhang, M. Li, S. Z. Li, and H. Zhang, "Multi-view face detection with FloatBoost," presented at Applications of Computer Vision, 2002. (WACV 2002). Proceedings. Sixth IEEE Workshop on, 2002.

[65] K. Matsui and H. Sato, "Evolutionary Feature Selection in Boosting," presented at International Conference on Systems, Man and Cybernetics, IEEE SMC 2004, The Hague, The Netherlands, 2004.

[66] A. L. C. Barczak, F. Dadgostar, and M. J. Johnson, "Real-time Hand Tracking Using the Viola and Jones Method," in *proceedings of the 7th International IEEE Conference on Image and Signal Processing: SIP 2005*, M. W. Marcelin, Ed. Honolulu, Hawaii: IASTED-SIP, 2005.

[67] E. J. Ong and R. Bowden, "A Boosted Classifier Tree for Hand Shape Detection," presented at The 6th International Conference on Automated Face and Gesture Recognition, 2004.

[68] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," presented at NIPS, 2000.

[69] T. Urano, T. Matsui, T. Nakata, and H. Mizoguchi, "Human Pose Recognition by Memory-Based Hierarchical Feature Matching," presented at International Conference on Systems, Man and Cybernetics, IEEE SMC 2004, The Hague, The Netherlands, 2004.

[70] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff, "3D Hand Pose Reconstruction Using Specialized Mappings," Boston Universtiy, Department of Computer Science, Boston, Technical Report 2000-22, 2000.

[71] R. Poppe, D. Heylen, A. Nijholt, and M. Poel, "Towards real-time body pose estimation for presenters in meeting environments," presented at Proceedings of the WSCG'2005, Plzen, Czech Republic, 2004.

[72] E. Loutas, N. Nikolaidis, and I. Pitas, "A mutual information approach to articulated object tracking," presented at Proceedings of the 2003 International Symposium on Circuits and Systems (SCAS '03), 2003.

[73] S. Lu, G. Huang, D. Samaras, and D. Metaxas, "Model-based Integration of Visual Cues for Hand Tracking," presented at Workshop on Motion and Video Computing (MOTION2002), 2002.

[74] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," Department of Computer Science University of North Carolina at Chapel Hill, Chapel Hill TR 95-041, April 5, 2004 2004.

[75] K. Imagawa, S. Lu, and S. Igi, "Color-Based Hands Tracking System for Sign Language Recognition," in *Proceedings of the 3rd International Conference on Face and Gesture Recognition*: IEEE Computer Society, 1998, pp. 462-467.

[76] P. KaewTraKulPong and R. Bowden, "An Adaptive Visual System for Tracking Low Resolution Colour Targets," presented at The British Machine Vision Conference (BMVC'01), Manchester, UK, 2001.

[77] K. Oka, Y. Sato, and H. Koike, "Real-time fingertip tracking and gesture recognition," *IEEE Computer Graphics and Applications*, vol. 22, pp. 64-71, 2002.

[78] K. Oka, Y. Sato, and H. Koike, "Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems," presented at Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition, 2002.

[79] T. Wang, Q. Diao, Y. Zhang, G. Song, C. Lai, and G. R. Bradski, "A Dynamic Bayesian Network Approach to Multi-cue based Visual Tracking," presented at 17th International Conference on Pattern Recognition (ICPR'04), Cambridge, United Kingdom, 2004.

[80] C. R. Wern, B. P. Clarkson, and A. P. Pentland, "Understanding Purposeful Human Motion," presented at Fourth IEEE international conference on Automatic Face and Gesture Recognition, Grenoble, France, 2000.

[81] Z. Ghahramani, "An Introduction to Hidden Markov Models and Bayesian Networks," *Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, pp. 9-42, 2001.

[82] J. Schlenzig, E. Hunter, and R. Jain, "Vision based hand gesture interpretation using recursive estimation," presented at Proceedings of the Twenty-Eighth Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 1994.

[83] T. Westeyn, H. Brashear, A. Atrash, and T. Starner, "Georgia Tech Gesture Toolkit: Supporting Experiments in Gesture Recognition," presented at Proceedings of the International Conference on Perceptive and Multimodal User Interfaces., Vancouver, B.C., Canada, 2003.

[84] Y. Zhu, H. Ren, G. Xu, and X. Lin, "Toward real-time human-computer interaction with continuous dynamic hand gestures," presented at Proceedings. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000.

[85] A. Wu, M. Shah, and N. d. V. Lobo, "A Virtual 3D Blackboard: 3D Finger Tracking Using a Single Camera," in *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition*: IEEE Computer Society, 2000, pp. 536-543.

