uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Qing Chen

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Real-time Vision-based Hand Tracking and Gesture Recognition

TITRE DE LA THÈSE / TITLE OF THESIS

Nicolas Georganas

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Emil Petriu

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Robert Laganière

Dorina Petriu

WonSook Lee

Kostas Plataniotis

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Real-Time Vision-Based Hand Tracking and Gesture Recognition

by

Qing Chen

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-41629-7
Our file Notre référence
ISBN: 978-0-494-41629-7

Abstract

Hand gestures can be used for natural and intuitive human-computer interaction. To achieve this goal, computers should be able to visually recognize hand gestures from video input. However, vision-based hand tracking and gesture recognition is an extremely challenging problem due to the complexity of hand gestures, which are rich in diversities due to high degrees of freedom involved by the human hand. On the other hand, computer vision algorithms are notoriously brittle and computation intensive, which make most current gesture recognition systems fragile and inefficient.

This thesis proposes a new architecture to solve the problem of real-time vision-based hand tracking and gesture recognition with the combination of statistical and syntactic analysis. The fundamental idea is to use a divide-and-conquer strategy based on the hierarchical composition property of hand gestures so that the problem can be decoupled into two levels. The low-level of the architecture focuses on hand posture detection and tracking with Haar-like features and the AdaBoost learning algorithm. The Haar-like features can effectively catch the appearance properties of the hand postures. The AdaBoost learning algorithm can significantly speed up the performance and construct an accurate cascade of classifiers by combining a sequence of weak classifiers. To recognize different hand postures, a parallel cascades structure is implemented. This structure achieves real-time performance and high classification accuracy. The 3D position of the hand is recovered according to the camera's perspective projection. To make the system robust against cluttered backgrounds, background subtraction and noise removal are applied.

For the high-level hand gestures recognition, a stochastic context-free grammar (SCFG) is used to analyze the syntactic structure of the hand gestures with the terminal strings converted from the postures detected by the low-level of the architecture. Based on the similarity measurement and the probabilities associated with the production rules, given an input string, the corresponding hand gesture can be identified by looking for the production rule that has the greatest probability to generate this string. For the hand motion analysis, two SCFGs are defined to analyze two structured hand gestures with different trajectory patterns: the rectangle gesture and the diamond gesture. Based on the different probabilities associated with these two grammars, the SCFGs can effectively disambiguate the distorted trajectories and classify them correctly.

An application of gesture-based interaction with a 3D gaming virtual environment is implemented. With this system, the user can navigate the 3D gaming world by driving the avatar car with a set of hand postures. When the user wants to manipulate the

virtual objects, he can use a set of hand gestures to select the target traffic sign and open a window to check the information of the correspondent learning object. This application demonstrates the gesture-based interface can achieve an improved interaction, which are more intuitive and flexible for the user.

Acknowledgements

First of all, I thank my supervisors Professor Nicolas D. Georganas and Professor Emil M. Petriu for their guidance, advice and encouragement throughout my Ph.D. study. I have benefited tremendously from their vision, technical insights and profound thinking. They are my great teachers.

I wish to thank the members of the DiscoverLab for their suggestions and helps on my work. I owe lots of thanks to Francois Malric, who helped so much on my cameras, computers and software. I also wish to thank Professor Abdulmotaleb El Saddik, who suggested and helped the project of navigating the virtual environment by hand gestures. I thank Dr. Xiaojun Shen and ASM Mahfujur Rahman for their cooperations in this project.

Last but not least, I thank my parents and families, I would never have made it without them.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Human-computer interfaces (HCI) have developed from text-based interfaces through 2D graphical-based interfaces, multimedia-supported interfaces, to fully fledged multimodel-based 3D virtual environment (VE) systems. While providing a new sophisticated paradigm for communication, learning, training and entertaining, VEs also provide new challenges for human-computer interaction. The traditional 2D HCI devices such as keyboards and mice are not adequate for the latest VE applications. Instead, VE systems provide the opportunity to integrate different communication modalities and sensing technologies together to provide a more immersive user experience [1, 2]. As shown in Figure 1.1, devices that can sense body position and orientation, speech and sound, facial and gesture expression, haptic feedback and other aspects of human behavior or state can be used for more powerful and effective interactions between human and computers.

To achieve natural and immersive human-computer interaction, the human hand could be used as an interface device [3, 4, 5]. Hand gestures are a powerful human-to-human communication channel, which forms a major part of information transfer in our everyday life. Hand gestures are an easy to use and natural way of interaction. For example, sign languages have been used extensively among speech-disabled people. People who can speak also use many kinds of gestures to help their communications. However, the expressiveness of hand gestures has not been fully explored for human-computer interaction. Compared with traditional HCI devices, hand gestures are less intrusive and more convenient for users to interact with computers and explore the 3D virtual worlds [6]. Hand gestures can be used in a wide range of applications such as

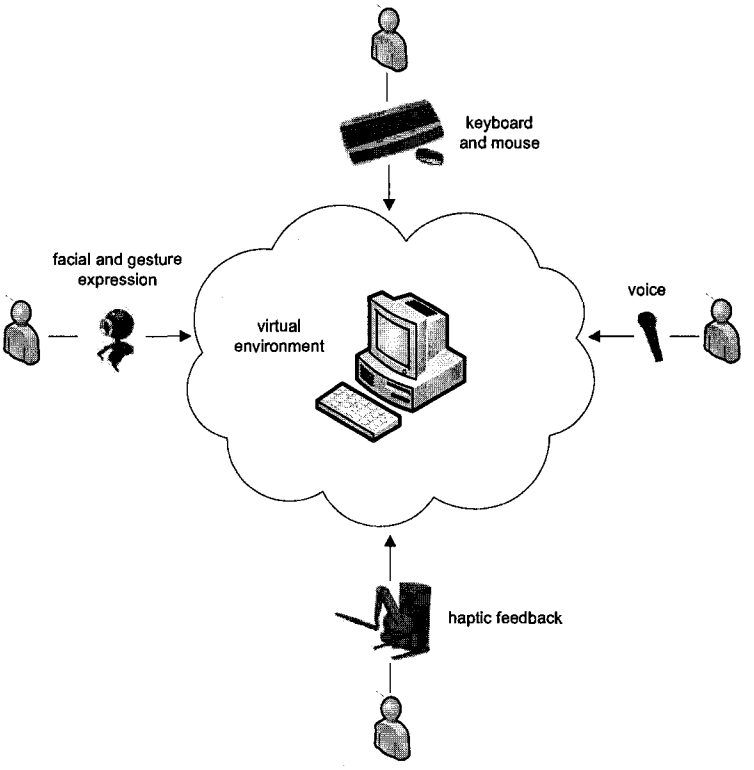


Figure 1.1: The multimodel-based HCI for VE systems.

conventional human-computer interactions through linguistic gestures (*e.g.* various sign languages) and objects manipulation in VEs.

To use the human hand as a natural HCI device, data gloves such as the Cyber-Glove [7] shown in Figure 1.2 have been used to capture hand motions [8, 9, 10]. With the attached sensors, the joint angles and spatial positions of the hand can be measured directly from the glove. However, the data glove and its attached wires are still inconvenient and awkward for users to wear. Moreover, the cost of the data glove is often too expensive for regular users.

Vision-based hand gesture recognition systems can identify different hand gestures from video input and use them as artificial commands, which computers can understand and respond to [11]. With the video camera as input device and computer vision and image processing techniques, vision-based hand gesture recognition systems are more convenient and efficient because of the uncumbersome hardware. Vision-based hand gesture recognition systems can provide an intuitive communication channel for human-computer interaction. For example, the user can navigate a 3D virtual environment

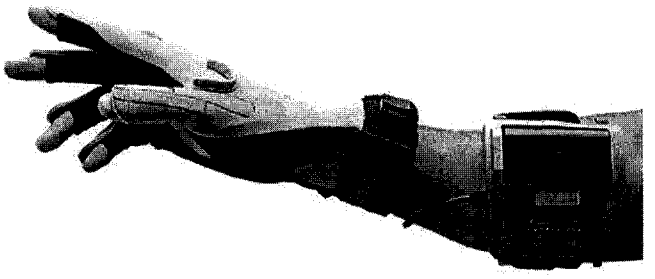


Figure 1.2: The CyberGlove from the Immersion Corporation (from [7]).

and manipulate the virtual objects with a set of gesture commands without the help of keyboards, mice or joysticks [12, 13, 14]. Early research on vision-based hand tracking and gesture recognition usually needs the help of markers or colored gloves [15, 16]. In current state-of-the-art vision-based hand tracking and gesture recognition techniques, research is more focused on tracking the bare hand and identify hand gestures without the help of any markers and gloves.

1.2 Motivation

The latest computer vision technologies and the advanced computer hardware capacity make real-time, accurate and robust hand tracking and gesture recognition promising. Many different approaches have been proposed such as appearance-based approaches and 3D hand model-based approaches [17]. Most of these approaches deal the hand gesture as a whole object and try to extract the corresponding mathematical description from a large number of training samples. These approaches analyze hand gestures without breaking them into their constituent atomic elements that could simplify the complexity of hand gestures. As a result, many current approaches are still limited by the lack of speed, accuracy and robustness. They are either too fragile or demand too many prerequisites such as markers, clean backgrounds or complex camera calibration steps, and thus make the gesture interaction indirect and unnatural. Currently there is no real-time vision-based hand tracking and gesture recognition system that can track and identify hand gestures in a fast, accurate, robust and easily accessible manner.

The goal of this work is to build a real-time 3D hand tracking and gesture recognition system for the purpose of human-computer interaction. To achieve this goal, the characters of hand gestures need to be taken into account and the principles that can improve the system's performance in terms of speed, accuracy and robustness need to be applied.

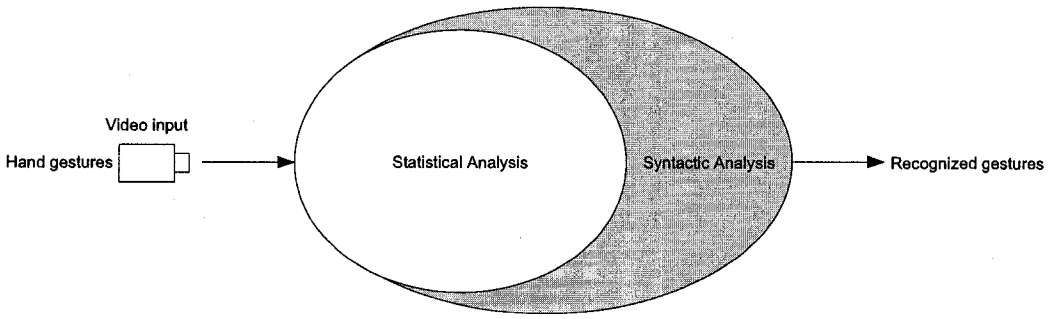


Figure 1.3: The hybrid approach for hand gesture recognition.

Hand gestures are complex human actions with rich structure information, which can be exploited for a syntactic pattern recognition approach. To fully use the composite property of hand gestures, we prefer a divide-and-conquer strategy by using a hybrid approach based on statistical and syntactic analysis as illustrated in Figure 1.3. This approach decomposes the complex hand gesture recognition problem into their atomic elements that would be easier to handle. The statistical analysis is responsible for the extraction of the primitives of hand gestures such as hand postures and motion directions. The syntactic analysis focuses on the structure analysis for hand gestures based on the extracted primitives and a set of grammars which defines the composition rules. The prerequisite for our system is that it will focus on bare hand gestures without the help of any markers or colored gloves. Meanwhile, the system will use only one regular video camera (*e.g.* web-camera) as input device to be cost-efficient. The performance requirements to be met by the 3D hand tracking and gesture recognition system are:

- *Real-time performance:* by real-time performance, we mean the system must be able to analyze the image at the frame rate of the input video to give the user instant feedback of the recognized gesture. For example, with the video input rate at 15 frames per second and the frame resolution at 320 $\times$ 240, the gesture recognition system must have the minimum speed of processing fifteen 320 $\times$ 240 images every second.

- *Accuracy:* a hand gesture recognition system should be able to tolerate some mistakes and errors. However, it still needs to be accurate enough in order to be viable. For instance, the system should be able to achieve a detection rate above 90% while maintaining a low false positive rate for each hand gesture. Meanwhile, the system should also be able to recognize different hand gestures without confusion among

them.

- *Robustness:* by robustness, we mean the system should be able to track the 3D hand position and recognize different hand gestures successfully under different lighting conditions and cluttered backgrounds. The system should also be robust against in-plane and out-of-plane image rotations.

- *Scalability:* by scalability, we mean a large gesture vocabulary can be included with a small number of primitives. The composition of different gesture commands can be easily controlled by the user.

- *User-independence:* by user-independence, we mean the system should be able to work for different users rather than a specific user. The system should recognize hand gestures performed by human hands of different sizes and colors.

1.3 Thesis Organization

This thesis includes 7 chapters:

- Chapter 1 introduces the background and motivations of this work. The goals to be achieved by the work are also given.

- Chapter 2 first introduces two fundamental concepts – *hand posture* and *hand gesture*. A comprehensive literature review is given based on two dichotomies for vision-based hand tracking and gesture recognition approaches: appearance-based approaches versus 3D hand model-based approaches; statistical approaches versus syntactic approaches. The pros and cons of these different approaches are compared and discussed.

- Chapter 3 proposes the overall architecture for the system: a two-level architecture which decouples hand gesture recognition into low-level hand posture detection and tracking and high-level hand gesture recognition and motion analysis. The low-level of the architecture detects hand postures using a statistical approach based on Haar-like features and a boosting algorithm. The high-level of the architecture employs a syntactic approach to recognize hand gestures and analyze hand motions based on grammars.

- Chapter 4 focuses on the low-level of the architecture: hand posture detection and tracking. A set of Haar-like features and the AdaBoost learning algorithm are introduced. The robustness of Haar-like features and the AdaBoost learning algorithm against image rotations and lighting variations are evaluated. A parallel cascades structure is implemented to recognize four different hand postures. The performance of the trained cascades of classifiers is tested. The camera's perspective projection is used to recover the 3D position of the hand. To achieve robustness against cluttered backgrounds, background subtraction and noise removal are applied.

- Chapter 5 presents the high-level hand gesture recognition and motion analysis using stochastic context-free grammars (SCFG). With the input string converted from the postures detected by the low-level of the architecture, the hand gestures are recognized based on a defined gesture SCFG. For the global hand motion analysis, two SCFGs are used to analyze two structured hand gestures with different trajectory patterns. Based on the different probabilities associated with these two grammars, the SCFGs can effectively disambiguate the distorted trajectories and classify them correctly.

- Chapter 6 demonstrates the effectiveness of the gesture recognition system for human-computer interaction. An application of the gesture interface for interaction with a 3D gaming VE is presented. With this system, the user can navigate the 3D gaming world by driving the avatar car with a set of hand postures. When the user wants to manipulate the virtual objects, he can use a set of hand gestures to select the target traffic sign and open a window to check the information of the correspondent learning object.

- Chapter 7 concludes the thesis by summarizing the contributions of this work as well as the possible improvements for future work.

1.4 Contributions

This thesis proposes a new architecture to solve the problem of real-time vision-based hand tracking and gesture recognition with the combination of statistical and syntactic analysis. The fundamental idea is to use a divide-and-conquer strategy based on the hierarchical composition property of hand gestures so that the problem can be decoupled

into two levels. The low-level of the architecture focuses on hand posture detection and tracking with Haar-like features and the AdaBoost learning algorithm. The high-level of the architecture focuses on the hand gesture recognition and motion analysis using syntactic analysis based on SCFGs. The original contributions of this thesis are:

1. A two-level system architecture is implemented, which combines the advantages of statistical and syntactic pattern recognition approaches effectively, and achieves real-time, accurate and robust hand tracking and gesture recognition with one camera as input device.

2. A parallel cascade structure for the architecture's low-level is implemented using the AdaBoost learning algorithm and a set of Haar-like features. This structure can correctly extract a set of hand postures track the hand motion in 3D in real-time.

3. The hand gestures are analyzed base on a SCFG, which defines the composite properties based on the constituent hand postures. The assignment of the probability to each production rule of the SCFG can be used to control the "wanted" gestures and the "unwanted" gestures. Smaller probability could be assigned to the "unwanted" gestures while greater value could be assigned to "wanted" gestures so that the resulting SCFG would generate the "wanted" gestures with higher probabilities.

4. For hand motion analysis, with the uncertainty of hand trajectories, the ambiguous versions can be identified by looking for the SCFG that has the higher probability to generate the input string. The motion patterns can be controlled by adjusting the probabilities associated with the production rules so that the resulting SCFG would generate the standard motion patterns with higher probabilities.

1.5 Publications Arising from the Thesis

1. Q. Chen, N. D. Georganas and E. M. Petriu, "Hand gesture recognition using Haar-like features and a stochastic context-free grammar," *IEEE Transactions on Instrumentation and Measurement*, accepted.

2. Q. Chen, A. M. Rahman, X. Shen, A. El Saddik and N. D. Georganas, "Navigating a 3D virtual environment of learning objects by hand gestures," *International Journal of Advanced Media and Communication*, vol.1, no.4, pp.351-368, 2007.

3. Q. Chen, E. M. Petriu and N. D. Georganas, "3D hand tracking and motion analysis with a combination approach of statistical and syntactic analysis, in *Proc. IEEE Intl. Workshop on Haptic, Audio and Visual Environments and their Applications*, 2007.

4. Q. Chen, N. D. Georganas, and E. M. Petriu, "Real-time vision-based hand gesture recognition using Haar-like features," in *Proc. IEEE Instrumentation and Measurement Technology Conference*, 2007.

5. Q. Chen, A. M. Rahman, A. El-Sawah, X. Shen, A. El Saddik and N. D. Georganas, "Accessing learning objects in virtual environment by hand gestures and voice," in *Proc. $I^2LOR-06$, 3rd Annual Scientific Conference, LORNET Research Network* , 2006.

Chapter 2

Literature Review

2.1 Introduction

The human hand is a complex articulated structure consisting of many connected links and joints. The skeleton structure and the joints of the human hand are shown in Figure 2.1 [18]. Each finger consists of three joints whose names are indicated. For the

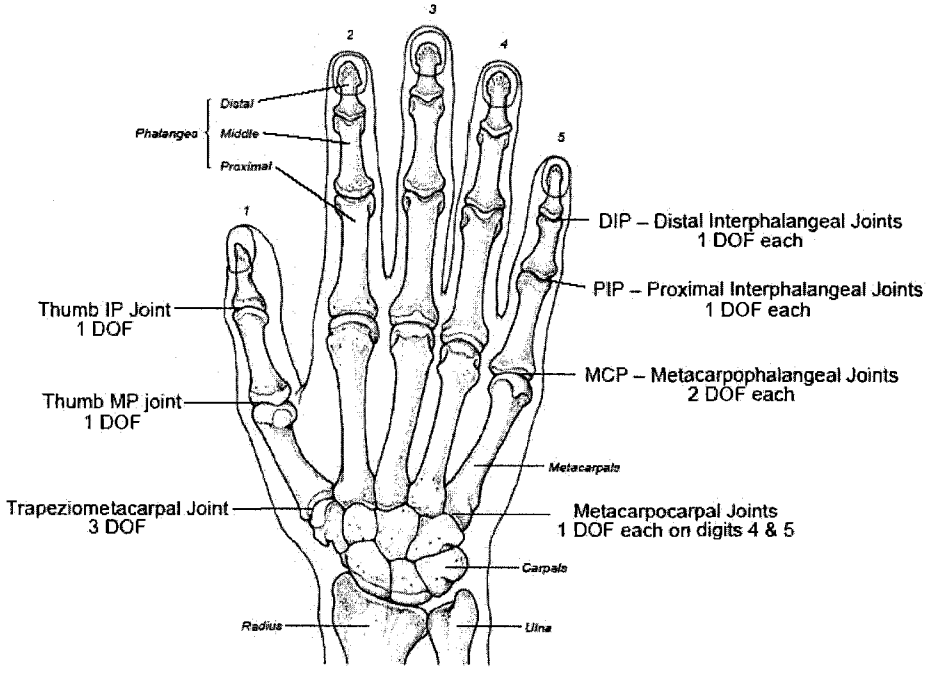


Figure 2.1: The hand skeleton and joints with associated DOF (from [18]).

thumb, there are 3 degrees of freedom (DOF) for the trapeziometacarpal joint, and 1 DOF for the thumb IP joint and the thumb MP joint respectively. For the rest four fingers, there are 2 DOF for MCP joints, and 1 DOF for each PIP joints and DIP joints. There is 1 DOF for each metacarpocarpal joint at the bottom of the ring finger and the little finger. Considering the 4 DOF for the hand wrist, there are 27 DOF for the human hand all together [6].

Due to the high DOF of the human hand, hand gesture recognition becomes a very challenging problem. To better understand hand gestures and hand motions, there are two important concepts need to be cleared [19, 20]:

- *Hand Posture:* a hand posture is a static hand pose and its current location without any movements involved.
- *Hand Gesture:* a hand gesture is a sequence of hand postures connected by continuous hand or finger movements over a short period of time.

A hand posture is defined as a static hand pose. For example, making a fist and holding it in a certain position is considered a hand posture. A hand gesture is defined as a dynamic movement, such as waving goodbye. The dynamic movement of hand gestures includes two aspects: global hand motions and local finger motions [21]. Global hand motions change the position or orientation of the hand. Local finger motions involve moving the fingers in some way but without change in the position and orientation of the hand. For example, moving the index finger back and forth to urge someone to come closer. Compared with hand postures, hand gestures can be viewed as complex composite hand actions constructed by global hand motions and a series of hand postures that act as transition states. To further explain the difference between hand postures and hand gestures, consider the signs for "d" and "z" in American Sign Language shown in Figure 2.2 [22]. According to above definitions, the sign for "d" is a hand posture,

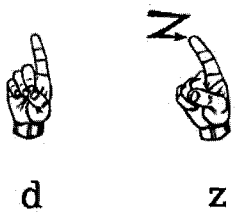


Figure 2.2: The signs for "d" and "z" in ASL (from [22]).

but the sign for "z" is a hand gesture.

To effectively explore the expressiveness of the hand gestures for HCI applications, a set of gestural commands need to be carefully selected. Lenman *et al.* suggested that the design space for gestural commands can be characterized along three dimensions: the cognitive aspect, the articulatory aspect and the technological aspect [23]. Based on Lenman's design space, we propose our standards for the selection of gesture commands:

- *The intuition aspect* means the selected gestures should be intuitive and comfortable for the user to learn and to remember. The gestures should be straightforward so that least effort will be required for the user to learn the gestures. The user should be able to use their natural hand configurations and not be required to learn any specific or complex hand configurations, which are very easy to cause fatigue and make the user uncomfortable.

- *The articulatory aspect* means the selected gestures should be easy for recognition and do not cause confusions for the user. Gestures involving complicated hand poses and finger movements should be avoided due to the difficulty to articulate and repeat.

- *The technology aspect* refers to the fact that in order to be viable, the selected gestures must take into account the properties of employed algorithms and techniques. The required data and information can be extracted and analyzed from the selected gesture commands without causing excessive computation cost for the employed approach.

To recognize hand gestures, we need a good set of characteristic features and the knowledge of how they interrelate in representing hand gestures. Many algorithms and approaches have been proposed. We summarized the latest trends and ideas for vision-based hand tracking and gesture recognition systems proposed by different researchers in Table 2.1.

One dichotomy to differentiate vision-based hand tracking and gesture recognition algorithms is appearance-based approaches versus 3D hand model-based approaches. Another dichotomy is based on the methodology used to describe hand gestures, which could be statistical approaches or syntactic approaches. We will give detailed introductions of these different approaches in the following sections.

Table 2.1: Latest vision-based hand tracking and gesture recognition systems.