[86] A. Licsar and T. Sziranyi, "Hand Gesture Recognition in Camera-Projector System," in *Lecture Notes in Computer Science, International Workshop on Human-Computer Interaction*, vol. LNCS 3058, 2004, pp. 83-93.

[87] J.-C. Lementec and P. Bajcsy, "Recognition of arm gestures using multiple orientation sensors: gesture classification," presented at Proceedings of The 7th International IEEE Conference on Intelligent Transportation Systems, 2004.

[88] G. R. Bradski and J. W. Davis, "Motion segmentation and pose recognition with motion history gradients," *Machine Vision and Applications*, vol. 13, pp. 174-184, 2002.

[89] M. Betke, J. Gips, and P. Fleming, "The Camera Mouse: visual tracking of body features to provide computer access for people with severe disabilities," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 10, pp. 1-10, 2002.

[90] A. D. Wilson and E. Cutrell, "FlowMouse: A Computer Vision-Based Pointing and Gesture Input Device," in *Proceedings of the International Conference on Human-Computer Interaction (INTERACT 2005)*, vol. 3585, M. F. Costabile and F. Paterno, Eds. Rome, Italy: Springer-Verlag Berlin Heidelberg, 2005, pp. 565.

[91] A. D. Wilson and N. Oliver, "GWindows: Robust Stereo Vision for Gesture-Based Control of Windows," presented at Proceedings of the 5th international conference on Multimodal interfaces, Vancouver, British Columbia, Canada, 2003.

[92] C. Cadoz, "Le geste canal de communication homme/machine," *Technique et Science Informatique*, vol. 13, pp. 31, 1994.

[93] K. R. Thorissonm, "Multimodal interaction with humanoid computer characters," presented at Proceedings of the Conference on Lifelike Computer Characters, Snowbird, Utah, USA, 1995.

[94] M. Brereton, N. Bidwell, J. Donovan, B. Campbell, and J. Buur, "Work at Hand: An exploration of gesture in the context of work and everyday life to inform the design of gestural input devices," presented at Fourth Australian conference on user interface, 2002.

[95] D. McNeill, *Hand and Mind*: The University of Chicago Press, 1992.

[96] J. Cassell, "A framework for gesture generation and interpretation," in *Computer Vision for Human-Machine Interaction*, R. Cipolla & A. Pentland ed: Cambridge University Press, 1998.

[97] O. N. Kwon, M. K. Ju, J. S. Park, K. H. Cho, and K. H. Ewha, "Gesture in the Context of Mathematical Argumentation," presented at Proceedings of the 27th conference of the International group for the Psychology of Mathematics Education held jointly with the 25th Conference of PME-NA, Honolulu, USA, 2003.

[98] M. Alibali and A. diRusso, "The function of gesture in learning to count: More than keeping track," *Cognitive Development*, vol. 14, pp. 37-56, 1999.

[99] L. D. Edwards, "Metaphors and gesture in fraction talk," presented at The 4th Congress of the European Society for Research in Mathematics Education, Spain, Sant Feliu de Guixols, 2005.

[100] J. I. Habermas, *The Theory of Communicative Action I*. London: Heinemann, 1984.

[101] J. Short, E. Williams, and B. Christie, *The Social Psychology of Telecommunications.* Chichester: Wiely, 1976.

[102] R. L. Daft and R. H. Lengel, "Information richness: a new approach to managerial behaviour and organizational design," in *Research in Organizational Behaviour*, L. L. Cummings & B. Staw ed. Greenwich, CT: JAI Press, 1984.

[103] H. H. Clark and E. F. Schaefer, "Collaborating on contributions to conversations," *Language and Cognitive Processes*, vol. 2, pp. 19-41, 1987.

[104] A. F. Monk, "Computer Mediated Communication: Clark's common ground theory," in *HCI Models, Theories, and Frameworks: Toward a Multidisciplinary Science*, J. M. Carroll ed: Morgan Kaufmann, 2003.

[105] C. J. Patterson and J. M. Cosgrove, "Nonverbal indicants of comprehensive and noncomprehensive in children," *Developmental psychology*, vol. 16, pp. 38-48, 1980.

[106] I. J. Ko and H. I. Choi, "Extracting the hand region with the aid of a tracking facility," *Electronics Letters*, vol. 32, pp. 1561-1563, 1996.

[107] D. Butler, S. Sridharan, and V. Chandran, "Chromatic colour spaces for skin detection using GMMs," presented at Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP, 2002.

[108] S. Srisuk and W. Kurutach, "A new robust face detection in color images," presented at Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition, 2002.

[109] Q. Zhu, K. T. Cheng, C. T. Wu, and Y. L. Wu, "Adaptive learning of an accurate skin-color model," presented at Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004.