Authors/Year	Task	Approach	Category	Speed	Restrictions
Barczak *et al.* 2005 [24]	Real-time hand tracking	The Viola-Jones method	Appearance-based Statistical	Real-time	Single posture (hand palm)
Zhou *et al.* 2005 [25]	Articulated object (*e.g.* body/hand postures) recognition	Inverted indexing in an image database using local image features	Hand model-based Statistical	$3s$/query	Non real-time
Derpanis *et al.* 2004 [26]	Hand gesture recognition from a monocular temporal sequence of images	Use linguistic analysis to decompose dynamic gestures into their static and dynamic components	Appearance-based Syntactic	$8s$/frame	Non real-time
Lin *et al.* 2004 [27]	Tracking the articulated hand motion in a video sequence	Searching for an optimal motion estimate in a high dimensional configuration space	Hand model-based Statistical	$2s$/frame	Non real-time
Ye *et al.* 2004 [28]	Classify manipulative and controlling gestures	Compute 3D hand appearance using a region-based coarse stereo matching algorithm	Stereo vision statistical	Real-time	Two cameras, calibration required
Kölsch *et al.* 2004 [29]	Fast tracking for non-rigid and highly articulated objects such as hands	Use flock of KLT features/colors to facilitate 2D hand tracking and posture recognition from a monocular view	Appearance-based statistical	Real-time	Strict requirement on hand pose configuration for recognition
Bowden *et al.* 2004 [30]	Sign language recognition	Use a two-stage classification by structuring the classification model around a linguistic definition of signed words and the Markov chain to encode temporal transitions	Appearance-based Syntactic	Real-time	Clean background, long-sleeve shirt is required for the user
Tomasi *et al.* 2003 [31]	3D tracking for hand finger spelling motions	Use a combination of 2D image classification and 3D motion interpolation	Hand model-based Statistical	Real-time	Clean background
Wu *et al.* 2002 [32]	Real-time face/hand localization	Use color-based image segmentation and non-stationary color-based target tracking	Appearance-based Statistical	Real-time	Limited robustness against lighting and background variations

2.2 Appearance-Based Approaches

From the perspective of the features used to represent the hand, vision based hand tracking and gesture recognition algorithms can be grouped into two categories: appearance-based approaches and 3D hand model-based approaches [17]. Appearance-based approaches use 2D image features to model the visual appearance of the hand and compare these parameters with the extracted image features from the input image. 3D Hand model-based approaches rely on a 3D kinematic hand model with considerable DOF and try to estimate the hand parameters by comparing the input image with the 2D appearances projected by the 3D hand model.

Appearance-based approaches are based on direct registration of hand gestures with 2D image features. The popular image features used to detect human hands and recognize gestures include hand colors [32, 33, 34, 35, 36, 37, 38, 39] and shapes [40, 41, 42, 43], local hand features [44, 45, 46, 47], optical flow [48, 49] and so on.

2.2.1 Colors and Shapes

Skin color is an important image feature to localize and track human hands. However, color-based algorithms face the difficult task of distinguishing objects which have the similar color with the hand such as human arm and face. In order to solve this problem, users are often required to wear long-sleeve shirts and restrictions are imposed on the colors of other objects in the observed scene. Color-based algorithms are also very sensitive to lighting variations. When the lighting does not meet the special requirements, color-based algorithms usually fail. To solve the problem of color distributions change under different lighting conditions, Wu *et al.* proposed the algorithm of nonstationary color-based target tracking by studying two different representations for color distributions [32]. The structure adaptive self-organizing map (SASOM) neural network is used as a new color model in this algorithm. Their experiments show that such a representation is powerful for efficient image segmentation. They formulate the nonstationary color tracking problem as a model transduction problem, the solution of which offers a way to adapt and transduce color classifiers in nonstationary color distributions. Their experiments for the task of real-time face/hand localization show that the algorithm can successfully handle some difficulties in nonstationary color tracking.

Manresa *et al.* implemented a real-time system that aims for the control of a video game based on hand gesture recognition [33]. Their system is based on three main steps: hand segmentation, hand tracking and gesture recognition. As illustrated in

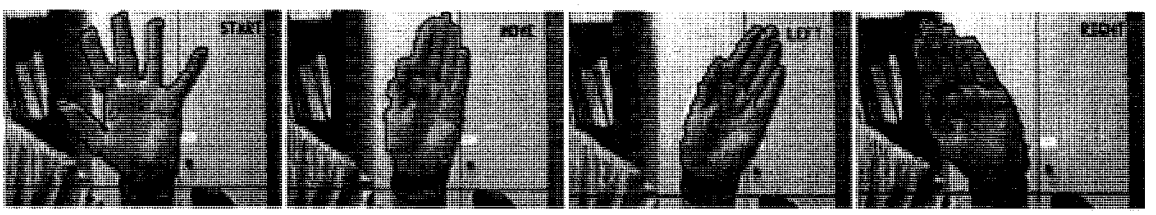


Figure 2.3: Hand tracking using the color cue (from [33]).

Figure 2.3, the color cue is used for the hand segmentation step due to its computational simplicity. To prevent errors from hand segmentation, they add a second step: hand tracking. Tracking is performed assuming a constant velocity model and using a pixel labeling approach. Several hand features are extracted and fed to a finite state classifier to identify the hand configuration. The hand can be classified into one of the four gesture classes or one of the four movement directions.

For shape-based algorithms, global shape descriptors such as Zernike moments and Fourier descriptors are used to represent different hand shapes [43]. Most shape descriptors are pixel-based and the computation cost is usually too high to implement real-time systems [50]. Another disadvantage for shape-based approaches is the requirement for noise-free image segmentation, which is a difficult task for the usually cluttered background images. Bowden *et al.* proposed an approach for sign language recognition that provides high classification rates on minimal training data [30]. Key to this approach is a 2 stage classification procedure where an initial classification stage extracts a high level description of hand shape and motion. This high level description is based upon sign linguistics and actions description at a conceptual level easily understood by humans. The second stage of classification is to model the temporal transitions of individual signs using a bank of Markov chains combined with Independent Component Analysis. The classification rates of their approach can reach 97.67% for a lexicon of 43 words using only single in-stance training which outperforms previous approaches where thousands of training examples are required.

2.2.2 Hand Features

Hand feature-based algorithms extract certain local image features such as fingertips or hand edges, and use some heuristics to find configurations or combinations of these features specific to an individual hand gesture. Oka *et al.* developed an augmented desk

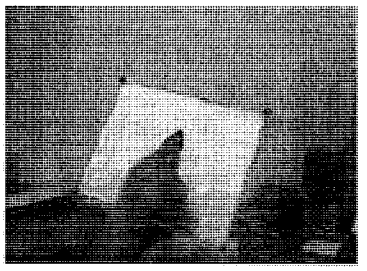


Figure 2.4: The "Visual Panel" system (from [45]).

interface depending on accurate, real-time hand and fingertip tracking for seamless integration between real objects and associated digital information [44]. They introduce a method for locating fingertip positions in image frames and measuring fingertips trajectories across image frames. By using an infrared camera, their method can track multiple fingertips reliably even on a complex background under changing lighting conditions without invasive devices or color markers. A mechanism for combining direct manipulation and symbolic gestures based on multiple fingertip motions was proposed.

Zhang *et al.* presented a vision-based interface system named "Visual Panel" (see Figure 2.4), which employs an arbitrary quadrangle-shaped panel (*e.g.* an ordinary piece of paper) and a fingertip pointer as an intuitive input device [45]. The system can accurately and reliably track the panel and the fingertip pointer. By detecting the clicking and dragging hand actions, the system can fulfill many tasks such as controlling a remote large display, and simulating a physical keyboard. Users can naturally use their fingers to issue commands and type text. Furthermore, by tracking the 3D position and orientation of the visual panel, the system can also provide 3D information, serving as a virtual joystick to control 3D virtual objects.

Malik *et al.* designed a plane-based augmented reality system that tracks planar patterns in real-time, onto which virtual 2D and 3D objects can be augmented. Interaction with the virtual objects is possible via a fingertip-based gesture recognition system [46]. As illustrated in Figure 2.5, the basis of the mechanism to capture the gesture is the number of detected fingertips. A single detected fingertip represents the gesture of pointing, whereas multiple detected fingertips represent the gesture of selecting. The fingers of the hand are detected by background subtraction and scanning the binary image for pixels of full intensity. Each time such a pixel is found, a subroutine is called to perform a neighborhood flood-fill operation to collect all neighboring pixels. The orientation of the finger can be calculated using the central moments of the detected finger blob. The axis

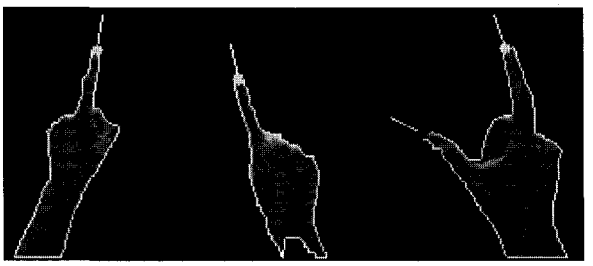


Figure 2.5: Gesture recognition based on fingertips (from [46]).

line is then defined by forcing it through the blob's centroid. The fingertip location is recovered by finding the farthest point in the blob from the root point, and is used as the pointer location.

Huang *et al.* introduced a model-based hand gesture recognition system, which consists of three phases: feature extraction, training, and recognition [47]. In the feature extraction phase, a hybrid technique combines hand edges and hand motions information of each frame to extract the feature images. Then, in the training phase, they use the principal component analysis (PCA) to characterize spatial shape variations and the hidden Markov models (HMM) to describe the temporal shape variations. Finally, in recognition phase, with the pre-trained PCA models and HMM, the observation patterns can be generated from the input sequences, and then apply the Viterbi algorithm to identify the gesture.

For feature-based based approaches, a clean image segmentation is generally a must step to recover the hand features. This is not a trivial task when the background is cluttered. On the other hand, for the highly articulated human hand, it is sometimes difficult to find local hand features and heuristics that can handle the large variety of hand gestures. It is not always clear about how to correlate local hand features with different hand gestures in an efficient manner.

2.2.3 Optical Flow

Optical flow reflects the image changes due to motion during a time interval [51]. Optical flow is represented by velocity vectors attached to the moving pixels in the image. In order to compute the optical flow, the image sequence is modeled by an intensity function $I(x, y, t)$, which varies continuously with the position (x, y) and the time t. By expanding

the intensity function in a Taylor series and ignoring the higher order terms, we have:

$$I(x+dx, y+dy, t+dt) = I(x,y,t) + \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt$$

If the intensity value at $(x+dx, y+dy, t+dt)$ is a translation of the intensity value at (x,y,t), then $I(x+dx, y+dy, t+dt) = I(x,y,t)$, so it must follow:

$$\frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt = 0$$

which equals:

$$-\frac{\partial I}{\partial t} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} = \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

$\partial I/\partial t$ at a given pixel is just how fast the intensity is changing with time. $\partial I/\partial x$ and $\partial I/\partial y$ are the spatial rates of the intensity change, *i.e.* how rapidly the intensity changes across the picture. All three of these quantities can be computed for each pixel from $f(x,y,t)$. The goal is to compute the velocity:

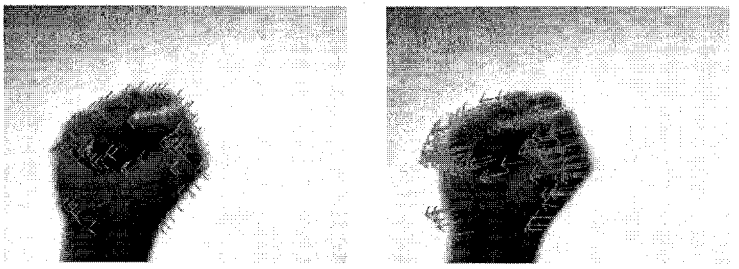
$$(u,v) = \left(\frac{dx}{dt}, \frac{dy}{dt}\right)$$

Unfortunately the above constraint gives us only one equation for two unknowns, which is not enough to get the answer. Thus an additional constraint is required to determine the value of (u,v). One popular constraint is the smoothness constraint proposed by Horn *et al.* in [51]. According to this constraint, when objects of finite size undergo rigid motion or deformation, neighboring points on the objects have similar velocities and the velocity field of the brightness patterns in the image varies smoothly almost everywhere. One way to express this additional constraint is to minimize the square of the magnitude of the gradient of the optical flow velocity:

$$\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2, \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2$$

Taking advantage of this constraint, the two unknowns (u,v) can be computed by an iteration process based on minimizing the square of the magnitude.

Figure 2.6 shows two connected frames of the hand motion analysis using the optical flow algorithm. We can easily tell the hand is moving left according to the directions of the optical flow. Cutler *et al.* presented a body gesture recognition system using optical flow in [48]. To recognize different body gestures, optical flow is estimated and



(a) Frame 1 (b) Frame 2

Figure 2.6: Hand motion analysis using the optical flow algorithm.

segmented into motion blobs. Gestures are recognized using a rule-based technique with characteristics of the motion blobs such as relative motion and size of the arm.

Yuan *et al.* described a 2D hand tracking method that extracts trajectories of unambiguous hand locations [52]. Candidate hand bounding squares are detected using a novel feature based on motion residue. This feature is combined with skin detection in color video. A temporal filter employs the Viterbi algorithm to identify consistent hand trajectories. An additional consistency check is added to the Viterbi algorithm, to increase the likelihood that each extracted trajectory will contain hand locations corresponding to the same hand. Their experiment on video sequences of several hundred frames demonstrates the systems ability to track hands robustly (see Figure 2.7).

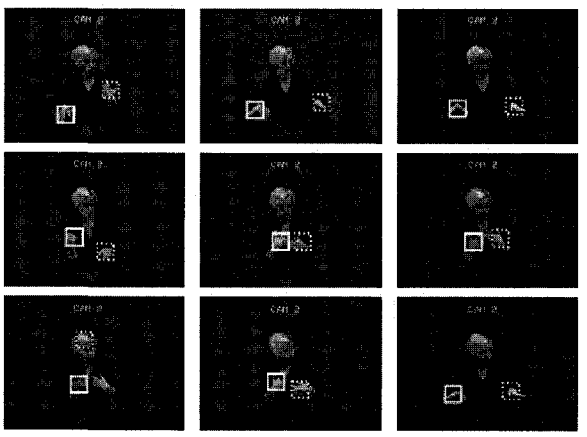


Figure 2.7: Hand tracking using motion residue and hand color (from [52]).

2.2.4 Mean Shift

Comaniciu *et al.* proposed a method for real-time tracking of non-rigid objects based on mean shift iterations and found the most probable target position in the current frame [53]. As illustrated in Figure 2.8, the mean shift algorithm is a simple iterative procedure that shifts the center of the region of interest to the center of mass (*i.e.* average of the data points). The data could be visual features such as colors or textures. The statistical distributions characterize the object of interest. Bradski proposed a modified mean shift algorithm named Continuously Adaptive Mean Shift (CamShift), which primarily intended to perform efficient head and face tracking in a perceptual user interface [54]. The CamShift algorithm finds the center and the size of the object on a color probability image frame. The probability is created via a histogram model of a specific color. The tracker moves and resizes the search window until its center converges with the center of mass. Compared with regular mean shift that uses static distributions (*i.e.* the distributions are not updated unless the target has significant changes in shape, size or color), CamShift can effectively track dynamically changing probability distributions in a visual scene. Figure 2.9 shows some results for hand tracking using this algorithm. The CamShift algorithm depends on image features such as colored blobs for fast tracking. However, it is a difficult task for the algorithm to effectively classify different hand postures that might have similar center of mass.

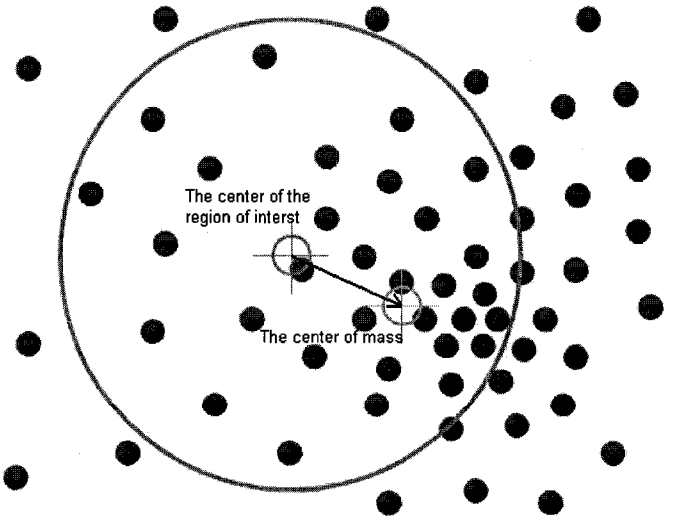
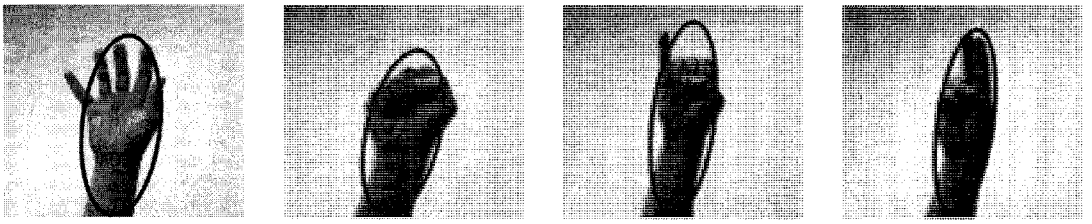


Figure 2.8: The mean shift algorithm.



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

Figure 2.9: Hand tracking use the CamShift algorithm.

2.2.5 SIFT Features

Lowe proposed an algorithm named Scale Invariant Feature Transform (SIFT) to extract distinctive invariant features from images that can be used to perform reliable matching between different views of an object or a scene (see Figure 2.10) [55]. The SIFT features use scale-space extrema in the difference-of-Gaussian function convolved with the image as the keypoints which are invariant to image scaling. By assigning a consistent orientation to each keypoint based on local image properties, the keypoint descriptor can be represented relative to this orientation and therefore achieve invariance to image rotation (see Figure 2.11). Besides the invariance to image scaling and rotation, the SIFT features are also partially invariant to changes in illumination and 3D camera viewpoint. Lowe also described an approach for object recognition using these features. The recognition proceeds by matching individual features to a database of features from known objects using a fast nearest-neighbor algorithm, followed by a Hough transform to identify clusters belonging to a single object, and finally performing verification through least-squares solution for consistent pose parameters. This recognition approach can robustly identify objects with cluttered backgrounds and occlusion, and the speed of this approach is close to real-time performance [55].

Because there are little texture and distinct corner points inside the hand region itself, especially when the resolution is low, the general object recognition framework with the SIFT features is not directly applicable to hand gesture recognition as the peaks in the difference-of-Gaussian pyramid are not stable. Zhou *et al.* found that the sub-windows around the boundary between the hand region and the back-ground region have a rich and distinctive edge gradient, which can be considered as pseudo-texture of the hand [56]. By clustering the local orientation histogram feature vectors extracted from all sub-windows, they can find the representative mean features that characterizes a particular hand shape. Their implementation takes 0.2 second to process and recognize one image when running

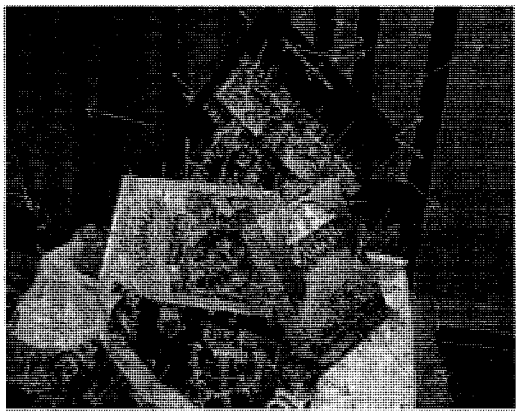


Figure 2.10: The SIFT features proposed by Lowe.

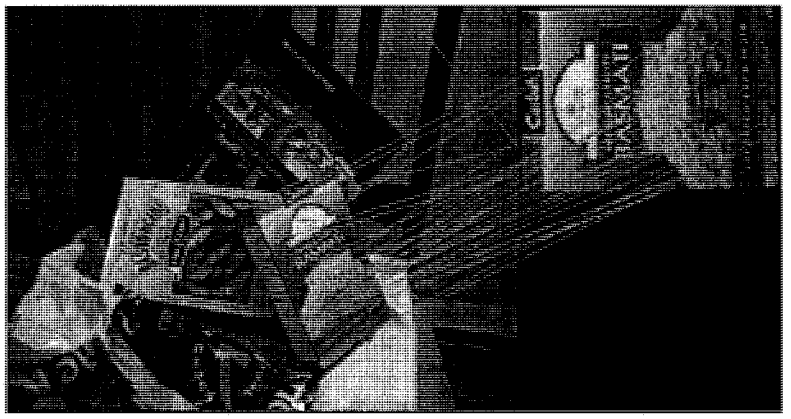


Figure 2.11: The robustness of the SIFT features against image rotation.

on a 1.3G Intel Pentium laptop processor with 512MB memory [56].

2.2.6 Stereo Image

Ye *et al.* presented a 3D gesture recognition scheme that combines the 3D appearance of the hand and the motion dynamics of the gesture to classify manipulative and controlling gestures [28]. Their method does not directly track the hand. Instead, they take an object-centered approach to compute the 3D appearance using a region-based coarse stereo matching algorithm in a volume around the hand. The motion cue is captured via differentiating the appearance feature. An unsupervised learning scheme is carried out to capture the cluster structure of these feature-volumes. Then, the image sequence of a gesture is converted to a series of symbols that indicate the cluster identities of

each image pair. Forward HMMs and neural networks are used to model the dynamics of the gestures. A real-time system for gesture recognition shown in Figure 2.12 is implemented to analyze the performance with different combinations of appearance and motion features. Their experiment results show the system can achieve a recognition accuracy of 96%.

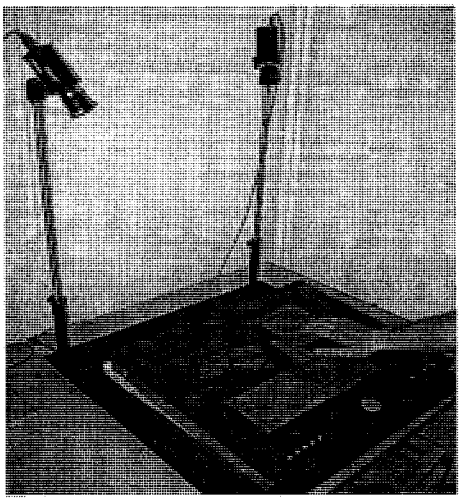


Figure 2.12: Ye's stereo gesture recognition system (from [28]).

2.2.7 The Viola-Jones Algorithm

Motivated by the task of face detection, Viola and Jones introduced a fast algorithm which minimizes the computation time while achieving high detection accuracy [57, 58]. In this algorithm, the concept of "Integral Image" is used to compute a rich set of image features. Compared with other approaches which must operate on multiple image scales, the integral image can achieve true scale invariance by eliminating the need to compute a multi-scale image pyramid, and significantly reduces the initial image processing time which is a must for most object detection algorithms. Another technique used in their approach is the feature selection algorithm based on the AdaBoost (Adaptive Boost) learning algorithm. Boosting is an aggressive feature selection technique in machine learning that can effectively improve the accuracy of a given learning algorithm. The Adaboost learning algorithm is a variation of the regular boosting algorithm, and can adaptively select the best features at each step and combine a series of weak classifiers into a strong classifier. The Viola-Jones algorithm has been primarily used for face

detection systems which is approximately 15 times faster than any previous approaches while achieving equivalent accuracy to the best published results [57]. However, limited research has been done to extend the method to hand detection and gesture recognition.

Kölsch and Turk studied view-specific hand posture detection with the Viola-Jones algorithm in [59]. They presented a frequency analysis-based method for instantaneous estimation of the "detectability" of different hand postures without the need for compute-intensive training. Their experiment results show the classification accuracy increases with a more expressive frequency-based feature type such as a closed hand palm. Kölsch and Turk also evaluated the in-plane rotational robustness of the Viola-Jones algorithm for hand detection in [60]. They found the in-plane rotation bounds for hand detection are around $\pm 15°$ for the same performance without increasing the classifier's complexity. A vision-based hand gesture interface named "HandVu" was developed by them, which can recognize standard hand postures [61]. The "HandVu" system divides hand posture recognition into three steps: hand detection, hand tracking and posture recognition [62]. To detect the hand, the user is required to put his hand palm on a predefined area of the camera scene, and the system is able to extract corresponding features that can represent the hand palm. As illustrated in Figure 2.13, the "HandVu" system employs "Flocks of Features" based on gradients and colors to facilitate 2D tracking from a monocular view [29]. After successful hand detection and tracking, the system attempts posture classification using a fanned detector based on the Viola-Jones method. The "HandVu" system uses a sequential process that the posture recognition is dependent on the previous results of hand detection and tracking. The posture recognition will fail if the hand tracking is lost, and the whole process need to be start over again from the beginning. Another limitation is the compulsory hand detection step in this system, which reduces the easiness and naturalness of the system.