[110] G. R. Bradski, "Computer Vision Face Tracking For Use in a Perceptual User Interface," *Intel Technology Journal*, vol. 2, pp. 1-15, 1998.

[111] M. C. Shin, K. I. Chang, and L. V. Tsap, "Does colorspace transformation make any difference on skin detection?," presented at Proceedings. Sixth IEEE Workshop on Applications of Computer Vision (WACV), 2002.

[112] A. Corradini and H. M. Gross, "Camera-based gesture recognition for robot control," presented at IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN, Como , Italy, 2000.

[113] T. Gevers, "Robust Segmentation and Tracking of Colored Objects in Video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, pp. 1-6, 2004.

[114] N. H. Reyes and E. P. Dadios, "Dynamic Object Recognition Using Fuzzy Logic," *Journal of Advanced Computational Intelligence and Intelligent Information*, vol. 8, pp. 29-38, 2004.

[115] J. Ruiz-del-Solar and R. Verschae, "Skin detection using neighbourhood information," presented at Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004.

[116] J. Ruiz-del-Solar and R. Verschae, "Robust skin segmentation using neighborhood information," presented at ICIP '04. International Conference on Image Processing, 2004.

[117] R. C. K. Hua, L. C. De Silva, and P. Vadakkepat, "Detection and tracking of faces in real-time environments," presented at CISST'2002, International Conference on Imaging Science, Systems and Technology, Las Vegas, United States, 2002.

[118] A. Albiol, L. Torres, and E. J. Delp, "Optimum color spaces for skin detection," presented at Proceedings of the IEEE International Conference on Image Processing, Thessaloniki, 2001.

[119] Y. Wu and T. S. Huang, "Non-stationary Color Tracking for Vision-based Human Computer Interaction," *IEEE Transactions on Neural Networks*, vol. 13, pp. 948-960, 2002.

[120] H. Zheng, M. Daoudi, and B. Jedynak, "From maximum entropy to belief propagation : an application to skin detection," presented at In Proceedings of the 2004 British Machine Vision Conference, London, United Kingdom, 2004.

[121] D. Brown, I. Craw, and J. Lewthwaite, "A SOM Based Approach to Skin Detection with Application in Real Time Systems," presented at Proceedings of the British Machine Vision Conference, 2001.

[122] J. Y. Lee, M. Perrone, J. Costeria, and J. Santos-Victor, "An Elliptical Boundary Model for Skin Color Detection," presented at In Proceedings of the International Conference on Imaging Science, Systems and Technology, 2002.

[123] P. Kuchi, P. Gabbur, P. S. Bhat, and D. S. Sumam, "Human Face Detection and Tracking using Skin Color Modeling and Connected Component Operators," *IETE Journal of Research*, vol. 38, pp. 289-293, 2002.

[124] B. Menser and M. Wien, "Segmentation and tracking of facial regions in color image sequences," presented at Proceedings of SPIE - Visual Communications and Image Processing, 2000.

[125] F. C. M. Kjeldsen, *Visual interpretation of hand gestures as a practical interface modality*: Columbia University, 1997.

[126] M. Kolsch and M. Turk, "Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration," presented at Proceedings of the Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04), Washington D.C., USA, 2004.

[127] A. P. Azcarraga, M. H. Hsieh, S. L. Pan, and R. Setiono, "Extracting Salient Dimensions for Automatic SOM Labeling," *IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews*, vol. 35, pp. 595-599, 2005.

[128] L. Sigal, S. Sclaroff, and V. Athitsos, "Estimation and Prediction of Evolving Color Distribution for Skin Segmentation Under Varying Illumination," presented at IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2000, 2000.

[129] M. Storring, H. J. Andersen, and E. Granum, "Skin colour detection under changing lighting conditions," presented at Proceedings of the 7th Symposium on Intelligent Robotics Systems, Coimbra, Portugal, 1999.

[130] J. Richefeu and A. Manzanera, "A new hybrid differential filter for motion detection," presented at International Conference on Computer Vision and Graphics, ICCVG'04, Warsaw, Poland, 2004.

[131] A. Manzanera and J. Richefeu, "A robust and computionally efficient motion detection algorithm based on Sigma-Delta background estimation," presented at Processing of the Indian conference on Computer Vision, Graphics and Image (ICVGIP'04), Kolkata, India, 2004.

[132] M. H. Yang, N. Ahuja, and M. Tabb, "Extraction of 2D motion trajectories and its application to hand gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1061-1074, 2002.

[133] S. Jayaram, S. Schmugge, M. C. Shin, and L. V. Tsap, "Effect of colorspace transformation, the illuminance component, and color modeling on skin detection," presented at Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), 2004.

[134] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," presented at Proceeding of 7th International Joint Conference on Artificial Intelligence (IJCAI), 1981.