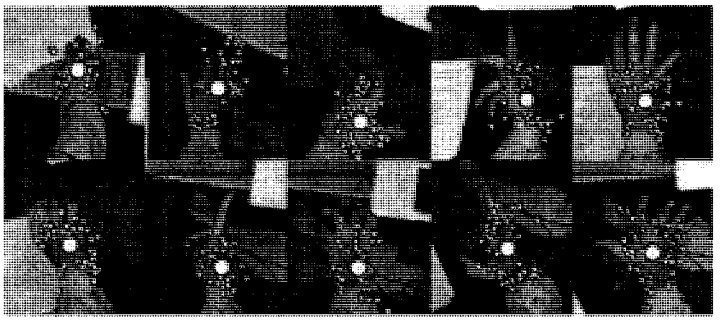


Figure 2.13: The "Flocks of Features" (from [62]).

Barczak *et al.* introduced a method to efficiently generate hand image samples for training the classifier based on the Viola-Jones algorithm [24]. The classifiers trained by the generated samples are able to track and recognize the hand posed in a single posture.

2.3 3D Hand Model-Based Approaches

Instead of using 2D image features to represent the hand directly, 3D hand model-based approaches use a 3D kinematic hand model that can be animated to represent different hand poses. An estimation-by-synthesis strategy is employed to recover the hand parameters by aligning the appearance projected by the 3D hand model with the observed images from the camera, and minimizing the discrepancy between them. This is a challenging exhaustive search problem in a high dimensional space as the human hand is a complex articulated object with 27 DOF. To describe a single hand posture, all of the key images from different viewpoints need to be registered. As illustrated in Figure 2.14, 3D hand model-based approaches search a region in the high dimensional space by three steps: first, the algorithm proposes hypothesis parameters based on prior knowledge like previously recovered hand configuration and hand dynamics; then the algorithm uses the hypothesis parameters to animate the 3D hand model and project the model to a 2D image and compare it with the input image; the algorithm keeps on varying the hypothesis parameters until the projected image match the observation image and successfully classify the hand gesture [56].

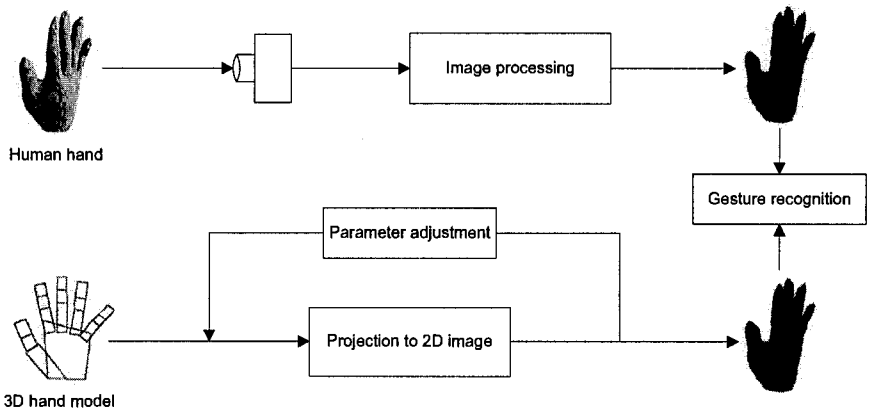


Figure 2.14: The block diagram of 3D hand model-based approaches.

2.3.1 Analysis-by-Synthesis

Tomasi *et al.* implemented an analysis-by-synthesis 3D hand tracking system with a detailed 3D hand model which can reflect the shape and articulations of a hand [31]. The 3D hand model can be animated by specifying pose parameters and joint angles. A database of known hand poses and configurations is set up in advance. With each configuration (and therefore for a whole set of views and samples for the same configuration), the database store the set of joint angles and pose parameters that describe that configuration, or at least a similar one, obtained manually through the same graphics package used for rendering. During tracking, they compare input video frames to these samples, and whenever they see a "familiar" view, they retrieve the corespondent hand configuration. In their experiments, a database with 15 views for each of 24 hand signs is used. Vector quantization principal component analysis is employed to reduce the dimensionality of the feature space. Because hand motions often occur too quickly in the video, and the hand configurations are often too complex to track from each single frame, 3D configurations interpolation between familiar views is used. This system requires single-user and restricted lighting and background as well as fairly disciplined way to sign the gesture.

To alleviate the computation load of searching a high dimensional space, Wu *et al.* proposed an approach by decoupling hand poses and finger articulations and integrating them in an iterative framework [63]. They treat the palm as a rigid planar object and use a 3D cardboard hand model to determine the hand pose based on the Iterative Closed Point (ICP) algorithm. Since the finger articulation is also highly constrained, they proposed an articulation prior model that reduces the dimensionality of the joint angle space and characterizes the articulation manifold in the lower-dimensional configuration space. To effectively incorporate the articulation prior into the tracking process, they proposed a sequential Monte Carlo tracking algorithm by using the important sampling technique. In their implementation, the hand gestures are performed in front of a clean black background, and the proposed cardboard hand model can not handle large out-of-plane rotations and scaling invariance very well. In addition, the system requires an user-specific calibration of the hand model that is manually done.

2.3.2 Image Retrieval

Due to the flexibility and high DOF of the 3D hand model, it is desirable to use an image database to store the large number of images projected by the hand model. The problem

of hand pose estimation can be converted to an image retrieval problem by finding the best match between the input image and the database. Zhou *et al.* proposed an approach to integrate the powerful text retrieval tools with computer vision techniques to improve the efficiency for hand images retrieval [25]. An Okapi-Chamfer matching algorithm is used in their work based on the inverted index technique. With the inverted index, a training image is treated as a document, and a test image is treated as a query. In the matching process, only documents that contain query terms are accessed and used so that the query image can be identified at constant computational cost. This approach can accelerate the database matching and improve the efficiency of image retrieval. To enable inverted indexing in an image database, they built a lexicon of local visual features by clustering the features extracted from the training images. Given a query image, they extract visual features and quantize them based on the lexicon, and then look up the inverted index to identify the subset of training images with non-zero matching score. The Okapi weighting formula matches a query with a document based on a weighted sum of the terms that appears in both the document and the query. To use the Okapi weighting formula for image matching, they proposed to combine the Okapi weighting formula with the Chamfer distance and assign a spacial tag to each local feature to record its relative image position, and use this tag to calculate the Chamfer distance. The local features they used are the binary patches along the boundary between foreground and background. The geometry of the boundary is modeled by the spacial tags. The trade off of their approach is that labeling real-world images is time consuming and error prone, and their tests on real-world query images are not very extensive.

Athitsos *et al.* proposed a method that can generate a ranked list of three-dimensional hand configurations that best match an input image in [64]. Hand pose estimation is

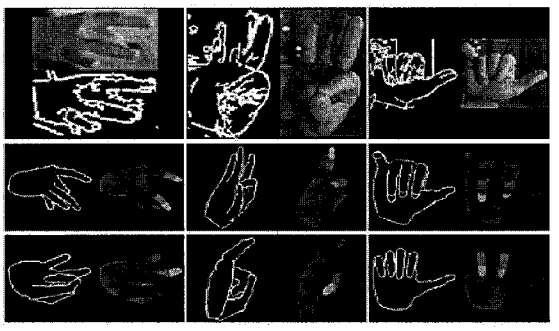
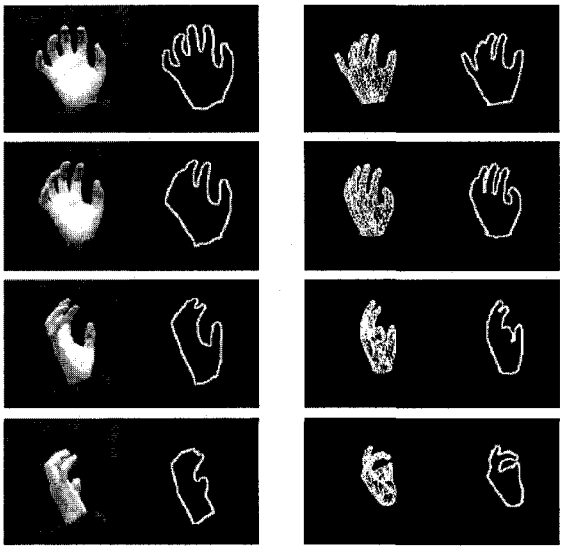


Figure 2.15: Clutter-tolerant image retrieval experiment results (from [64]).

formulated as an image database indexing problem, where the closest matches for an input hand image are retrieved from a large database of synthetic hand images. The novelty of their system is the ability to handle the presence of clutter by using two clutter-tolerant indexing methods. First, a computationally efficient approximation of the image-to-model chamfer distance is obtained by embedding binary edge images into a high-dimensional Euclidean space. Second, a general-purpose, probabilistic line matching method identifies those line segment correspondences between model and input images that are the least likely to have occurred by chance. The performance of this clutter tolerant approach is demonstrated in experiments with hundreds of real hand images as Figure 2.15 shows. The total processing time, including hand segmentation, extraction of line segments, and the two retrieval steps, was about 15 seconds per input image on a PC with a 1.2GHz Athlon processor.

To overcome the high computation cost of the large number of appearance variations due to the high DOF of the 3D hand model and different viewpoints, Imai *et al.* proposed a 2D appearance-based method by using hand contours to estimate 3D hand posture in [65]. In their method, the variations of possible hand contours around the registered typical appearances are trained from a number of computer graphic images generated from a 3D hand model. The possible variations are efficiently represented as



(a) Input images (b) Retrieved images

Figure 2.16: 3D posture estimation by matching hand contours (from [65]).

the Locally-Compressed Feature Manifold (LCFM) in an appearance feature space. The posture estimation for the sequential images is done by tracking the posture in the LCFM. They experimented 7 image sequences which contain 700 real human hand images (*i.e.* 100 images for each sequence), 81.3% images are correctly matched by searching 80,000 posture classes and their contours. Figure 2.16 shows a fraction of their experimental results. The computation time is 92.5ms per match in the case of using the online hand posture estimation system.

2.4 Statistical Approaches

From the point of view of the methodology used to describe hand gestures, vision-based hand gesture recognition algorithms can be grouped into statistical approaches and syntactic approaches as Table 2.2 shows [66].

Table 2.2: Statistical and syntactic approaches.

Approach	**Gesture Representation**	**Decision Rules**
Statistical	Raw data, feature vector, *etc.*	Direct matching, minimum distance, nearest neighbor, maximum likelihood, Bayes rule, *etc.*
Syntactic	String, tree, graph, *etc.*	Parsing, string matching, tree matching, graph matching, *etc.*

For statistical approaches, hand gestures are represented by a number of features, and each hand gesture is viewed as a point in the feature space. The goal is to select appropriate features that allow hand gestures belonging to different regions in the feature space so that the classifier can identify them correctly. For syntactic approaches, hand gestures are decomposed into a set of simpler primitives, and each hand gesture is represented by the interrelationships of these primitives using a string, a tree or a graph. The classification is achieved by parsing the corresponding representation according to a given grammar.

As illustrated in Figure 2.17, hand gesture recognition systems based on statistical approaches can be decomposed into two components: the training component and the recognition component. In the training component, the feature selection module finds appropriate features to represent the training samples for hand gestures and a classifier is obtained by the learning module based on the training samples. The feedback path

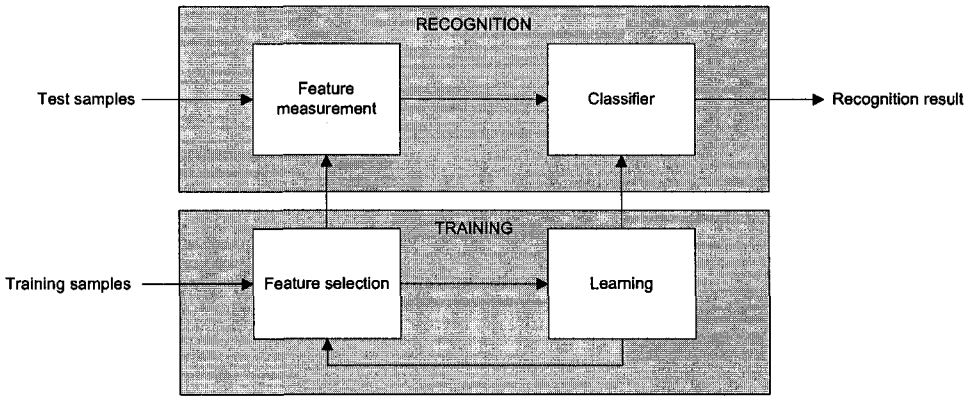


Figure 2.17: The block diagram of statistical approaches.

between the feature selection module and the learning module allows the user to optimize the training process by adjusting the training strategies and parameters. In the recognition component, the feature measurement module extracts the features from the test sample and the trained classifier recognizes them based on the measured features and the corresponding decision rules.

There are four different methods to design a classifier [67]:

- *Similarity measurement:* this is the most intuitive and simplest method. With this approach, patterns that are similar should belong to the same class. Template matching and minimum distance classifiers are typical methods of this category.

- *Probabilistic classifiers:* the most popular probabilistic classifier is the Bayes classifier that uses the Bayes rule to estimate the conditional probability of a class.

- *Geometric approach:* the classifier of this category constructs decision boundaries by optimizing certain error criterion. The classic example of this type of classifier is Fisher's linear discriminant, which can project high-dimensional data onto a line and performs classification in this one-dimensional space. The projection maximizes the distance between the means of the two classes while minimizing the variance within each class. One example of the application of Fisher's linear discriminant is human face recognition [68]. Human faces include a large number of image features. Fisher's linear discriminant can reduce the large number of features to a more manageable number for easier and more accurate classification. Each of the new dimensions is a linear combination of pixel values, which form a new

2.5 Syntactic Approaches

In many computer vision problems involving complex patterns and activities, statistic approaches and numeric measurements might not be enough to represent the complex structures of these patterns and activities. Under this situation, it is more appropriate and effective to use a syntactic approach to describe these patterns and activities with their simpler sub-patterns and elementary parts [70].

The elementary parts used to syntactically describe a complex pattern or an activity are called *primitives*. For computer vision applications, the principles to identify the primitives are [71]:

- The number of primitive types should be small.
- The primitives selected must be able to form an appropriate object representation.
- Primitives should be easily segmentable from the image.
- Primitives should be easily recognizable using some statistical pattern recognition method.
- Primitives should correspond with significant natural elements of the object structure being described.

After the primitives are extracted, a *grammar* representing a set of rules must be defined so that different patterns and activities can be constructed based on the extracted primitives. This syntactic approach can be better explained by analogy with the English language. A mathematical model for the grammar's structure can be defined as:

$$G = [V_t, V_n, P, S]$$

In this model:

- V_t is a set of *terminals*, which are the most primitive symbols in the grammar.
- V_n is a set of *non-terminals*, which are symbols composed by a collection of terminals and/or other non-terminals. V_n and V_t are disjoint alphabets, *i.e.* $V_n \cap V_t = \emptyset$.
- P is a finite set of *production rules*, which is a transformation from one sequence of terminals and/or non-terminals to another sequence of terminals and non-terminals.

template.

- *Decision tree classifiers:* this type of classifiers can deduce the conclusion using an iterative multistage decisions based on individual features at each node of the decision tree. The basic idea is to break up a complex decision problem into a series of several simpler decisions, so that the final conclusion would resemble the desired solution. The most popular decision tree classifiers are binary decision tree classifiers, which make true or false decisions at each stage based on the single corresponding feature at the node.

To improve the the overall classification accuracy, different classifiers can be combined so that the overall performance can be optimized. A classifier combination can achieve better performance especially when the individual classifiers are largely independent [67]. A typical example is the boosting algorithm, which combines a series of weak classifiers (whose accuracies are only slightly better than 50%) into a strong classifier which has a very small error rate on the training data [69]. In the boosting algorithm, individual classifiers are invoked in a linear sequence. The inaccurate but cheap classifiers (low computational cost) are applied first, followed by more accurate and expensive classifiers. The number of mistakenly classified samples is reduced gradually as more individual classifiers have been invoked and added to the sequence. The final strong classifier (whose accuracy meets the requirement) is a linear combination of the invoked individual classifiers.

Besides the discussed statistical models, there is a special model need to be introduced: the neural networks, which have become an important tool in computer vision. Neural networks are parallel computing systems consisting of a large number of interconnected neurons (elementary processors), which mimics the complex structure of neurons in human brains. Neural networks are able to learn complex nonlinear input-output relationships using sequential training procedures and adapt themselves to the data. In spite of the seemingly different mechanisms, there is considerable overlap between neural networks and statistical models in the field of pattern recognition. Most of the well known neural network models are implicitly equivalent or similar to classical statistical approaches [67]. Neural networks implement pattern recognition as black boxes by concealing the complex statistics from the user.

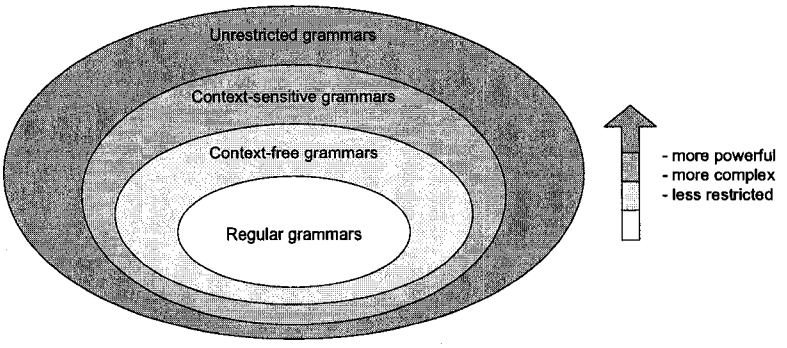


Figure 2.18: The Chomsky hierarchy of grammars.

- S is a *start symbol*, which shows where all valid sequences of symbols we want to be able to produce can be derived from.

According to the Chomsky hierarchy illustrated in Figure 2.18, grammars can be divided into four types ordered from general (*i.e.* unrestricted grammars) to specific (*i.e.* regular grammars) [72]:

Type 0: Unrestricted Grammars
Unrestricted grammars include all grammars. There is no limitation for the substitution rules. It can have any strings on either the left side or the right side of the substitution arrow.

Type 1: Context-Sensitive Grammars
The substitution rule of context-sensitive grammars is:

$$\alpha A \beta \rightarrow \alpha \gamma \beta$$

where A is a nonterminal, α, β and γ are strings of terminals and nonterminals. α and β can be empty strings but γ must be nonempty. This substitution rule can be understood as "A can be replaced by γ in the context of α, β".

Type 2: Context-Free Grammars
The substitution rule of context-free grammars (CFG) is:

$$A \rightarrow \gamma$$

where A is a nonterminal, γ is a nonempty string of terminals and nonterminals. This substitution rule limits the left side consisting of only a single nonterminal.

The context-free grammar allows the nonterminal A to be replaced by the string γ independently of the context where A appears. One example of context free grammars is: $G = (V_N, V_T, P, S)$, where $V_N = \{S, A, B\}$, $V_T = \{a, b\}$, and P:

$$\begin{array}{ll} (1) S \rightarrow aB & (2) S \rightarrow aB \\ (3) A \rightarrow aS & (4) A \rightarrow bAA \\ (5) A \rightarrow a & (6) B \rightarrow bS \\ (7) B \rightarrow aBB & (8) B \rightarrow b \end{array}$$

This grammar defines a language $L(G)$ which is the set of strings consisting of an equal number of a's and b's such as ab, ba, $abba$ and $bbaa$.

Type 3: Regular Grammars

The substitution rule of regular grammars is:

$$A \rightarrow \alpha B \text{ or } A \rightarrow \beta$$

where A and B are nonterminals, α and β are terminals or empty strings. Besides limiting the left side consisting of only a single nonterminal, this substitution rule also restricts the right side: it may be an empty string, or a single terminal symbol, or a single terminal symbol followed by a nonterminal symbol, but nothing else. The languages generated by regular grammars are called regular or finite state languages. One example is: $G = (V_N, V_T, P, S)$, where $V_N = \{S, A\}$, $V_T = \{a, b\}$, and P:

$$\begin{array}{l} S \rightarrow aA \\ A \rightarrow aA \\ A \rightarrow b \end{array}$$

This grammar defines a language $L(G) = \{a^n b | n = 1, 2, \ldots\}$, which includes strings such as ab, aab, $aaaab$.

The Chomsky hierarchy is inclusive: every regular grammar is context-free, every context-free grammar is context-sensitive and every context-sensitive grammar belongs to unrestricted grammars. Although unrestricted grammars are the most inclusive grammars, they are too general to be useful, and it cannot be decided whether a specific string is generated by an unrestricted grammar. Context-free grammars and regular grammars are much less powerful than unrestricted grammars, however, these two restricted grammars are most often used because parsers for them can be efficiently implemented [73].

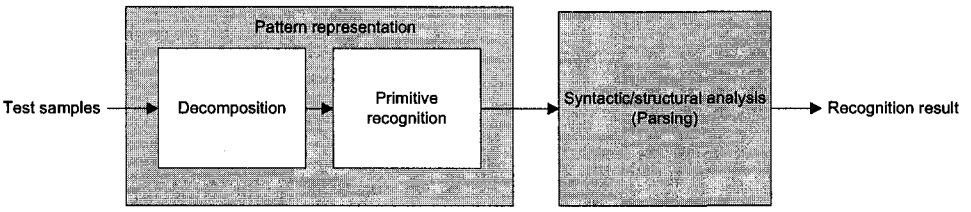


Figure 2.19: The block diagram of syntactic approaches.

All regular languages can be recognized by a finite state machine, and for context-free grammars, there are many efficient parsers to recognize the corresponding languages.

After appropriate primitives are selected and the grammar is defined, a pattern or an activity can be analyzed according to the block diagram shown in Figure 2.19. There are two major components included in this diagram: the pattern representation component and the structural/syntactic analysis (parsing) component. The pattern representation component includes the decomposition module and the primitive recognition module. After all of the primitives are extracted and recognized, the structural/syntactic analysis component will analyze the relationship of the extracted primitives according to the defined grammar. Different from statistical approaches, the training component is not required for syntactic approaches.

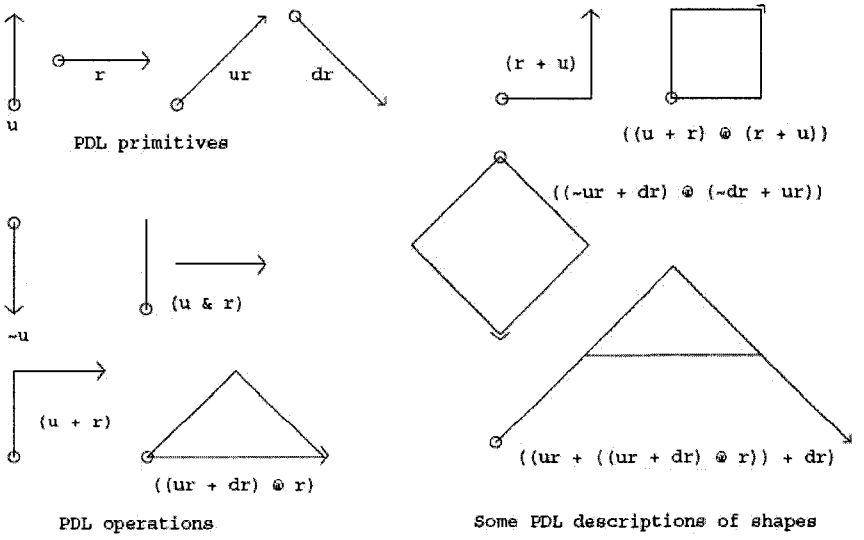


Figure 2.20: Shaw's picture description language (from [74]).

One classic example for the application of syntactic approaches in computer vision is the Picture Description Language (PDL) proposed by Shaw in [74]. As illustrated in Figure 2.20, Shaw's PDL accepts the description of line drawings in terms of a set of primitives, operations and a grammar generating strings. The grammar is used to direct the analysis or parse, and to control the calls on pattern classification routines for primitive picture components. The benefits of Shaw's PDL include ease of implementation and modification of picture processing systems, and simplification of the pattern recognition problem by automatically taking advantage of contextual information.

Hand *et al.* believe it is feasible to employ a syntactic approach to understand hand gestures in [75]. To do this, they defined a set of hand postures as the terminals of the language. These terminals included hand postures like *HO* – hand open, *HF* – hand fist, *IF* – index finger outstretched. Besides these hand posture terminals, they added several hand movement terminals such as: *MU*, *MD* – move up and down; *ML*, *MR* – move left and right; and *MT*, *MA* – move towards and away. After all of the terminals are decided, a set of production rules is defined:

<Gesture> :== <Pick>|<Point>|<Translate>|<Undo>|<Redo>|<Stop>

<Pick> :== <ChooseLocation><Grab><Drag><Drop>

<ChooseLocation> :== TIF-Motion <ChooseLocation>

<Grab> :== TIFC

<Drag> :== TIFC-Motion <Drag>

<Drop> :== TIF

<Point> :== <MovePoint> <SelectPoint>

<MovePoint> :== IF-Motion <MovePoint>

<SelectPoint> :== TIF

<Translate> :== FTFO <Direction>

<Direction> :== ML|MR|MU|MD|MT|MA

<Undo> :== IMFO MA

<Redo> :== IMFO MT

<Stop> :== HF HO

According to the production rules, if the user wants to perform a "stop" command, he would make a fist and then release it. To drag an object, the user need to hold the thumb and index finger closed and continue to move it.

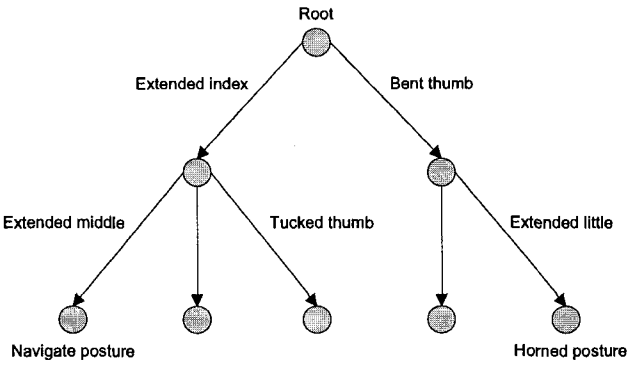


Figure 2.21: A partial tree for hand postural features (from [76]).

Jones *et al.* suggest a method to sequentially parsing the structure for hand postures given a set of observed values [76]. The approach is based on constructing and parsing a feature tree like the one shown in Figure 2.21. Key features are represented by branches such as "extended index", and as the observed values are examined, a tree parsing takes place. For example, if the user's index finger is extended, the parsing first takes the left branch, and all of the postures with this feature would be represented under this branch.

Derpanis *et al.* proposed an approach to exploit previous linguistic theory to represent complex gestures in terms of their primitive components [26]. In their approach, dynamic gestures are decomposed into static and dynamic components in terms of three sets of primitives: hand shape, location and movement. An algorithm is proposed, which can recognize gesture movement primitives given data captured with a single video camera. By working with a finite set of primitives, which can be combined in a wide variety of ways, their approach has the potential to deal with a large vocabulary of gestures. They demonstrated that given a monocular gesture sequence, kinematic features can be recovered from the apparent motion that provide distinctive signatures for 14 primitive movements of ASL.

Another important application of syntactic approaches is for activity recognition. Ivanov *et al.* used the stochastic context-free grammar parsing to recognize activities taking place over extended sequences such as car parking and structured gestures composed of simple hand trajectories [77]. The author first split activities extended over time into events using probabilistic event detectors. The sequence of events was then sent to a stochastic context-free grammar parsing mechanism for recognition. The grammar parsing mechanism provides longer range temporal constraints, disambiguates uncertain

of the events, and allows the inclusion of a priori knowledge about the structure of temporal events in a given domain. They demonstrated how the system correctly interprets activities of single and multiple interacting objects in experiments on gesture recognition and video surveillance.

Ryoo *et al.* proposed a general methodology for automated recognition of complex human activities [78]. They use a context-free grammar based representation scheme to represent composite actions and interactions. The context-free grammar describes complex human activities based on simple actions or movements. Human activities are classified into three categories: atomic action, composite action, and interaction. The system was tested to represent and recognize eight types of interactions: approach, depart, point, shake-hands, hug, punch, kick, and push. The experiments show that the system can recognize sequences of represented composite actions and interactions with a high recognition rate.

Moore *et al.* presented a model of stochastic context-free grammar for characterizing complex, multi-tasked activities that require both exemplars and models [79]. Exemplars are used to represent object context, image features, and motion appearances to label domain-specific events. Then, by representing each event with a unique symbol, a sequence of interactions can be described as an ordered symbolic string. The stochastic context-free grammar, which is developed using underlying rules of an activity, provides the structure for recognizing semantically meaningful behavior over extended periods. Symbolic strings are parsed using the Earley-Stolcke algorithm to determine the most likely semantic derivation for recognition. Parsing substrings allows the system to recognize patterns that describe high-level, complex events taking place over segments of the video sequence. The performance of the system is shown through experiments with a popular card game by identifying player strategies and behavior extracted from real-time video input.

Minnen *et al.* implemented a system that uses an extended stochastic grammar to recognize a person performing the Towers of Hanoi task from a video sequence by analyzing object interaction events [80]. In this system, they extend stochastic grammars by adding event parameters, state checks, and sensitivity to an internal scene model. Experimental results from several videos showed robust recognition for the full task and the constituent sub-tasks even though no appearance models are provided for the objects in the video.

Yamamoto *et al.* proposed a new approach for recognition of task-oriented actions based on a stochastic context-free grammar [81]. They use a context-free grammar to

recognize the action in Japanese tea services. A segmentation method was proposed to segment the tea service action into a string of finer actions corresponding to terminal symbols with very few errors. A stochastic context-free grammar-based parsing and a Bayesian classifier were used to analyze and identify the action with a maximum posterior probability.

2.6 Discussions

2.6.1 Appearance vs. 3D Hand Model

Generally speaking, it is easier for appearance-based approaches to achieve real-time performance due to the comparatively simpler 2D image features. 3D hand model-based approaches offer a rich description that potentially allows a wide class of hand gestures. However, as the 3D hand model is a complex articulated deformable object with many degrees of freedom, a very large image database is required to cover all the characteristic hand images under different views. Matching the query images from the video input with all hand images in the database is time-consuming and computationally expensive. Another limitation of 3D hand model-based approaches is the lack of the capability to deal with singularities that arise from ambiguous views [82]. Based on the literature review, rather than hand gesture recognition, most current 3D hand model-based approaches focus on real-time tracking for global hand motions and local finger motions with restricted lighting and background conditions. Another issue for 3D hand model-based approaches is the scalability problem, where a 3D hand model with specific kinematic parameters cannot deal with a wide variety of hand sizes from different people.

2.6.2 Statistical vs. Syntactic

When the patterns and activities under research are complex and include explicit structural information and relationships among sub-patterns and primitives, it is more effective to use a syntactic approach to decompose the complex pattern into their simpler sub-patterns and primitives, which are easier to process.

Statistical approaches and syntactic approaches are not absolute disjoint categories [83]. In many situations, statistical approaches and syntactic approaches can complement with each other, and a combination of statistical and syntactic approaches can be used to construct a more efficient hybrid system: the statistical approach is responsible for prim-

itives extraction and identification; the syntactic approach is responsible for analyzing the structural relationship among the identified primitives so that the whole pattern can be recognized.

Most current gesture recognition systems treat the hand gesture as a whole element without considering its hierarchical composite property and breaking it into its simpler constituent components that would be easier to process. This results in a rather slow and inefficient system unsuited for real-time applications. To solve the problem, the advantages brought by syntactic approaches need to be considered. An appropriate combination of statistical and syntactic approaches can result in an efficient and effective hand gesture recognition system.

2.7 Summary

This chapter introduces the concepts of hand posture, hand gesture, and how to select appropriate gesture commands. The previous research work is reviewed from two different dichotomies: appearance-based approaches versus hand model-based approaches and statistical approaches versus syntactic approaches.

For appearance-based approaches, it is generally easier to achieve real-time performance. The tradeoff is their limited ability to cover different classes of hand gestures due to simpler image features employed. 3D hand model-based approaches offer a rich description that potentially allow a wide class of hand gestures. However, the higher computation cost often reduces the system's processing speed.

For computer vision problems involving complex patterns and activities, it is more appropriate and effective to use a syntactic approach to describe each pattern or activity with their primitives. The recognition of primitives may be better accomplished by statistical approaches due to little structural information involved.

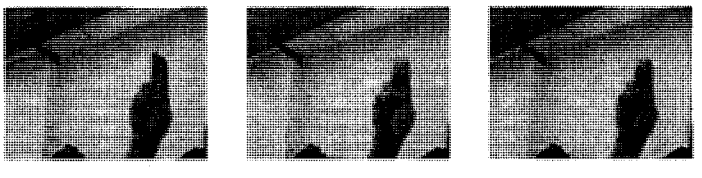
Chapter 3

A Two-Level Architecture

3.1 Selection of Postures and Gestures

A hand gesture is a composite action constructed by a series of hand postures acting as transition states. For example, Figure 3.1 shows 6 consecutive frames of a normal speed hand gesture recorded with a video camera at 30 frames per second. A hand gesture may also involve global hand motions, which can result in different semantic meanings. For example, for the hand gestures shown in Figure 3.2, according to the Chinese Sign Language, the gesture shown by Figure 3.2 (a) means "By driving", while the gesture shown by Figure 3.2 (b) means "By cycling" [84].

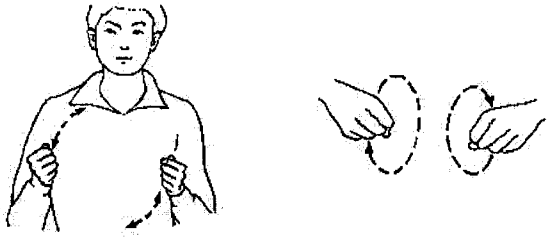
To better understand hand gestures of different classes, Quek proposed a gesture taxonomy that better fits the context of human-computer interaction [85]. In the taxonomy



(a) Frame 1 (b) Frame 2 (c) Frame 3

(d) Frame 4 (e) Frame 5 (f) Frame 6

Figure 3.1: Six consecutive frames of a hand gesture.



(a) By driving (b) By cycling

Figure 3.2: Different semantic meanings for different global hand motions (from [84]).

shown in Figure 3.3, all hand movements are divided into two categories: hand gestures and unintentional movements. Unintentional movements are hand motions that do not have any intentions to communicate information. Hand gestures can be divided into two groups: manipulative gestures and communicative gestures. Manipulative gestures are the ones used to act on objects in an environment (such as picking up a box). Communicative gestures intend to communicate information. Communicative gestures can be further divided into acts and symbols. Acts are gestures that are directly related to the interpretation of the movement itself. Symbols are the gestures that have a linguistic role and symbolize some referential actions. For the applications of human-computer interaction, communicative gestures are the most commonly used since they can often be represented by different static hand postures and movements.

In order to implement a system for hand gesture recognition, it is important to first build an architecture that can provide a framework for the task. The goal of our architec-

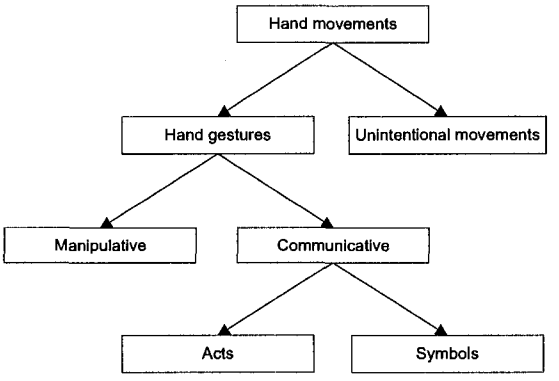


Figure 3.3: The gesture taxonomy in the context of human-computer interaction.

Table 3.1: Selected hand postures.

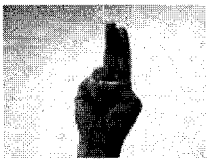
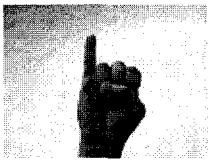
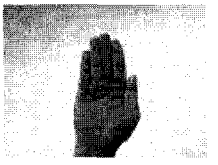
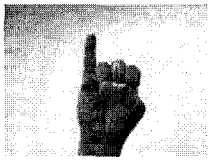
Palm	Fist	Two Fingers	Little Finger
			

Table 3.2: Hand gestures constructed by selected hand postures.

Hand Gesture	Constituent Postures	
Grasp		
Quote		
J		

ture is not to build a system capable of recognizing all manipulative and communicative hand gestures, but to demonstrate an effective methodology which can achieve accurate and robust recognition of a set of intuitive communicative hand gestures for the purpose of human-computer interaction. For communicative hand gestures, only a relatively small subset of all possible hand postures play a fundamental role [31]. The key hand postures selected for our system are shown in Table 3.1. The selection of this posture set is based on our experiment results which show no confusion is caused among these postures by the employed algorithm. This set of key hand postures is sufficient for the hand doing a set of hand gestures listed in Table 3.2. Each gesture is composed of two postures. The order of the postures can be reversed so that each gesture can be a repetitive action, which improves the comfort for the user.

3.2 System Architecture

Based on the hierarchical composite property of hand gestures, it is natural to use a divide-and-conquer strategy to break hand gestures into their constituent hand postures and movements, which can be treated as primitives for the syntactic analysis based on grammars. To implement this strategy, we propose a two-level architecture shown in Figure 3.4. This architecture decouples hand gesture recognition into two levels: the low-level hand posture detection and tracking and the high-level hand gesture recognition and motion analysis. Because there is no obvious structural information for hand postures, it is more appropriate to use a statistical approach for the low-level of the architecture. Two components are included in the statistical approach: the training component and the posture detection and tracking component. The high-level of the architecture is responsible for gesture recognition and motion analysis. A syntactic approach is selected to fully use the composite property of hand gestures. The detected hand postures and motion trajectories from the low-level are sent to the high-level of the architecture as primitives for syntactic analysis so that the whole gesture can be recognized according to the defined grammars. There are two components included in the high-level of the architecture: the local finger motion analysis component and the global hand motion analysis component.

The block diagram of the two-level architecture is shown in Figure 3.5. At the low-level of the architecture, hand gesture images are collected from the web-camera shown

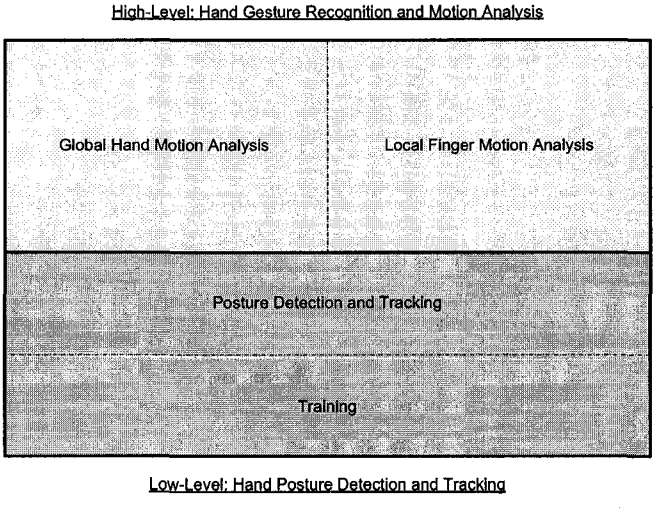


Figure 3.4: The two-level architecture for hand gesture recognition.

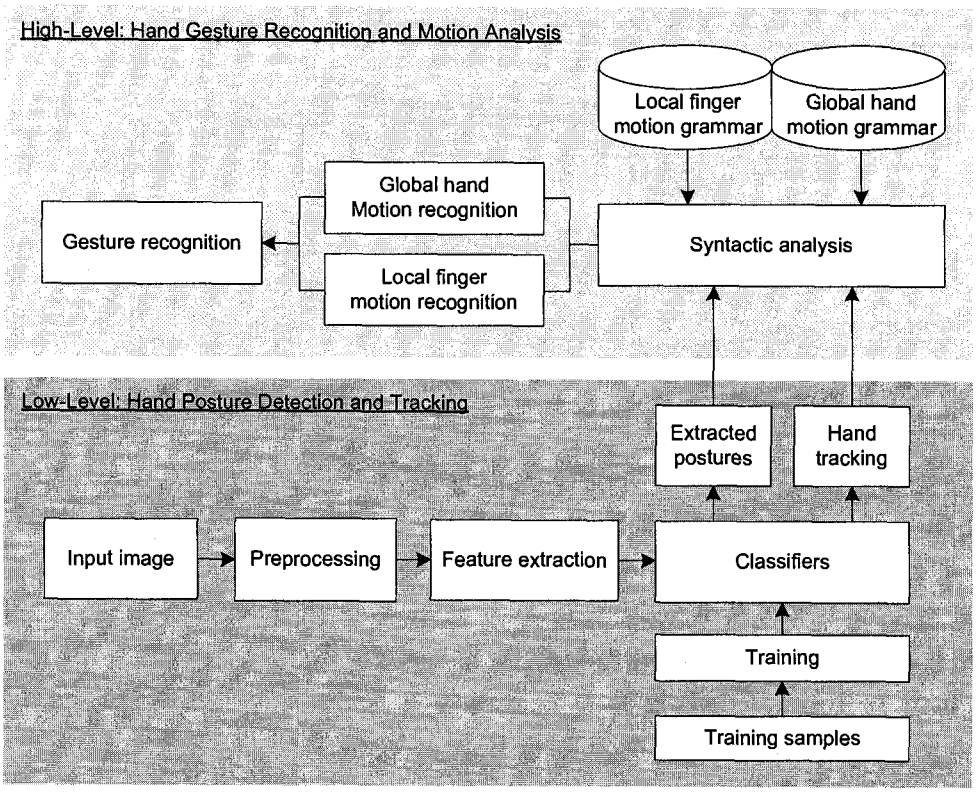


Figure 3.5: The block diagram of the two-level architecture.

in Figure 3.6. This web-camera provides video capture with maximum resolution of 640×480 up to 15 frames-per-second. For our system, we set the camera resolution at 320×240 with 15 frames per second.

After the input image is loaded, the preprocessing module will segment the hand posture from the background, remove noise, and perform any other operation which will contribute to a clean image of the hand posture. The classifiers will detect and recognize the posture so that they can be converted into terminal strings for the syntactic analysis. The hand tracking module tracks global hand motions and keep the trajectory for the high-level global hand motion analysis.

To meet the requirement of real-time performance, we train our classifiers with a statistical approach based on a set of Haar-like features, which can be computed very fast by using the "Integral Image" technique. To achieve detection accuracy and further improve the computation speed, we use a boosting algorithm which can efficiently get rid of false images and detect target images with selected Haar-like features stage by stage.

Figure 3.6: The web-camera for the video input.

With a set of positive samples (*i.e.* images contain hand postures) and negative samples (*i.e.* random images do not contain hand postures), the training module selects a series of Haar-like features that can achieve the best classification accuracy, and combine them into a final accurate classifier.

The high-level of the architecture is responsible for syntactic analysis for hand gestures. The local finger motion is analyzed according to the primitives represented by different hand postures detected from the low-level and a grammar that defines the relationship between the gestures and the primitives. The goal of global hand motion analysis module is to identify different patterns of hand motion trajectories, which can have different semantic meanings. The global hand motion is analyzed based on the primitives represented by movement directions detected by the hand tracking module at the low-level. The global hand motion grammar defines the relationship between the hand motion trajectories and the correspondent primitives.

3.3 Summary

In this chapter, we propose a hybrid two-level architecture to solve the problem of hand gesture recognition. Considering the hierarchical composite property of hand gestures, this architecture decouples hand gesture recognition into two levels: low-level hand posture detection and tracking and high-level hand gesture recognition and motion analysis. The low-level of the architecture detect hand postures using a statistical approach based on Haar-like features and a boosting algorithm. The high-level of the architecture employs a syntactic approach to recognize hand gestures and analyze hand motions based on defined grammars.

Chapter 4

Hand Posture Detection and Tracking

4.1 Haar-Like Features

Viola and Jones employed a statistical approach for the task of face detection to handle the large variety of human faces (*e.g.* faces maybe rotated in three directions, different face colors, some faces with glasses, some faces with beard *etc.*) [57, 58]. In their approach, the concept of "Integral Image" is used to compute a rich set of Haar-like features, which can significantly reduce the image processing time. The training algorithm of the Viola-Jones approach takes a set of "positive" samples, which contain the objects of interest (in our case: hand postures) and a set of "negative" samples, *i.e.* images do not contain the objects of interest [86]. During the training process, distinctive Haar-like features are selected at each stage to identify the images containing the object of interest. When the trained classifier misses an object or detects a false object, adjustments can be made easily by adding more Haar-like features so that the mistakenly classified samples can be corrected. The Viola and Jones algorithm is approximately 15 times faster than any previous approaches while achieving equivalent accuracy as the best published results [58].

The simple Haar-like features are computed similarly to the coefficients in the Haar wavelet transform. Each Haar-like feature is described by a template which includes connected black and white rectangles, their relative coordinates to the origin of the search window and the size of the feature. Figure 4.1 shows the extended Haar-like feature set proposed by Lienhart [87]. This Haar-like features set includes four edge features, two

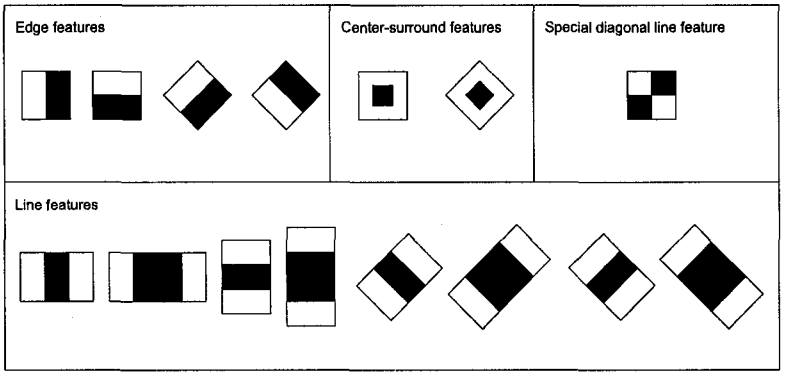


Figure 4.1: The extended set of Haar-like features.

center-surround features, one special diagonal line feature and eight line features. There are three reasons for the employment of the Haar-like features rather than raw pixels. The first reason is that the Haar-like features can encode ad-hoc domain knowledge which is difficult to learn using a finite quantity of training data. Haar-like features are effective to catch the characters represented by the difference between the dark and bright areas within an image. One typical example is that the eye region of the human face is darker than the cheek region, and one Haar-like feature can efficiently catch that character. Compared with raw pixels, the Haar-like features can efficiently reduce/increase the in-class/out-of-class variability and thus making classification easier [87]. The second reason is that a Haar-like feature-based system can operate much faster than a pixel-based system. The third advantage brought by Haar-like features is that they are more robust against noises and lighting variations than other image features such as colors. Haar-like features focus more on the difference between two or three connected areas (*i.e.* the white and black rectangles) inside an image rather than the value of each single pixel. Noises and lighting variations affect the pixel values of all areas, and this influence can be effectively counteracted by computing the difference between them.

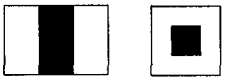
The value of a Haar-like feature is the difference between the sum of the pixels' values within the black and white rectangular regions:

$$f(x) = W_{black} \cdot \sum_{blackRec} (pixelValue) - W_{white} \cdot \sum_{whiteRec} (pixelValue)$$

where W_{black} and W_{white} are the weights that meet the compensation condition:

$$W_{black} \cdot blackRec = W_{white} \cdot whiteRec$$

For example, for the Haar-like feature shown in Figure 4.2(a), $W_{black} = 2 \cdot W_{white}$; for the



(a) (b)

Figure 4.2: The weight compensation for different Haar-like features.