[135] Intel, "Open Computer Vision library (http://www.intel.com/research/mrl/research/opencv/)," 0.9.5 ed, 2005.

[136] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on Information Theory*, vol. 21, pp. 32-40, 1975.

[137] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 790-799, 1995.

[138] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-Based Object Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, 2003.

[139] D. Comaniciu and P. Meer, "Mean Shift Analysis and Applications," presented at Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 1999.

[140] D. Comaniciu and P. Meer, "Mean Shift: A Robust Approach Toward Feature Space Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603-619, 2002.

[141] D. Comaniciu, V. Ramesh, and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift," presented at IEEE Computer Vision and Pattern Recognition, 2000.

[142] J. G. Allen, R. Y. D. Xu, and J. S. Jin, "Object Tracking Using CamShift Algorithm and Multiple Quantized Feature Spaces," presented at Workshop on Visual Information Processing, Conferences in Research and Practice in Information Technology, Sydney, Australia, 2003.

[143] R. Wang, Y. Chen, and T. S. Huang, "Basis Pursuit for Tracking," presented at Proceedings of the IEEE International Conference on Image Processing (ICIP'01), Thessaloniki, Greece, 2001.

[144] A. Djouadi, O. Snorrason, and F. D. Garber, "The Quality of Training-Sample Estimates of the Bhattacharayya Coefficient," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 92-97, 1990.

[145] C. Yang, R. Duraiswami, and L. Davis, "Efficient Mean-Shift Tracking via a New Similarity Measure," presented at Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 2005.

[146] R. Y. D. Xu, J. G. Allen, and J. S. Jin, "Robust Mean-shift Tracking with Extended Fast Colour Thresholding," presented at Proceedings of the International Symposium on Intelligent Multimedia, Video & Speech Processing, Hong Kong, 2004.

[147] J. Sherrah and S. Gong, "Tracking Discontinuous Motion Using Bayesian Inference," presented at ECCV, 2000.

[148] R. T. Collins, "Mean-shift Blob Tracking through Scale Space," presented at IEEE Conference on Computer Vision and Pattern Recognition, 2003.

[149] C. Shan, Y. Wei, T. Tan, and O. Ojardias, "Real Time Hand Tracking by Combining Particle Filtering and Mean Shift," presented at Sixth IEEE International Conference on Automatic Face and Gesture Recognition (FGR 2004), Seoul, Korea, 2004.

[150] T. Kanade, "Development of a Video-Rate Stereo Machine," presented at Proceedings of the 1994 ARPA Image Understanding Workshop (IUW'94), 1994.

[151] T. G. Dietterich, "Machine learning for sequential data: A review," in *Proceedings of the Fourth International Workshop on Statistical Techniques in Pattern Recognition*, vol. 2396, *Lecture Notes in Computer Science*, T. Caelli, Ed.: Springer-Verlag, 2002, pp. 15-30.

[152] T. Watanabe and M. Yachida, "Real-time gesture recognition using KL expansion of image sequence," presented at Intelligent Robots and Systems, 1997. IROS '97, Proceedings of the 1997 IEEE/RSJ International Conference, 1997.

[153] T. Watanabe and M. Yachida, "Real time gesture recognition using eigenspace from multi-input image sequences," presented at Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition, Nara, Japan, 1998.

[154] V. M. Mantyla, J. Mantyjarvi, T. Seppanen, and E. Tuulari, "Hand gesture recognition of a mobile device user," presented at Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2000), 2000.

[155] K. Ogawara, S. Iba, T. Tanuki, H. Kimura, and K. Ikeuchi, "Acquiring hand-action models by attention point analysis," presented at IEEE International Conference on Robotics and Automation (ICRA), 2001.

[156] J. R. New, E. Hasanbelliu, and M. Aguilar, "Facilitating User Interaction with Complex Systems via Hand Gesture Recognition," presented at Proceedings of the 2003 Southeastern ACM Conference, Savannah, GA, 2003.

[157] S. Perrin, A. Cassinelli, and M. Ishikawa, "Gesture recognition using laser-based tracking system," presented at Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on, 2004.

[158] T. Darrell and A. Pentland, "Space-time gestures," presented at Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVRP '93), New York, NY, USA, 1993.

[159] J. Eisenstein, S. Ghandeharizadeh, L. Golubchik, C. Shahabi, D. Yan, and R. Zimmermann, "Multi-Layer Gesture Recognition: An Experimental Evaluation," Department of Computer Science, University of Southern California, Technical Report 2002.

[160] J. Eisenstein, S. Ghandeharizadeh, L. Golubchik, C. Shahabi, D. Yan, and R. Zimmermann, "Device independence and extensibility in gesture recognition," presented at Proceedings of the IEEE Conference on Virtual Reality, 2003.