Haar-like feature shown in Figure 4.2(b), $W_{black} = 8 \cdot W_{white}$. If we set $W_{white} = 1$, then the value of Haar-like feature in Figure 4.2(a) is:

$$f(a) = 2 \cdot \sum_{blackRec} (pixelValue) - 1 \cdot \sum_{whiteRec} (pixelValue)$$

The value of Haar-like feature in Figure 4.2(b) is:

$$f(b) = 8 \cdot \sum_{blackRec} (pixelValue) - 1 \cdot \sum_{whiteRec} (pixelValue)$$

The concept of "Integral Image" is used to compute the Haar-like features containing upright rectangles [57]. The "Integral Image" at the location of $p(x, y)$ contains the sum of the pixel values above and left of this pixel inclusive (see Figure 4.3(a)):

$$P(x, y) = \sum_{x' \leq x, y' \leq y} p(x', y')$$

According to the definition of "Integral Image", the sum of the grey level value within the area "D" in Figure 4.3(b) can be computed as:

$$P_1 + P_4 - P_2 - P_3$$

since

$$P_1 + P_4 - P_2 - P_3 = (A) + (A + B + C + D) - (A + B) - (A + C) = D$$

For Haar-like features containing 45° rotated rectangles, the concept of "Rotated Summed Area Table (RSAT)" was introduced by Lienhart in [87]. RSAT is defined as the sum of the pixels of a rotated rectangle with the bottom most corner at $p(x, y)$ and extending upwards to the boundaries of the image, which is illustrated in Figure 4.4(a):

$$R(x, y) = \sum_{y' \leq y, y' \leq y - |x - x'|} p(x', y')$$

According to the definition of "RSAT", the sum of the grey level value within the area "D" in Figure 4.4(b) can be computed as:

$$R_1 + R_4 - R_2 - R_3$$

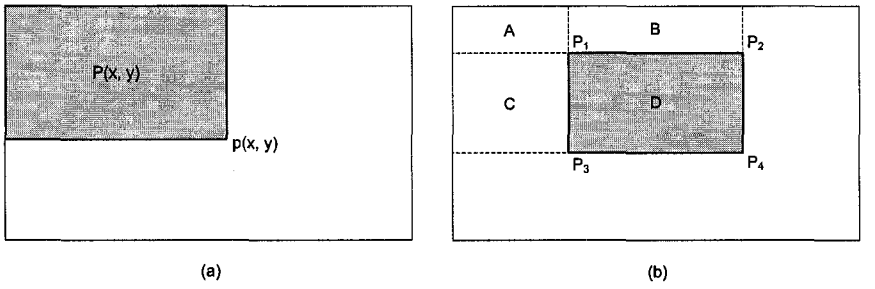


Figure 4.3: The concept of "Integral Image".

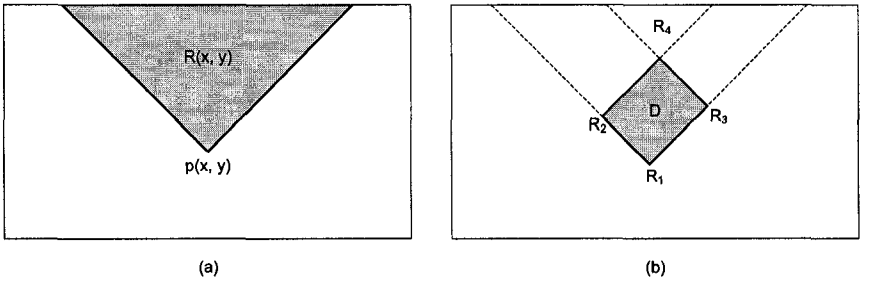


Figure 4.4: The concept of "Rotated Summed Area Table (RSAT)".

To detect an object of interest, the image is scanned by a sub-window containing a specific Haar-like feature (see the face detection example in Figure 4.5). Based on each Haar-like feature f_j , a corespondent classifier $h_j(x)$ is defined by:

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

where x is a sub-window, θ is a threshold, p_j shows the direction of the inequality sign.

Figure 4.5: Detect a face with a sub-window containing a Haar-like feature.

4.2 AdaBoost Learning Algorithm

In practice, no single Haar-like feature can identify the object with a high accuracy. However, it is not difficult to find one feature-based classifier that has a slightly better accuracy than random guessing (*i.e.* better than 50%). In machine learning, these "better than random guessing" classifiers are called "weak classifiers". Boosting is a general method to improve the accuracy of a given learning algorithm stage by stage based on a series of weak classifiers [88]. A weak classifier is trained with a training set at each step. The trained weak classifier is then added to the learned function with a strength parameter proportional to the accuracy of this weak classifier. The training samples missed by the current weak classifier are re-weighted with a bigger value and the future weak classifier will attempt to fix the errors made by the current weak classifier so that the overall accuracy can be improved.

Figure 4.6 shows the overall structure of the AdaBoost learning algorithm. The whole algorithm is based on a set of Haar-like features. For each selected Haar-like feature, a correspondent weak classifier is constructed. Based on a series of weak classifiers, the final cascade classifiers are composed, which can quickly improve the accuracy and efficiency. With the cascade classifiers, a sub-window scans the input image to detect objects at different locations. The size of the sub-window can be stretched for each scanning process so that the objects of different scales can be detected.

The AdaBoost learning algorithm, which is first introduced by Freund and Schapire, solved many practical difficulties of the earlier boosting algorithms [89]. As illustrated in Figure 4.7, the AdaBoost learning algorithm initially maintains a uniform distribution of

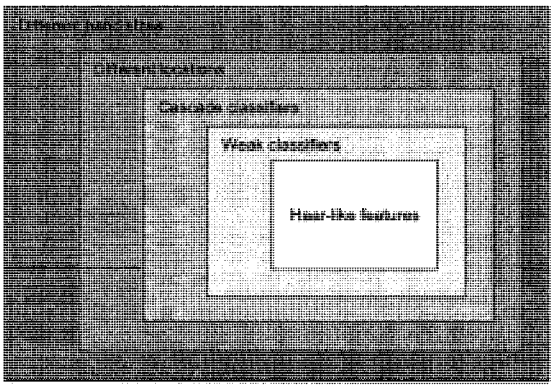


Figure 4.6: The structure of the AdaBoost learning algorithm.

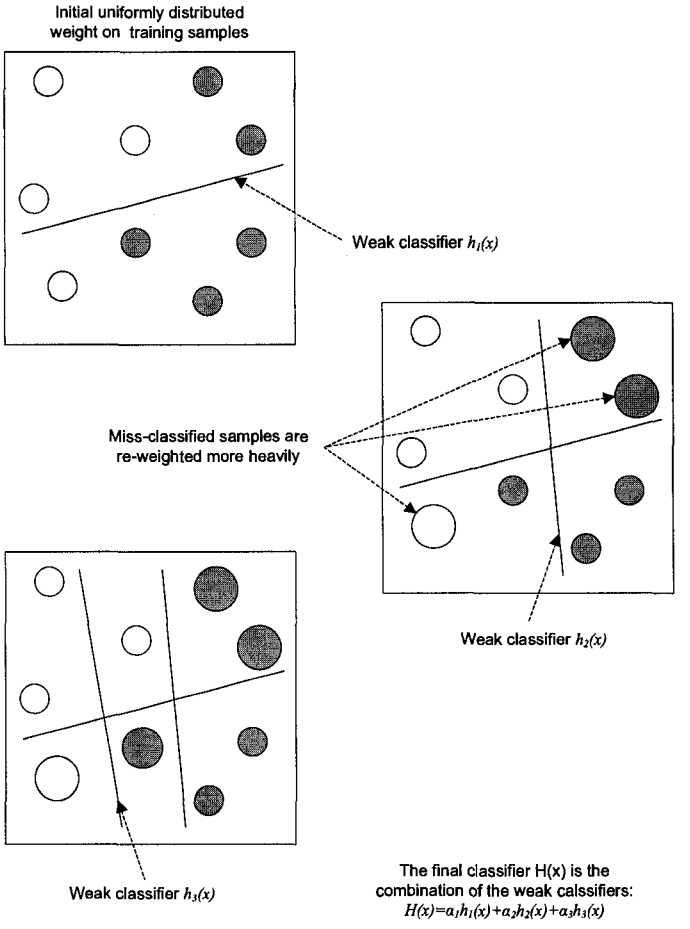


Figure 4.7: The boosting process of the AdaBoost learning algorithm.

weights over each training sample (in our case, the hand posture images). It should be noted that a Haar-like feature could be used repeatedly in the training process. We start with the selection of the first Haar-like feature in the boosting process. The training algorithm keeps the Haar-like feature that yields the best classification accuracy in the first iteration. The classifier based on this Haar-like feature is added to the linear combination with a strength proportional to the resulting accuracy. For the next iteration, the training samples are re-weighted: training samples missed by the first weak classifier are boosted in importance so that the second selected Haar-like feature must pay more attention towards these misclassified samples. To be selected, the second Haar-like feature must achieve a better accuracy for these miss-classified training samples so that

the error can be reduced. The iteration goes on by adding new weak classifiers based on selected Haar-like features to the linear combination until the required overall accuracy is achieved. The final training result is a strong classifier composed by a linear combination of the selected weak classifiers.

The detailed steps of the AdaBoost learning algorithm are [57]:

- Given training images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- For $t = 1, \ldots, T$:
  1. Start with the weights: $w_{1,i} = \frac{1}{n}, i = 1, \ldots, n$.
  2. For each feature j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i |h_j(x_i) - y_i|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights: $w_{t+1,i} = w_{t,i} \cdot \beta_t^{1-e_i}$, where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$, re-normalize weights so that $\sum_i w_{t+1,i} = 1$.
- The final strong classifier is:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

  where $\alpha_t = log\frac{1}{\beta_t}$.

In practical implementation, a cascade of classifiers structure is employed to speed up the performance. At each stage of the cascade, a classifier, which is composed of a group of weak classifiers, is trained to detect almost all of the positive samples while rejecting a certain fraction of negative samples. For example, we set the negative samples' elimination percentage at 50% for the classifier at each stage while keep the positive samples' elimination percentage at 0.2%. If we obtained a cascade of classifiers with 15 stages of at the end of training, the expected false alarm rate will be $0.5^{15} \approx 3.1 \times 10^{-5}$, and the detection rate will be $0.998^{15} \approx 0.97$ (see Figure 4.8). The reason for this strategy is based on the fact that the majority of the sub-windows are background (*i.e.* negative) within a single image frame, and it is a rare event for a positive instance to go through all of the stages. With this strategy, the cascade can significantly save the processing

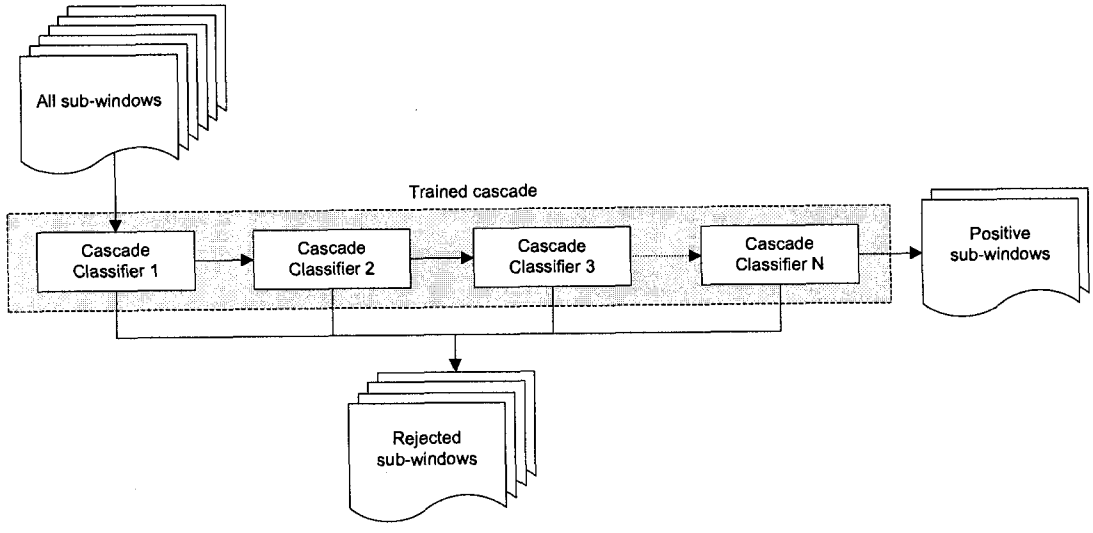


Figure 4.8: Detect positive sub-windows using the cascade classifiers.

time as the initial stage of classifiers try to quickly remove a large fraction of background as illustrated in Figure 4.9, and more computation will be focused on the more difficult sub-windows which passed the scrutiny of the initial stages of the cascade.

The object of interest is detected and tracked frame by frame from the images of the video input. To detect the object of interest, each frame of input image series is swept from the top-left corner to the bottom-right corner by a stretchable sub-window. The sub-window of a specific size scans the picture pixel by pixel. As the hand size

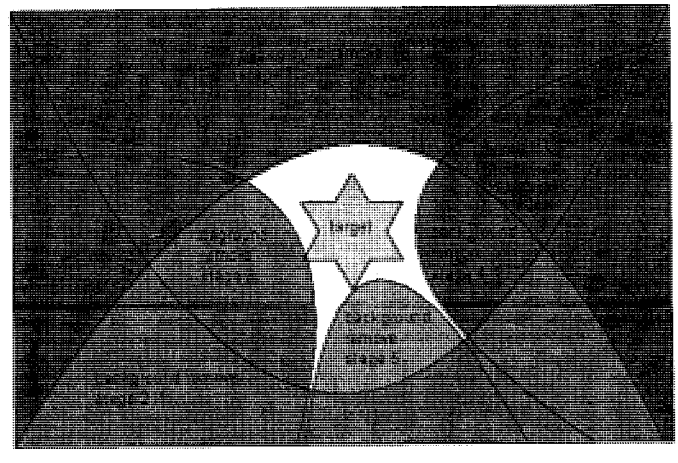


Figure 4.9: Rapid background removal by the cascade classifiers.

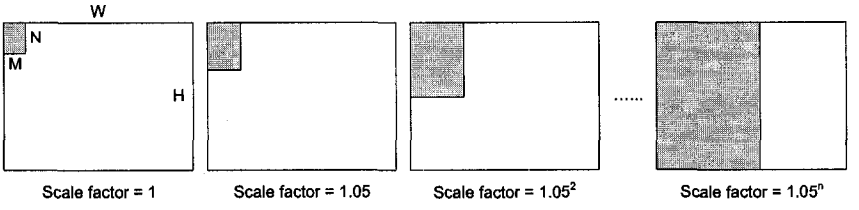


Figure 4.10: Stretch the sub-window's size with the scale factor.

in the video input varies, the size of the sub-window also changes consistently so that different hand sizes can be detected (see Figure 4.10). For the stretchable sub-window, the ideal approach is to start from a very small initial kernel size and increase its width and hight for one pixel at each scan. However, this strategy will result in a large number of sub-windows, and it is not feasible if the real-time requirement is to be met. To avoid the excessive computation cost, a number of sub-windows have to be discarded. A scale factor is used to implement this strategy. For instance, to detect a hand palm within an image frame, we can start the sub-window with an initial kernel size of 15×30 and a scale factor of 1.05. For the next scan, the sub-window's size will be stretched to $(15 \times 1.05) \times (30 \times 1.05)$. This process goes on until one side of the sub-window reaches the image frame. According to the example shown in Figure 4.10, if the size of the input image is $W \times H$, the initial kernel size is $M \times N$ and the scale factor is f, the total number of sub-windows that have to be processed is:

$$\sum_{i=0}^{n}(W - M \cdot f^i)(H - N \cdot f^i)$$

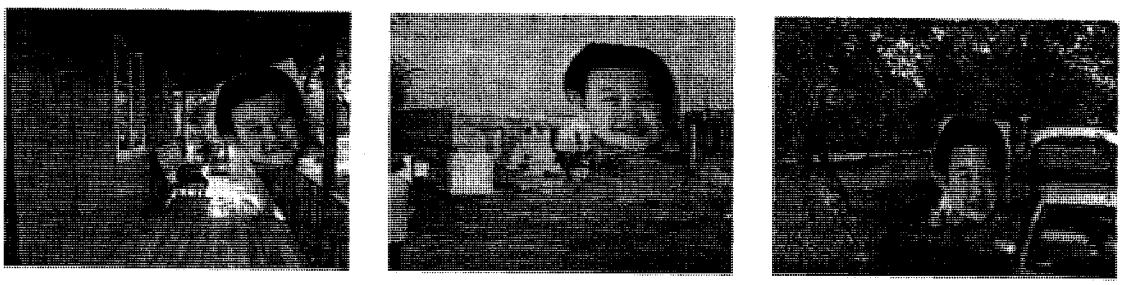
where n is the number of times the scale factor has been increased, which satisfies $M \cdot f^n \leq W$ or $N \cdot f^n \leq H$. The bigger the scale factor, the faster the computation. However, if the scale factor is too big, the object with a size in between may be missed by the sub-window.

4.3 Robustness Evaluation

To test the robustness of Haar-like features and the AdaBoost learning algorithm, we use the cascade classifier for front face detection provided by OpenCV, which is the benchmark for vision-based object detection [90]. This face detection cascade classifier is trained with 5000 positive frontal face samples and 3000 negative samples. Each stage

is trained to reject about half of the negative samples, while correctly accepting 99.9% of the face samples. The final trained cascade consists of 20 stages [91].

We first evaluate the robustness against image rotations, which includes in-plane rotations and out-of-plane rotations. In-plane rotation means the image is rotated for a certain degree around "Z" axis perpendicular to the image plane such as the example shown in Figure 4.11(a). Out-of-plane rotations are the rotations around "X" axis or "Y" axis (*e.g.* Figure 4.11(b) and Figure 4.11(c)).



(a) Rotation around "Z" axis (b) Rotation around "X" axis (c) Rotation around "Y" axis

Figure 4.11: Rotated face images superimposed on random backgrounds.

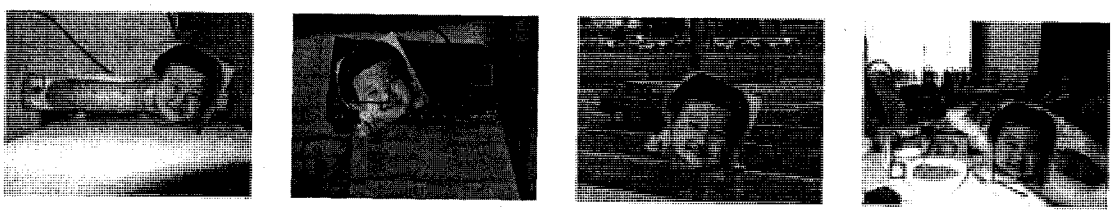


Figure 4.12: Test results for face images rotated around "Z" axis.

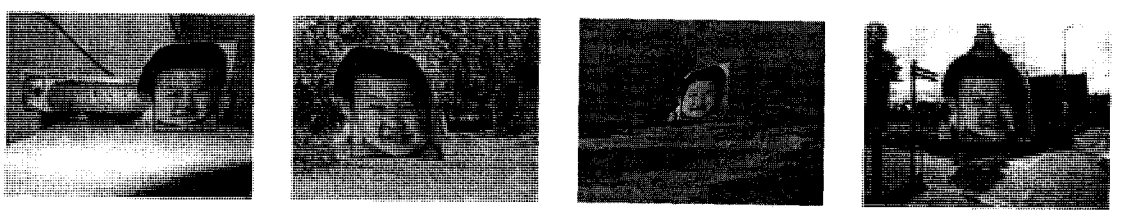


Figure 4.13: Test results for face images rotated around "X" axis.

We generate 500 test samples with a rotated face image superimposed on random backgrounds as Figure 4.11 shows. The rotation range is from 0° to 20° with a step of 5°. For out-of-plane rotations, we also generate 500 test images with the rotation from

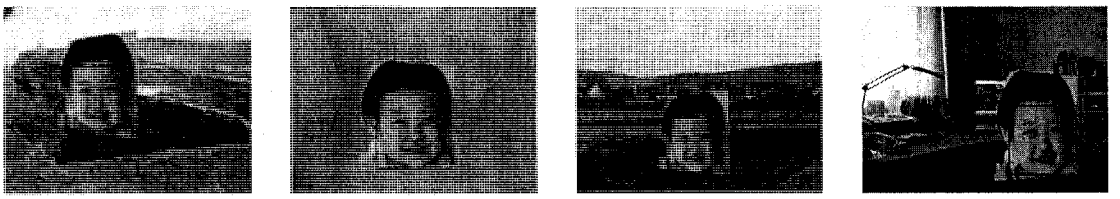


Figure 4.14: Test results for face images rotated around "Y" axis.

0° to 40° at the step of 10°. Some test results are shown in Figure 4.12, Figure 4.13 and Figure 4.14.

Figure 4.15 shows the detection rates corresponding to different in-plane rotation degrees. The detection rate reduces to 83.8% when the maximum rotation degree reaches 10°. The detection rate further reduces to 62% when the rotation degree reaches 15°. Figure 4.16 and Figure 4.17 show the detection rates corresponding to different out-of-plane rotations. The detection rates are kept around 99% when the rotation reaches 20°. When the rotation reaches 30°, the detection rates reduce to 87.8% and 82.4% respectively for rotations around X axis and Y axis.

To evaluate the robustness against different lighting conditions, we adopt the model

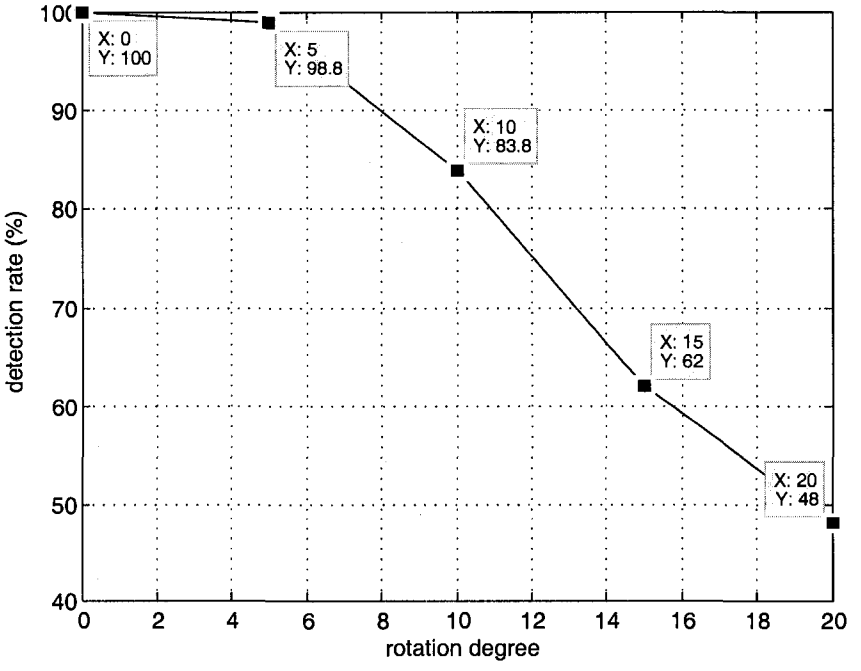


Figure 4.15: The robustness evaluation for images rotated around "Z" axis.

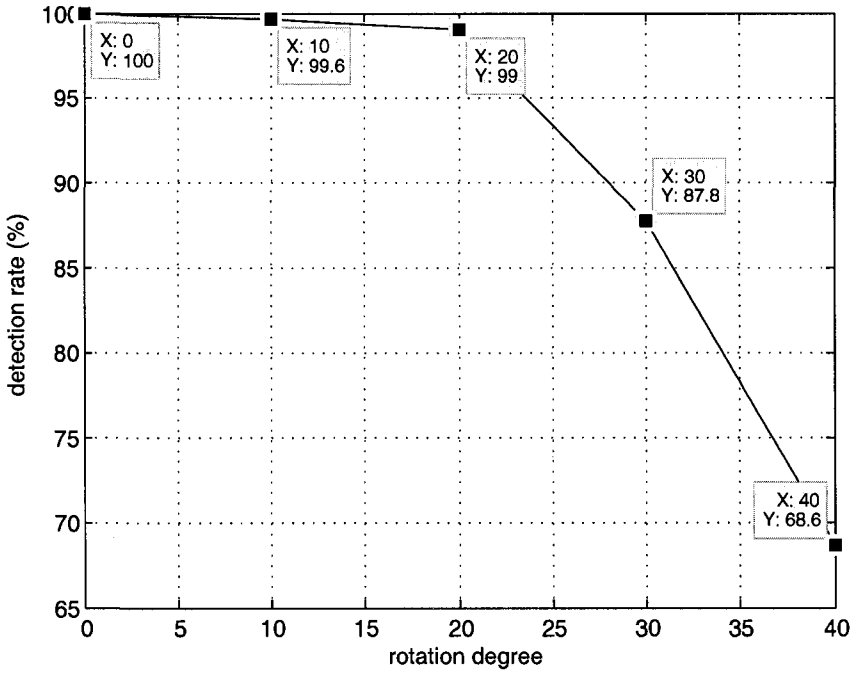


Figure 4.16: The robustness evaluation for images rotated around "X" axis.

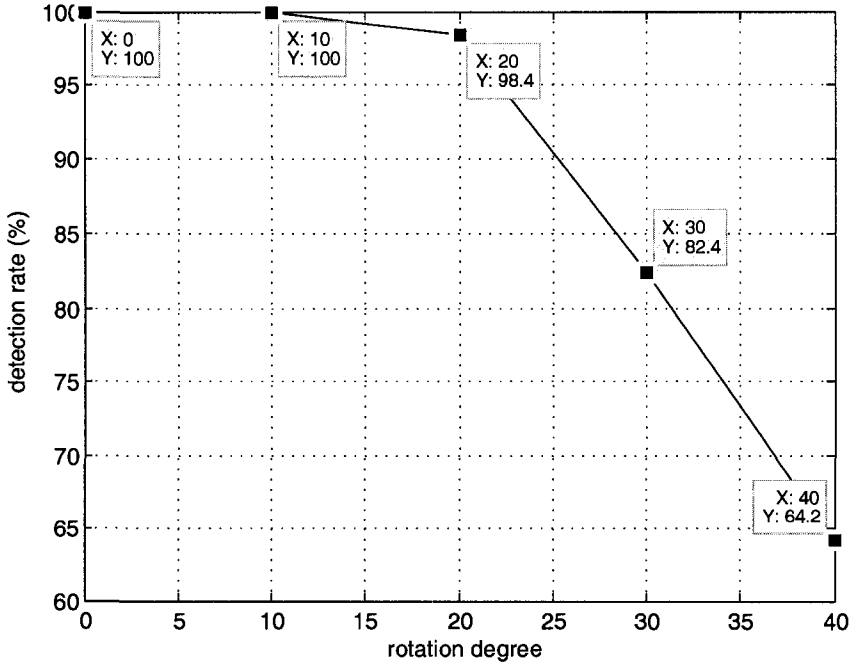


Figure 4.17: The robustness evaluation for images rotated around "Y" axis.

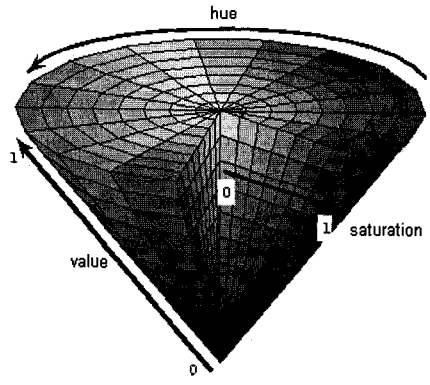
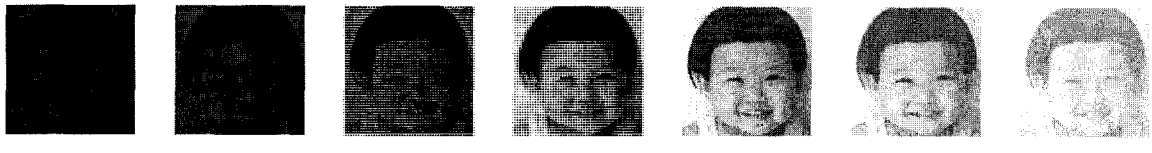


Figure 4.18: The HSV color space (from [92]).

of HSV (Hue, Saturation, Value) color space shown in Figure 4.18 [92]. In HSV color space, hue is the color reflected from an object. Saturation is the strength or purity of the color. Value is the relative brightness or darkness of the color, usually measured as a percentage from 0% (black) to 100% (white).

We variate the brightness of the test images by adjusting the "Value" range. For example, the "Value" range of an original image is from 0% to 100%, to brighten the image, we can narrow this range to 60% to 100% (*i.e.* +60%), and this shifting can brighten the image accordingly. Figure 4.19 shows the test images with different brightness values. We narrow the range of the input image from original to ±40%, ±60% and ±80%. At each brightness value, 500 test images are generated by superimposing the image on random backgrounds. Based on the evaluation results, it is noticed that the front face cascade classifier achieved 100% detection rate for all of the test images of different brightness. Figure 4.20 shows some test results.



(a) -80% (b) -60% (c) -40% (d) original (e) +40% (f) +60% (g) +80%

Figure 4.19: Test images with different brightness values.

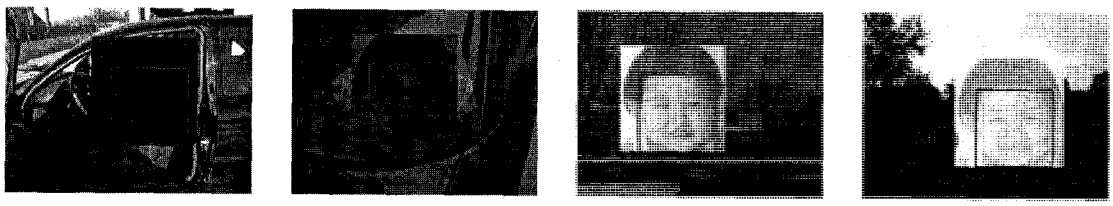


Figure 4.20: Test results for face images with different brightness values.

4.4 Data Collection

The Haar-like features can achieve very robust detection for human faces which have a single non-concave shape (*i.e.* a roughly round shape with comparatively consistent positions of eyes, nose, and mouth). However, for the a complex articulated human hand with high DOF, different image patterns are projected on the 2D plane for different hand postures. Many of the projections are concave shapes such as the the pointing posture and the "V" posture. Part of the background are inevitably included when align the rectangle-shaped Haar-like features with these concave posture images. To minimize the standard deviation of the positive samples so that it will be easier for the AdaBoost learning algorithm to generalize the class, we decide to use a plain white background instead of cluttered backgrounds for the hand postures.

All of the positive samples are collected in the laboratory with a natural fluorescent lighting condition. To variate the illumination, we installed an extra incandescent light bulb to create a tungsten lighting condition. During the positive samples collection phase, we intentionally include a number of positive samples with a certain degree of in-plane and out-of-plane rotations according to the limits of our previous robustness evaluation results for Haar-like features. As all of the positive samples look similar, it is therefore not important to overfill the positive set with a large number of samples.

We collected 450 positive samples for each selected posture in Table 3.1. Figure 4.21 is a fraction of the positive samples for the "two fingers" posture. The scale of the posture image does not affect the training process at this point because we will mark out the hand area in all of the positive samples and adjust them to a unified size of 15×30 for the final training.

We collected 500 random images as negative samples for the training process. It is a must that these images do not contain any of the hand posture images. Figure 4.22 shows some negative samples we collected. The negative samples should vary with respect to their gray levels, shapes, edges, textures, and orientations so that the weak classifier can

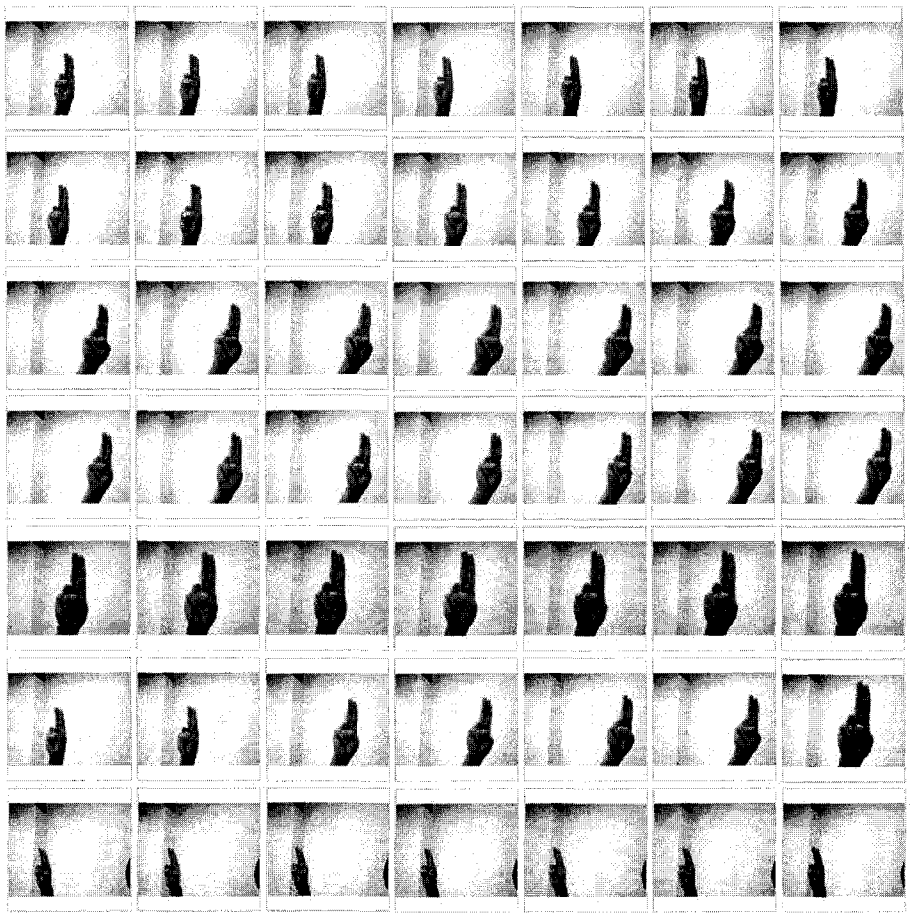


Figure 4.21: A fraction of the positive samples for the "two fingers" posture.

achieve a better accuracy by getting rid of false samples with a large variety. All negative samples are passed through a background description file which contains the filenames (relative to the directory of the description file) of all negative sample images.

The numbers of the positive samples and negative samples are based on our experiment result. When the cascades trained with 450 positive and 500 negative samples already come close to the representation power, larger training sets do not affect the training result significantly. However, the training time will be increased dramatically by a larger number of training sets.

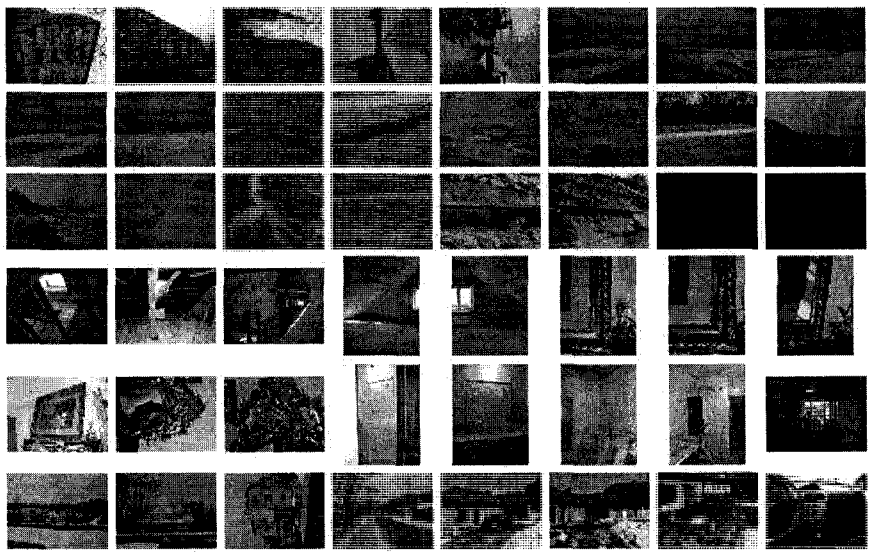


Figure 4.22: A fraction of the negative samples used in the training process.

4.5 Training and Testing

We set the minimum desired hit rate at 99.5%, and the maximum desired false alarm rate at 50% for each stage of the cascade of classifiers. The overall false alarm rate must meet the accuracy of 1×10^{-6} to terminate the training process. A 15-stage cascade is trained for the "two-finger" posture. The hit rate of the final classifier is 97.5% when the required false alarm rate is met. For the "palm" posture, a 10-stage cascade is trained with a hit rate at 98%. For the "fist" posture, a 15-stage cascade is trained with a hit rate at 98%. For the "little finger" posture, a 14-stage cascade is trained with a hit rate at 97%. Table 4.2 lists the initial Haar-like features selected by the trained cascades of classifiers for different hand postures. It is noticed that the selected Haar-like features do reflect the key image features of their correspondent hand postures.

Table 4.1: The performance of the trained cascades of classifiers.

Classifier	**Samples Detected**	**Samples Missed**	**False Alarm**	**Processing Time**
Two Fingers	100	0	29	3.0 seconds
Palm	90	10	0	1.9 seconds
Fist	100	0	1	2.8 seconds
Little Finger	93	7	2	2.5 seconds

Table 4.2: The initial Haar-like features selected by the trained cascades of classifiers.

Hand Postures	Haar-Like Features
	
	
	
	

In order to evaluate the performance of the cascade classifiers, 100 marked-up test images for each posture is collected. The scale factor is set at 1.05 since it has the best balance between the detection rate and the processing speed. Table 4.1 shows the performance of the four cascades of classifiers and the time spent to detect all 100 test images. Figure 4.23, Figure 4.24, Figure 4.25 and Figure 4.26 showed some of the detection pictures for each posture.

By analyzing the detection results, we find that some of the missed positive samples are caused by the excessive in-plane rotations. For the false alarms, the majority of them happened in very small areas in the image scene, which contains similar black and right patterns to be detected by the selected Haar-like features. These false detections can be easily removed by defining a threshold of the minimum size for the detected object. Among the four classifiers, the "fist" classifier achieves the most stable performance. The major reason for this is because the "fist" posture has a comparatively uniformed round contour (similar to the human face) and displays a set of stable image features which can be detected more easily by the Haar-like features.

The times required for classifiers to detect 100 testing images are all within 3 seconds. We tested the real-time performance with live input from the web-camera with 15 frames-per-second at the resolution of 320 $\times$ 240, and there is no detectable pause and latency to track and detect the hand postures with all our trained classifiers.

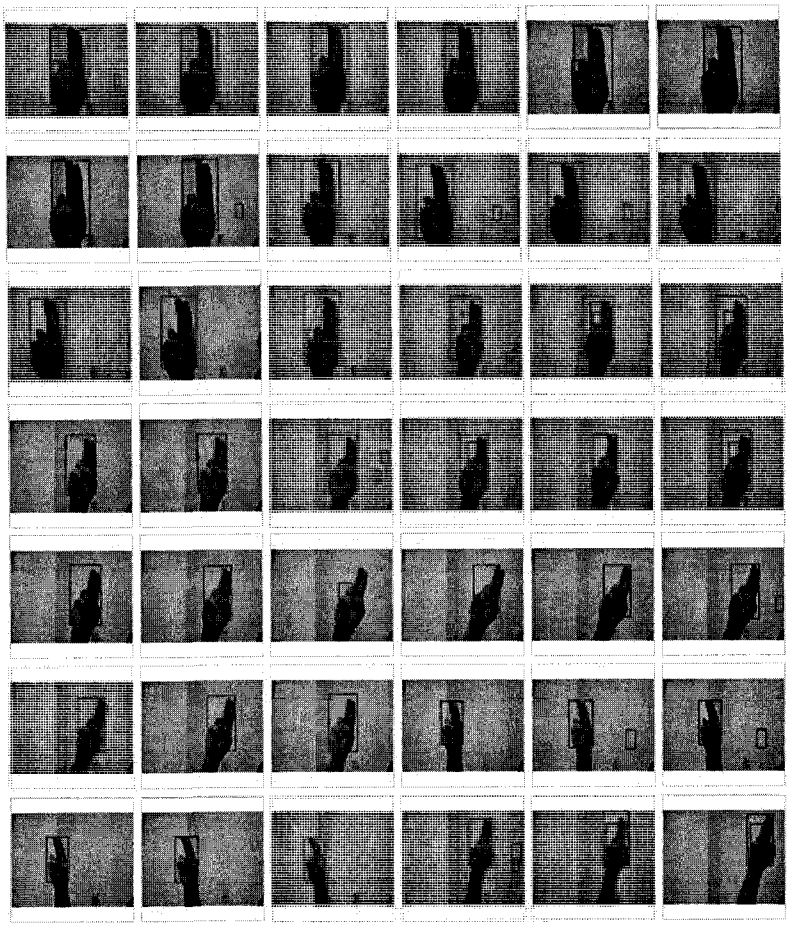


Figure 4.23: A fraction of the testing results for the "two fingers" cascade classifier.

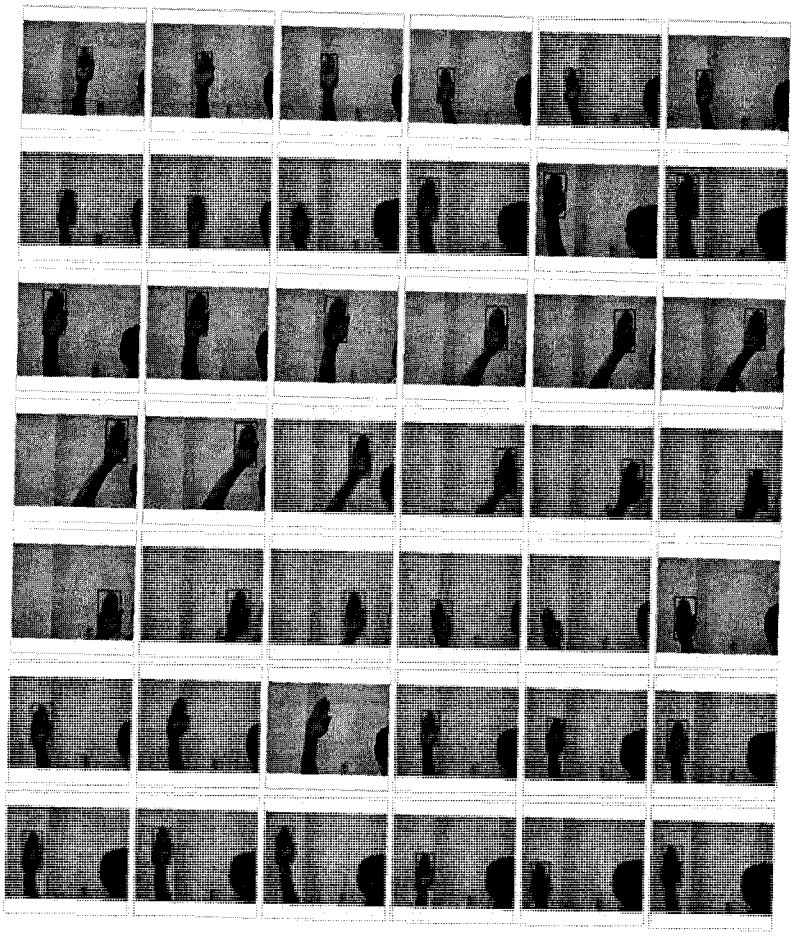


Figure 4.24: A fraction of the testing results for the "palm" cascade classifier.

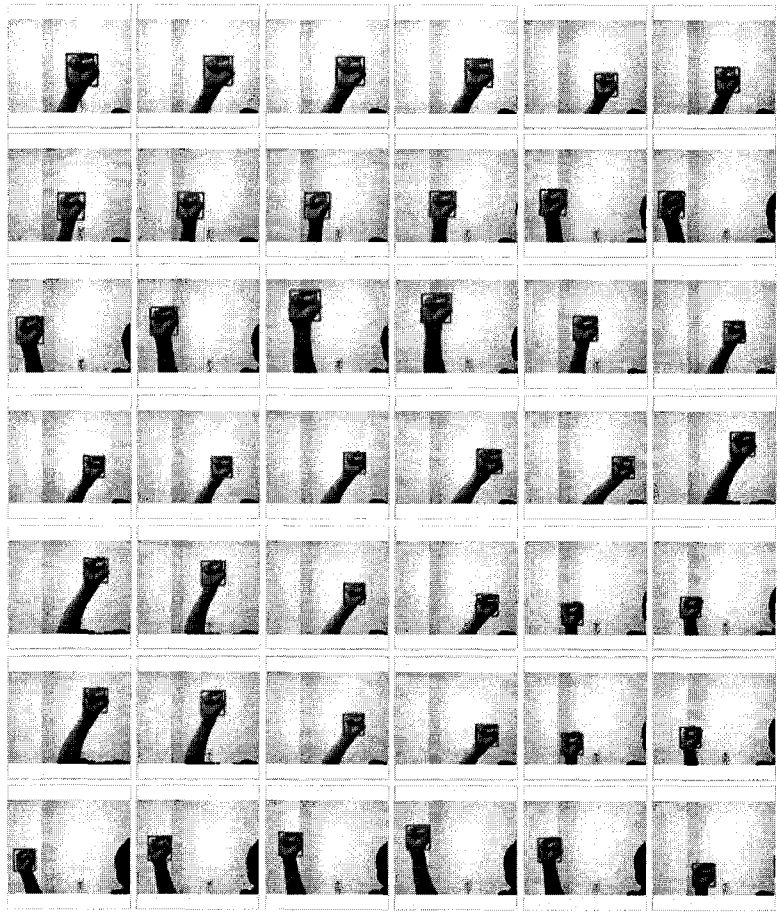


Figure 4.25: A fraction of the testing results for the "fist" cascade classifier.

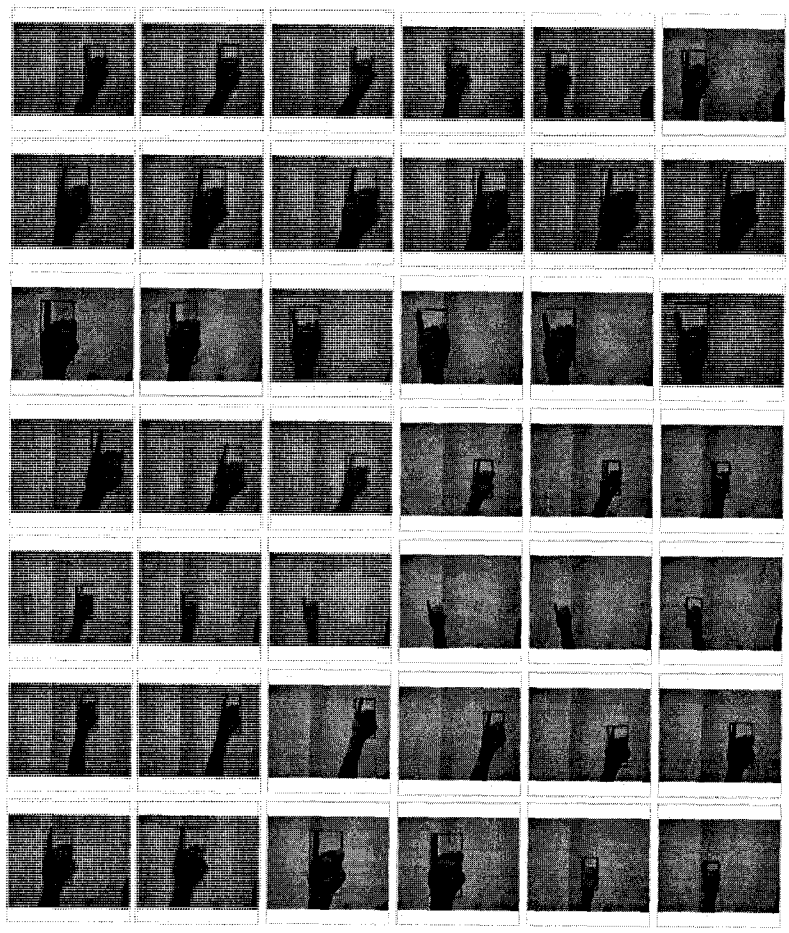


Figure 4.26: A fraction of the testing results for the "little finger" cascade classifier.

4.6 A Parallel Cascades Structure

A bank of posture-specific cascade classifiers is assembled for a parallel recognition of the selected hand postures (Figure 4.27). In this structure, multiple cascades are loaded into the system simultaneously, and each cascade is responsible for a specific hand posture. This parallel structure effectively increases the speed of the hand posture recognition process. It is noticed that the parallel structure of four cascade classifiers allows us achieving real-time recognition of the selected hand postures from the live input of the camera with 15 frames-per-second at the resolution of 320×240. There is no detectable pause or latency for the parallel structure to identify the hand postures and tracking them at the same time.

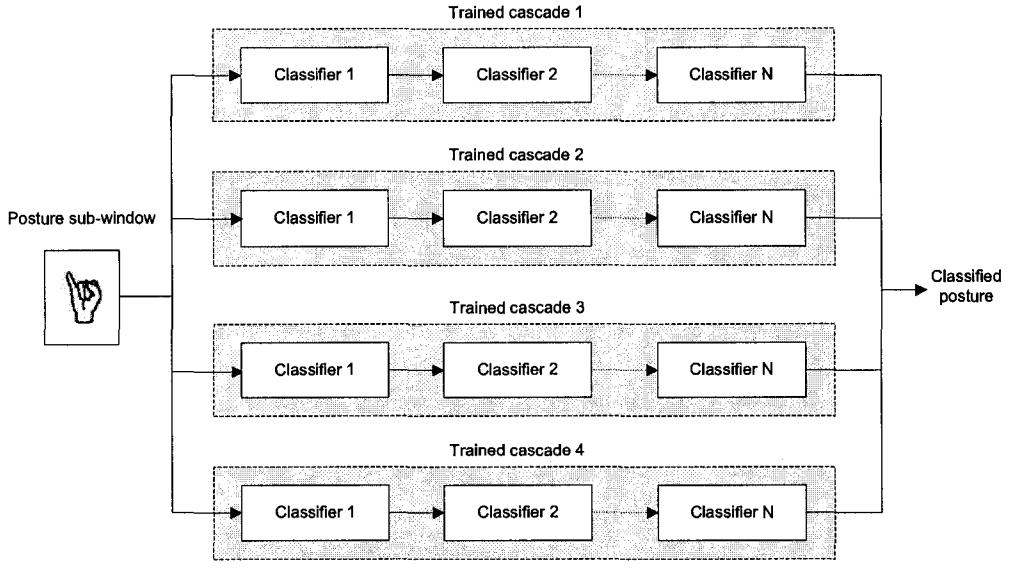
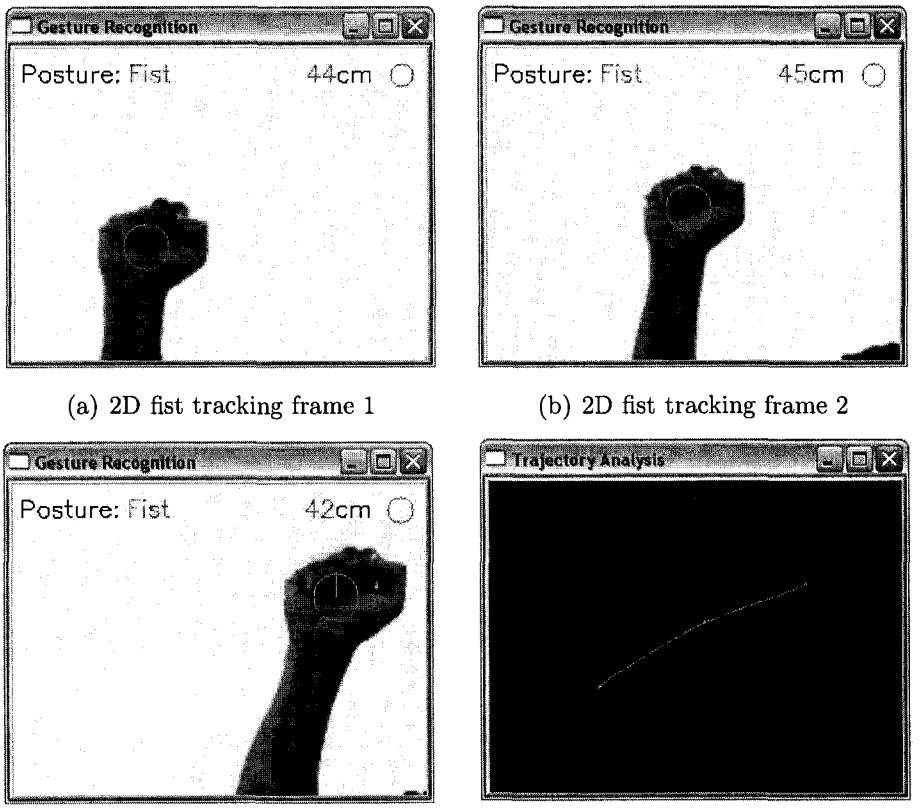


Figure 4.27: The parallel cascades structure for hand posture classification.

4.7 3D Hand Tracking and Background Subtraction

The 3D position of the hand is recovered according to the location and the size of the sub-window as well as the camera's perspective projection. For each frame, if a hand posture is detected by a sub-window, the center coordinates of the sub-widow is pushed into a stack so that the 2D position of the hand posture can be kept, and the correspondent trajectory can be drawn (see Figure 4.28). As illustrated in Figure 4.28(c), a compass is



(a) 2D fist tracking frame 1

(b) 2D fist tracking frame 2

(c) 2D fist tracking frame 3

(d) Trajectory

Figure 4.28: 2D tracking for the hand posture "fist".

displayed at the center coordinates of the detected posture showing the motion direction of the hand movement. For our implementation, the major challenge is to recover the depth information z, which is the distance between the camera and the hand. The camera performs a linear transformation from the 3D projective space to the 2D projective space. With the known camera's intrinsic parameters, according to the perspective projection illustrated by Figure 4.29, the projected point (x', y', z') on the image plane satisfies the triangle equations:

$$\frac{x'}{x} = \frac{y'}{y} = \frac{f}{z}$$

where f is the focal length of the camera. Based on the triangle equation, we can get the perspective projection equations:

$$x' = \frac{f}{z}x, \; y' = \frac{f}{z}y$$

According to above equations, if the sizes of the user's hand and the sub-window are

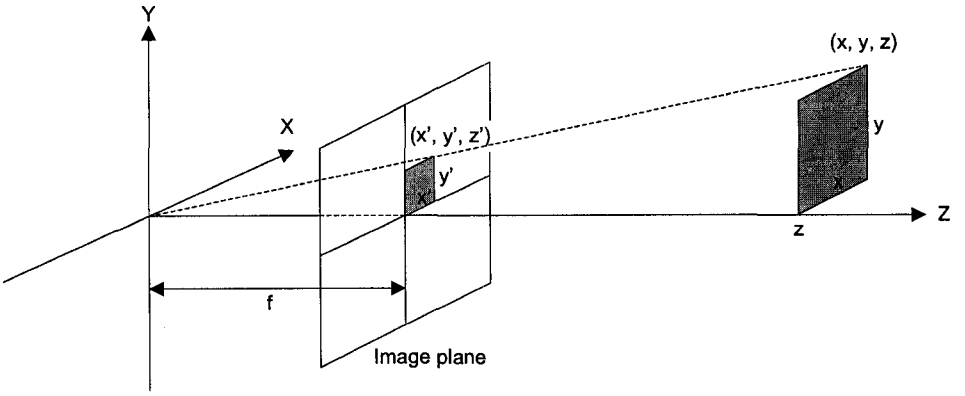


Figure 4.29: The camera's perspective projection.

known, the depth information z can be calculated according to:

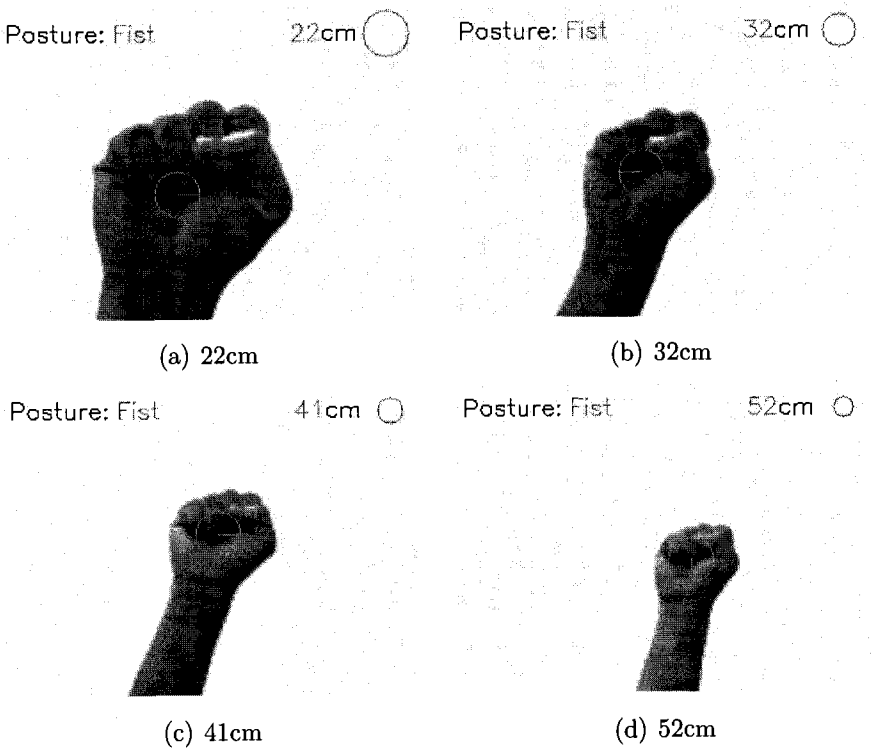
$$z = \frac{x}{x'} \cdot f = \frac{y}{y'} \cdot f$$

As long as we can consistently find the size of the sub-window for the detected hand postures from the video input, the depth data z can be computed according to above analysis. As illustrated in Figure 4.30, the recovered depth information is shown at the top-right corner of the window, and a zoom circle is drawn to intuitively reflect the distance between the camera and the user's hand. The error of the recovered distance is less than 2cm according to our experiment results for a distance range from 20cm to 150cm.

Background subtraction is used by the system to achieve the robustness against cluttered backgrounds. Figure 4.31 illustrates the background subtraction process. The system takes the first input image (no postures included in the image scene) as the background (see Figure 4.31 (a)), and subtracts this image from later input frames. To reduce the noise produced after the background subtraction (see Figure 4.31 (c)), median filter and image dilation/erosion are used. Figure 4.31 (d)) shows the effect of the smoothing measures on the resulting image. It is noticed the performance of the Haar-like features were improved after the noise removal measure is taken.

4.8 Summary

This chapter presents the low-level of the architecture: hand posture detection and tracking. To meet the real-time requirement, we trained a series of cascades of classifiers for

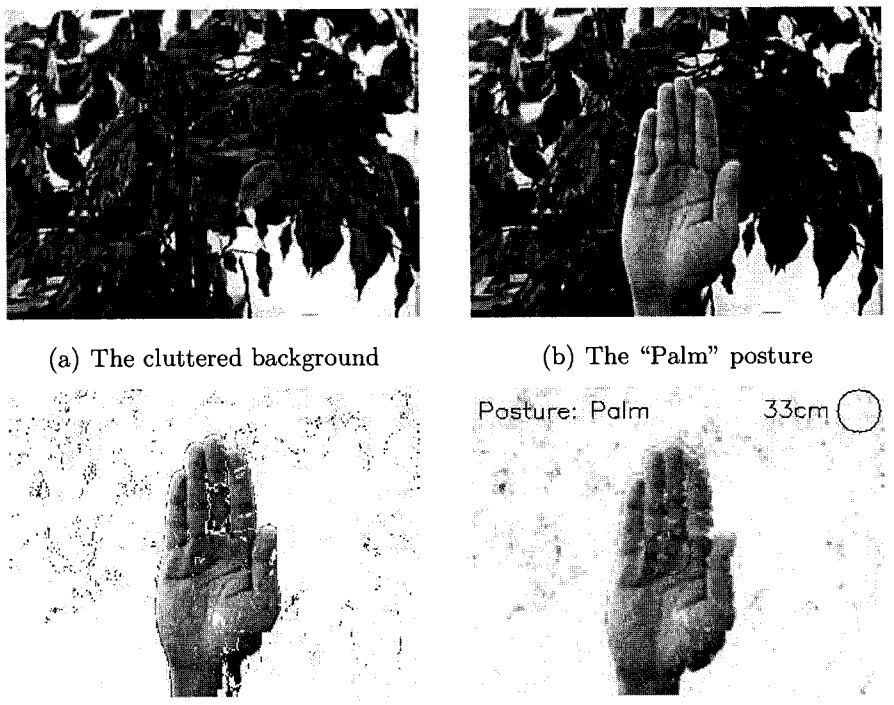


(a) 22cm (b) 32cm

(c) 41cm (d) 52cm

Figure 4.30: The depth information recovered according the perspective projection.

each hand posture based on Haar-like features and the AdaBoost learning algorithm, which can achieve fast, accurate and robust performance. A parallel cascades structure is implemented to identify the selected hand postures from the camera's live input. The 3D position of the hand is recovered according to the location of the sub-window and the perspective projection based on the camera's intrinsic parameters. Background subtraction and noise removal are used to achieve the robustness against cluttered backgrounds.



(a) The cluttered background

(b) The "Palm" posture

(c) Background subtraction

(d) Smoothing and noise removal

Figure 4.31: The background subtraction and noise removal.

Chapter 5

Hand Gesture Recognition and Motion Analysis

5.1 Stochastic Context-Free Grammars

As a hand gesture is an action composed of a sequence of hand postures, it makes the syntactic approach appropriate to describe the hand gesture in terms of its constituent hand postures. With the syntactic approach, hand gestures can be specified as building up by a group of hand postures in various ways of composition, just as phrases are built up by words. The rules governing the composition of hand postures into different hand gestures can be specified by a grammar. The syntactic approach provides the capability to describe a set of complex hand gestures by using a small group of simple hand postures and a set of grammatical rules.

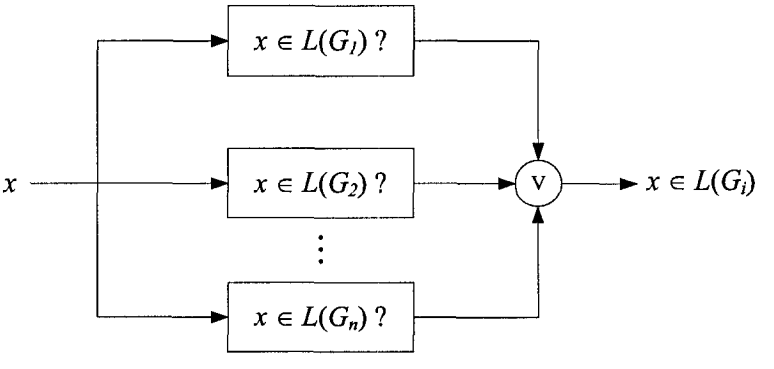


Figure 5.1: The parallel structure of syntactic classifiers.

When a grammar is constructed to describe the patterns under study, a syntactic classifier needs to be designed so that the patterns can be recognized. There are two approaches to design a syntactic classifier. The straightforward approach is to construct a grammar for each pattern class, and the correspondent syntactic classifier will recognize only the patterns belonging to this class. As illustrated in Figure 5.1, for an unknown pattern x entering a parallel structure constructed by n syntactic classifiers, the problem of recognition is essentially to make the decision:

$$Is\ x \in L(G_i), \quad for\ i = 1, \ldots n?$$

If the language generated by a grammar is finite with a reasonable size, the syntactic classifier can search for a match between the unknown pattern and all the words of the language. When the grammar is complex and the generated language involves a large number of words, it is more appropriate to use a syntactic analysis approach (*i.e.* parsing) based on whether the production rules of the grammar can generate the unknown pattern. If the parsing process is successful, which means the unknown pattern can be generated by this grammar, we can classify this unknown pattern to the class represented by this grammar. If the parsing process is not successful, then the unknown pattern is not accepted as an object of this class.

Syntactic classifier can also be constructed based on the production rules of a grammar. With this approach, each unknown pattern corresponds to a specific word generated by the grammar. By analyzing the correspondent production rule as well as the associated probability (*e.g.* stochastic grammars), the unknown pattern can be classified according to the production rule which has the greatest probability to generated the pattern.

In practical applications such as hand motion analysis, a certain amount of uncertainty exists such as distorted movement trajectories. Grammars used to describe distorted patterns are often ambiguous in the sense that a distorted pattern might be generated by more than one grammar. Under these situations, stochastic grammars can be used in order to avoid the confusions caused by the distorted patterns.

Stochastic context-free grammars (SCFG) are used in our system to describe the structural and dynamic information about hand gestures. The SCFG is an extension of the context-cree grammar (CFG). The difference between SCFGs and CFGs is that for each production rule in SCFGs, there is a probability associated with it. Each SCFG is a four tuple:

$$G_S = (V_N, V_T, P_S, S)$$

where V_N and V_T are finite sets of nonterminals and terminals; $S \in V_N$ is the start symbol; P_S is a finite set of stochastic production rules each of which is of the form:

$$X \xrightarrow{P} \lambda$$

where $X \in V_N$, $\lambda \in V^+$ (i.e., V_N, V_T, or the combination of them) and P is the probability associated with this production rule. The probability P can also be expressed as $P(X \rightarrow \lambda)$, and it satisfies:

$$\sum_j P(X \rightarrow \mu_j) = 1$$

where μ_j are all of the strings that are derived from X. In SCFG, the notion of context-free essentially means that the production rules are conditionally independent [93].

If a string $y \in L(G_S)$ is unambiguous and has a derivation with production rules $r_1, r_2, \ldots, r_k \in P_S$, then the probability of y with respect to G_S is:

$$P(y|G_S) = \prod_{i=1}^{k} P(r_i)$$

If y is ambiguous and has l different derivation trees with corresponding probabilities $P_1(y|G_S)$, $P_2(y|G_S)$,..., $P_l(y|G_S)$, then the probability of y with respect to G_S is given by:

$$P(y|G_S) = \sum_{i=1}^{l} P_i(y|G_S)$$

SCFGs extend CFGs in the same way that Hidden Markov models (HMMs) extend regular grammars. The relation between SCFGs and HMMs is very similar to that between CFGs and non-probabilistic Finite State Machines (FSMs), where CFGs relax some of the structural limitations imposed by FSMs, and because of this, SCFGs have more flexibilities than HMMs [77]. Compared with non-stochastic CFGs, with the probability attached to each production rule, SCFGs provide a quantitative basis for ranking and pruning syntactic analysis [79]. With SCFGs, we just need to compute the probability of the pattern belonging to different classes (each class is described with its own grammar), and then choose the class with the highest probability value. The probabilistic information allows us to use statistical techniques not only for finding best matches, but also for implementing robust feature detection guided by the grammar's probability structure.

5.2 Hand Gesture Recognition

Based on the classified postures from the low-level, a simple SCFG is defined to describe the local finger motions which generate three different gestures: "Grasp", "Quote", and "J". Each gesture is composed of two postures as illustrated in Figure 5.2. The order of the postures included can be reversed so that each gesture is a repetitive action which maximize the comfort for the user. The SCFG that generates these gestures is defined by $G_G = (V_{NG}, V_{TG}, P_G, S)$, where

$$V_{NG} = \{S\},\ V_{TG} = \{p, f, t, l\}$$

and P_G:

$$r_1 : S \xrightarrow{40\%} pf,\ r_2 : S \xrightarrow{35\%} tf,\ r_3 : S \xrightarrow{25\%} lf$$

The terminals p, f, t, l stand for the four postures: "palm", "fist", "two fingers" and "little finger". The probabilities assigned to different production rules take two aspects into account: the user's preference for different gestures and the avoidance of confusion for distorted gestures.

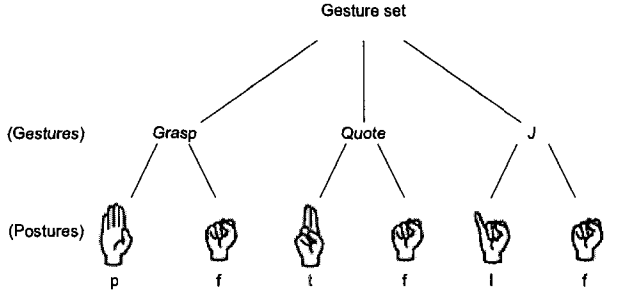


Figure 5.2: The hierarchical composition of the gesture set.

A pipe structure shown in Figure 5.3 is implemented to convert the input postures into a sequence of terminal strings.

After a string "x" is converted from the postures, we can decide the most likely product rule that can generate this string by computing the probability:

$$P(r \Rightarrow x) = D(z_r, x)P(r)$$

$D(z_r, x)$ is the similarity between the input string "x" and "z_r", which is the string derived by the production rule "r". $D(z_r, x)$ can be computed according to:

$$D(z_r, x) = \frac{Count(z_r \cap x)}{Count(z_r) + Count(x)}$$

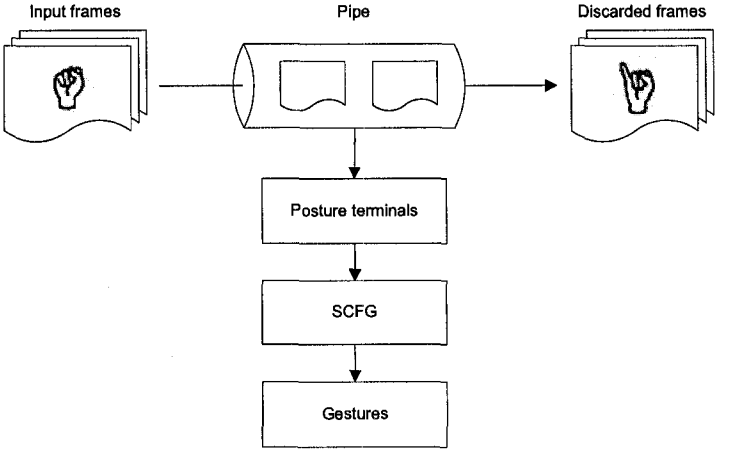


Figure 5.3: The pipe structure to convert hand postures to terminal strings.

Example 5.1: Let us consider an input string is detected as "pf", since

$$P(r_1 \Rightarrow pf) = \frac{2}{4} \times \frac{40}{100} = 20\%$$
$$P(r_2 \Rightarrow pf) = \frac{1}{4} \times \frac{35}{100} = 8.75\%$$
$$P(r_3 \Rightarrow pf) = \frac{1}{4} \times \frac{25}{100} = 6.25\%$$

According to the greatest probability, the gesture represented by this string should be classified as a "Grasp" gesture.

Example 5.2: Now consider another input string "lf", since

$$P(r_1 \Rightarrow lf) = \frac{1}{4} \times \frac{40}{100} = 10\%$$
$$P(r_2 \Rightarrow lf) = \frac{1}{4} \times \frac{35}{100} = 8.75\%$$
$$P(r_3 \Rightarrow lf) = \frac{2}{4} \times \frac{25}{100} = 12.5\%$$

Based on the greatest probability, we can classify the gesture "lf" as a "J" gesture.

Example 5.3: If an input string is detected as "pl", which does not match any of

our defined gestures, according to the grammar:

$$P(r_1 \Rightarrow pl) = \frac{1}{4} \times \frac{40}{100} = 10\%$$
$$P(r_2 \Rightarrow pl) = \frac{0}{4} \times \frac{35}{100} = 0$$
$$P(r_3 \Rightarrow pl) = \frac{1}{4} \times \frac{25}{100} = 6.25\%$$

Based on the greatest probability, we can decide that the gesture represented by string "*pl*" is most possibly a "Grasp" gesture.

The flexibility of the SCFG allows the user to easily change the grammar so that other gestures with different combinations of detected postures or more complex gestures can be described. The assignment of the probability to each production rule can also be used to control the "wanted" gestures and the "unwanted" gestures. Greater values of probability could be assigned to the "wanted" gestures such as the "Grasp" gesture in previous examples. Smaller probability could be assigned to the "unwanted" gestures so that the resulting SCFG would generate the "wanted" gestures with higher probabilities.

Example 5.4: If we noticed the "Quote" gesture is more often performed by users and the "J" gesture is least popular, we can define P_G as:

$$r_1 : S \xrightarrow{35\%} pf,\ r_2 : S \xrightarrow{40\%} tf,\ r_3 : S \xrightarrow{25\%} lf$$

where "r_2" (corresponding to the "Quote" gesture) is assigned with the highest probability. If an input string is detected as "*tl*", since

$$P(r_1 \Rightarrow tl) = \frac{0}{4} \times \frac{35}{100} = 0$$
$$P(r_2 \Rightarrow tl) = \frac{1}{4} \times \frac{40}{100} = 10\%$$
$$P(r_3 \Rightarrow tl) = \frac{1}{4} \times \frac{25}{100} = 6.25\%$$

Based on the highest probability, the gesture represented by string "*tl*" will be recognized as a "Quote" gesture. Even though the production rule "r_3" also has a posture primitive "*t*" in its derivation, but the associated probability is only 25%, which resulted in a smaller probability to generate the string "*tl*".

5.3 Global Hand Motion Analysis

To analyze global hand motions, we define a set of direction primitives similar to the chain code shown in Figure 5.4. As illustrated in Figure 5.5, the hand motion direction is estimated by computing the slope value of hand movements according to:

$$S = \frac{dy}{dx}$$

Based on the value of the slope, we arranged the correspondent direction primitives according to the assignments shown in Figure 5.6.

For the task of hand motion analysis, we try to classify two structured gestures with different trajectory patterns. As shown in Figure 5.7, we define a counterclockwise rectangle gesture starting from the up-right corner and a counterclockwise diamond gesture

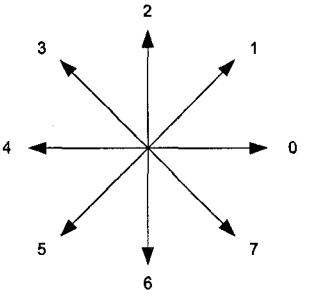


Figure 5.4: The primitives for hand motion directions.

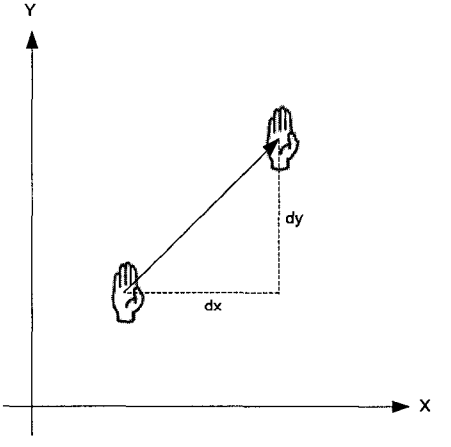


Figure 5.5: Calculate the slope value for hand motions.

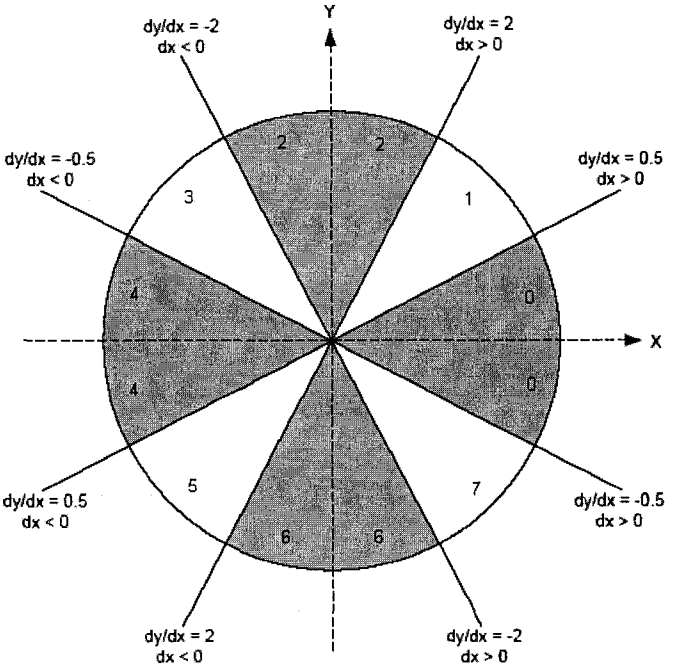
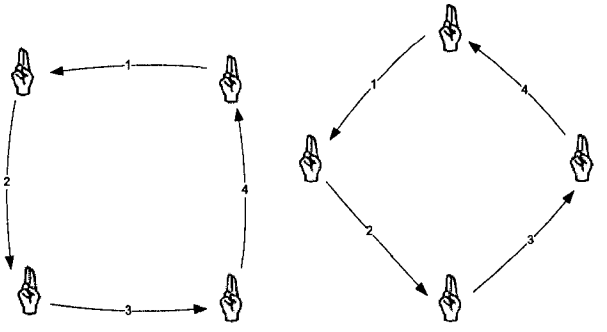


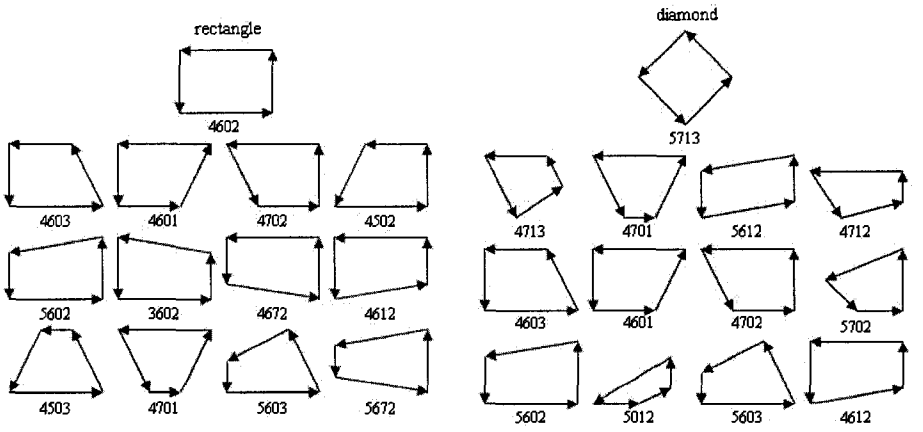
Figure 5.6: The assignment of direction primitives according to the slope values.



(a) Rectangle (b) Diamond

Figure 5.7: Two structured gestures with different trajectory patterns.

starting from the top vertex. The rectangle and the diamond can be drawn by any of the four hand postures. When moving the hand in front of the camera, it is very hard to keep the hand moving strictly along the line so that perfect shapes can be drawn. As the result of this, many noisy versions will be drawn like the distorted shapes shown in Figure 5.8. Among these noisy versions, some distorted shapes can be clustered into either the rectangle class or the diamond class, and thus cause ambiguity for our motion



(a) The rectangle's distorted versions (b) The diamond's distorted versions

Figure 5.8: The distorted versions for the two standard trajectories.

analysis.

To solve the problem, we define two SCFGs to describe the structured gestures. The SCFG defined for the rectangle gesture is:

$$G_{rectangle} = (V_{N\_\,rectangle}, V_{T\_\,rectangle}, R_{rectangle}, S)$$

where

$$V_{N\_\,rectangle} = \{S, A_1, A_2, A_3, A_4\},$$
$$V_{T\_\,rectangle} = \{t, p, l, f, 0, 1, 2, 3, 4, 5, 6, 7\}$$

The terminals p, f, t, l stand for the four postures: "palm", "fist", "two fingers" and "little finger". The numbers are the primitives for the motion directions defined in Figure 5.4. The grammar $R_{rectangle}$ is defined by:

$$\begin{array}{lll}
S \xrightarrow{25\%} tA_1, & S \xrightarrow{25\%} pA_1 & \\
S \xrightarrow{25\%} lA_1, & S \xrightarrow{25\%} fA_1 & \\
A_1 \xrightarrow{80\%} 4A_2, & A_1 \xrightarrow{10\%} 3A_2, & A_1 \xrightarrow{10\%} 5A_2 \\
A_2 \xrightarrow{80\%} 6A_3, & A_2 \xrightarrow{10\%} 5A_3, & A_2 \xrightarrow{10\%} 7A_3 \\
A_3 \xrightarrow{80\%} 0A_4, & A_3 \xrightarrow{10\%} 1A_4, & A_3 \xrightarrow{10\%} 7A_4 \\
A_4 \xrightarrow{80\%} 2, & A_4 \xrightarrow{10\%} 1, & A_4 \xrightarrow{10\%} 3
\end{array}$$

The SCFG defined for the diamond gesture is:

$$G_{diamond} = (V_{N\_\,diamond}, V_{T\_\,diamond}, R_{diamond}, S)$$

where

$$V_{N\_\,diamond} = \{S, A_1, A_2, A_3, A_4\},$$
$$V_{T\_\,diamond} = \{t, p, l, f, 0, 1, 2, 3, 4, 5, 6, 7\}$$

The grammar $R_{diamond}$ is:

$$\begin{array}{lll}
S \xrightarrow{25\%} tA_1, & S \xrightarrow{25\%} pA_1 & \\
S \xrightarrow{25\%} lA_1, & S \xrightarrow{25\%} fA_1 & \\
A_1 \xrightarrow{70\%} 5A_2, & A_1 \xrightarrow{15\%} 4A_2, & A_1 \xrightarrow{15\%} 6A_2 \\
A_2 \xrightarrow{70\%} 7A_3, & A_2 \xrightarrow{15\%} 0A_3, & A_2 \xrightarrow{15\%} 6A_3 \\
A_3 \xrightarrow{70\%} 1A_4, & A_3 \xrightarrow{15\%} 0A_4, & A_3 \xrightarrow{15\%} 2A_4 \\
A_4 \xrightarrow{70\%} 3, & A_4 \xrightarrow{15\%} 1, & A_4 \xrightarrow{15\%} 2
\end{array}$$

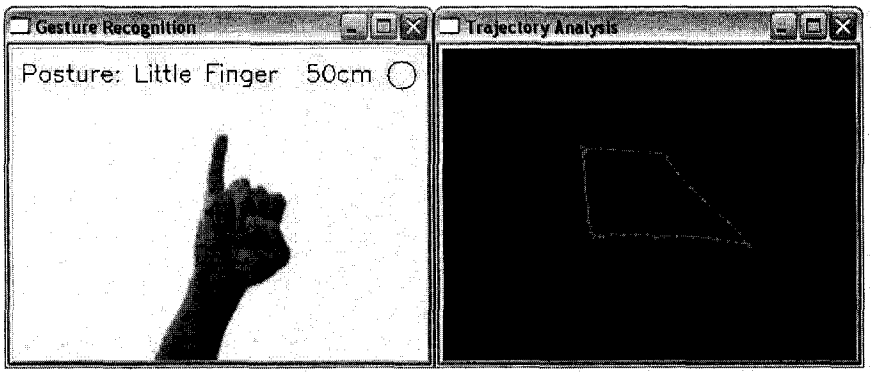
In $R_{rectangle}$ and $R_{diamond}$, there are four possibilities for the start symbol S to begin with, which means both gestures can be performed by any of the four hand postures with the same probability of 25%. The string for the standard rectangle gesture is "4602" according to the defined primitives. Considering the noisy versions, we set a probability of 80% for the first primitive to be 4, and 10% for its distorted version that can be either 3 or 5. The same probabilities are also assigned to the other three primitives and their noisy versions. The string for the standard diamond gesture is "5713". We set a probability of 70% for the standard primitives, and 15% for the distorted versions.

Example 5.5: Consider the distorted shape "4603" drawn by the "little finger" posture in Figure 5.9 (a), the parsing according to the grammar $R_{rectangle}$ is:

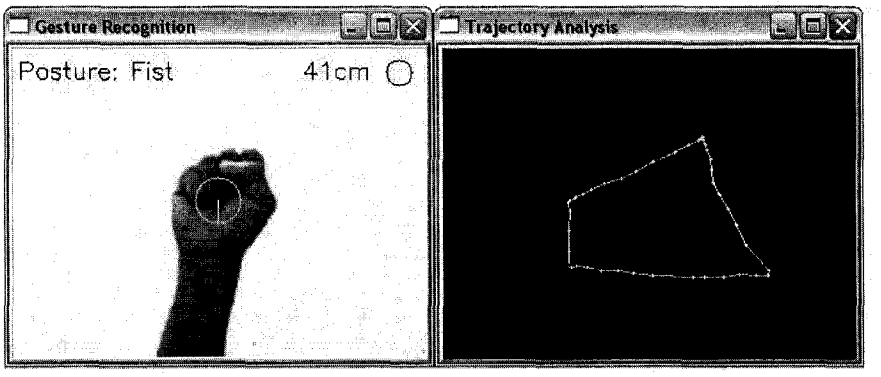
$$\begin{array}{l}
S \xrightarrow{25\%} pA_1 \\
A_1 \xrightarrow{80\%} 4A_2 \\
A_2 \xrightarrow{80\%} 6A_3 \\
A_3 \xrightarrow{80\%} 0A_4 \\
A_4 \xrightarrow{10\%} 3
\end{array}$$

Based on the parsing, the probability for this distorted shape to be generated by the rectangle gesture class is:

$$\begin{aligned}
&P(4603|G_{rectangle}) \\
&= 25\% \times 80\% \times 80\% \times 80\% \times 10\% \\
&= 1.28\%
\end{aligned}$$



(a) The distorted shape "4603".



(b) The distorted shape "5603".

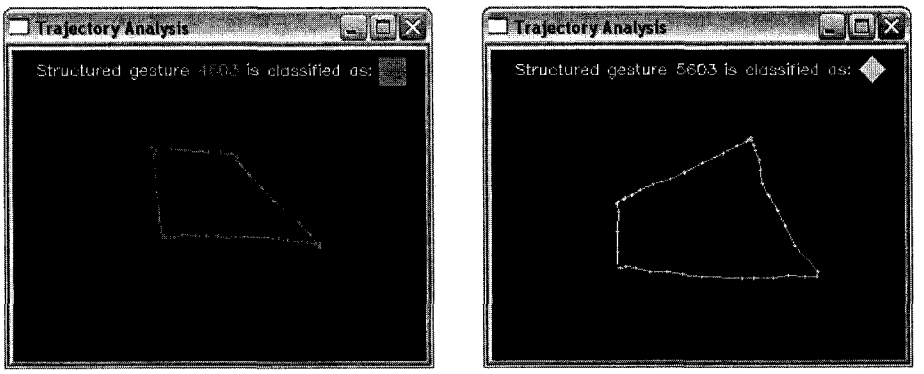
Figure 5.9: The examples of two distorted structured gestures.

It is noticed that this distorted shape can also be generated by the grammar $R_{diamond}$ with the parsing:

$$
\begin{aligned}
S &\xrightarrow{25\%} pA_1 \\
A_1 &\xrightarrow{15\%} 4A_2 \\
A_2 &\xrightarrow{15\%} 6A_3 \\
A_3 &\xrightarrow{15\%} 0A_4 \\
A_4 &\xrightarrow{70\%} 3
\end{aligned}
$$

The probability for this distorted shape to be generated by the diamond gesture class is:

$$
\begin{aligned}
&P(4603|G_{diamond}) \\
&= 25\% \times 15\% \times 15\% \times 15\% \times 70\% \\
&= 0.06\%
\end{aligned}
$$



(a) The classification result for "4603". (b) The classification result for "5603".

Figure 5.10: The classification results for the distorted trajectories.

By comparing the probabilities, since

$$P(4603|G_{rectangle}) > P(4603|G_{diamond})$$

the distorted shape represented by string "4603" should be classified as a rectangle gesture as Figure 5.10(a) shows.

Example 5.6: Now consider another distorted shape "5603" drawn by the "fist" posture in Figure 5.9 (b). The probabilities for the grammars to generate this distorted shape are:

$$\begin{aligned} &P(5603|G_{rectangle}) \\ &= 25\% \times 10\% \times 80\% \times 80\% \times 10\% \\ &= 0.16\% \end{aligned}$$

$$\begin{aligned} &P(5603|G_{diamond}) \\ &= 25\% \times 70\% \times 15\% \times 15\% \times 70\% \\ &= 0.28\% \end{aligned}$$

Since

$$P(5603|G_{diamond}) > P(5603|G_{rectangle})$$

the distorted shape represented by string "5603" should be classified as a diamond gesture as Figure 5.10(b) shows.

5.4 Summary

This chapter introduces the high-level hand gesture recognition and motion analysis based on SCFGs. The postures detected by the lower level are first converted into a sequence of terminal strings according to the grammar. Based on the similarity measurement and the probability associated with the correspondent production rule, the corresponding gesture can be identified by looking for the production rule that has the highest probability to generate this input string. For the global hand motion analysis, we use SCFGs to analyze two structured hand gestures with different trajectory patterns: the rectangle gesture and the diamond gesture. Based on the probabilities associated with the production rules, given an input string, the SCFGs can effectively disambiguate the distorted trajectory patterns and identify them correctly.

Chapter 6

Gesture-Based Interaction with a 3D Gaming Virtual Environment

6.1 3D Gaming Virtual Environment

To demonstrate the effectiveness of our gesture recognition system for human-computer interaction, we develop an application of interacting a 3D gaming virtual environment of learning objects by hand gestures. Virtual environment (VE) technology provides the simulations with immersion, interactivity and information intensity, which gives the user the sense of being mentally immersed in the environment by delivering feedback to one or more modalities. The 3D information provided by VEs offers possibilities such as perceiving more information at a time, displaying meaningful patterns in data, and understanding the relationship among different objects [94]. The ability to visualize data in 3D extends the capabilities of many applications which are previously limited to 2D views [95, 96]. For instance, the traditional 2D blueprint of the plumbing system of a city can now be visualized as 3D structures inside a virtual city, which greatly improves the experience of design and understanding.

In education, VEs offer a whole new way of learning by providing the visualizable learning objects which are entities that can be used for learning, education and training [97]. Learning objects are stored in learning object repositories, which can be searched with the metadata associated with each learning object. Traditional learning object repositories are 2D web-based. To search a learning object, users are required to formulate a simple or an advanced query by filling out an electronic form that enables the web-based repository compose boolean combinations of search criteria.

Visual metaphors such as graphs and charts are more effective and easier for people to understand and learn abstract information [98, 99]. These visual metaphors can motivate people, improve memorizing and focus the attention of the learner. To exploit the advantages brought by visual metaphors, appropriate information visualization tools need to be selected to represent the abstract data efficiently. With the immersive experience provided by VEs, learning object repositories can be mapped to a 3D gaming VE, which can provide a novel access paradigm and improves users' leaning experience significantly. Students and teachers can search, access and interact with learning objects in an interesting 3D gaming VE rather than formulating a complex search criteria and reading a long list of search results.

To facilitate the information transformation process, as illustrated in Figure 6.1, we implement a 3D gaming VE structure to search the learning object repository with hand gestures. The layout of the 3D gaming VE is shown in Figure 6.2, which allows user visualizing search results in an attractive display space. When a search query is sent by the user, as Figure 6.3 shows, the search results are displayed as different traffic signs grouped along the virtual highways according to the keywords. The visually organized representation of the information allows users to get insight into the data and interact with them directly. The user is represented by the avatar car that can navigate through the gaming VE along the virtual highways. As Figure 6.4 shows, the attributes of a search result can be displayed by selecting its traffic sign, and an attribute window will pop up with the information including the result's title, format, keyword, *etc.* If the user

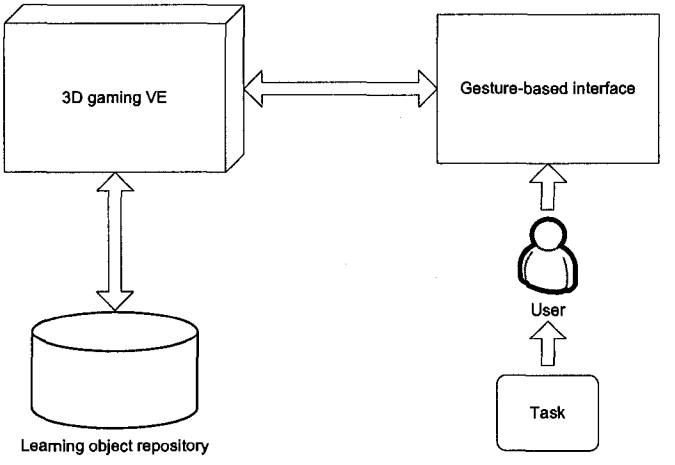


Figure 6.1: The gesture-based 3D gaming VE structure.

Figure 6.2: The 3D gaming VE layout for searching learning objects.

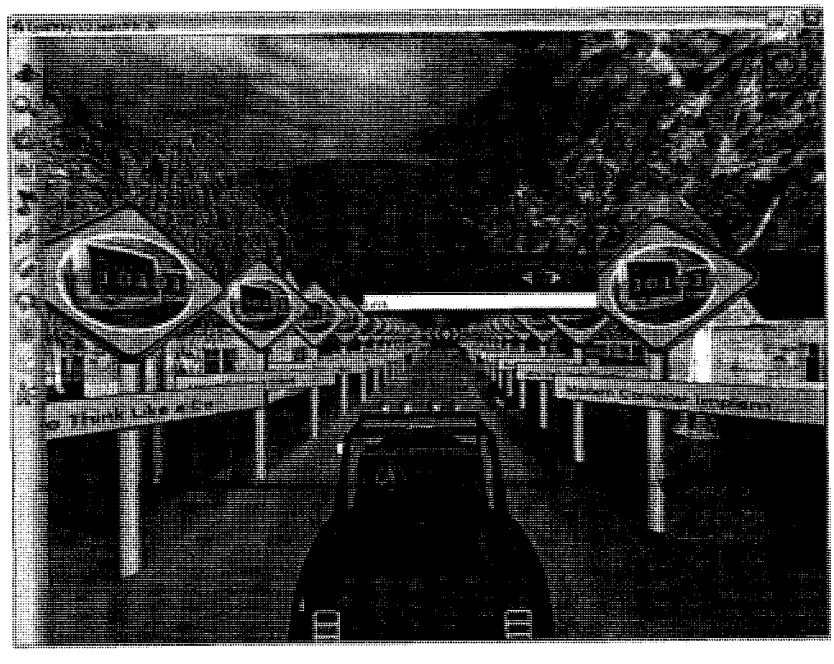


Figure 6.3: Search results are mapped to traffic signs along the virtual highway.

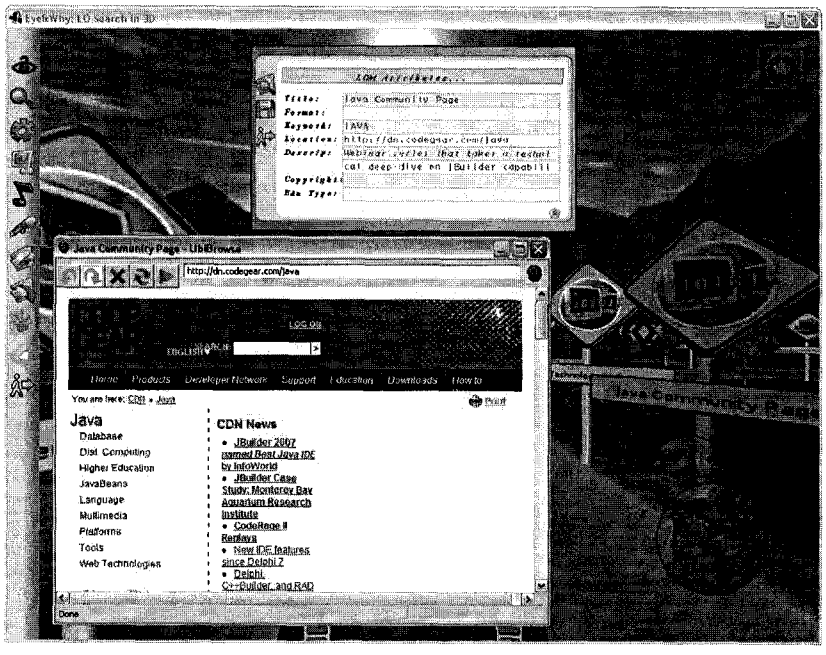


Figure 6.4: Checking the attributes and detailed information of the search result.

feels necessary to have a detailed look of the correspondent learning object, he can open a sperate explorer window that goes to the location and displays all of the information of this learning object. An overall world layout is provided to tell the user his current position. The learning experience can be enhanced by presenting the game-like avatar model so that the user can be entertained while learning.

6.2 Gesture-Based Interface

Once the 3D gaming VE has been created, to explore the virtual world and access the search results, the user, who is represented by the avatar car, need to navigate through it. A navigation and manipulation interface allows the interactive view change to the VE and exploration by selecting and manipulating the virtual object of interest [100]. A good navigation and manipulation interface to the 3D gaming VE should be natural and intuitive. The interactions with the VE and virtual objects should inherently make sense to the user. Meanwhile, the interface should also be easy to learn and yet powerful enough to allow the user to accomplish the tasks. The interactions should be precise so that the user and the VE are in agreement on what objects is being interacted with. The interface should allow direct manipulation of virtual objects so that more intuitiveness

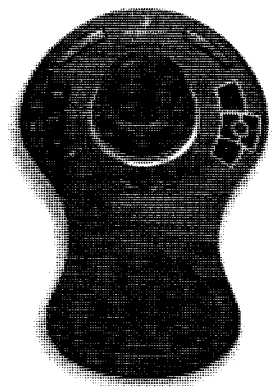


Figure 6.5: The Logitech SpaceExplorer trackball (from [101]).

and effectiveness can be brought to the user.

A typical class of interfaces that allow navigation and manipulation in 3D VEs is trackballs such as the Logitech SpaceExplorer shown in Figure 6.5 [101]. Unlike conventional mice which limit the user's movement on flat planes, with the SpaceExplorer, the user can move the cursor freely within a 3D environment without the limits imposed by the outmoded "flatworld" technology. The trackball has a sensorized cylinder that measures three forces and three torques applied by the user's hand on a compliant element. Forces and torques are measured indirectly based on the spring deformation law. These forces and torques are converted to a differential change in the controlled object position and orientation. Several buttons are integrated with the trackball, which are

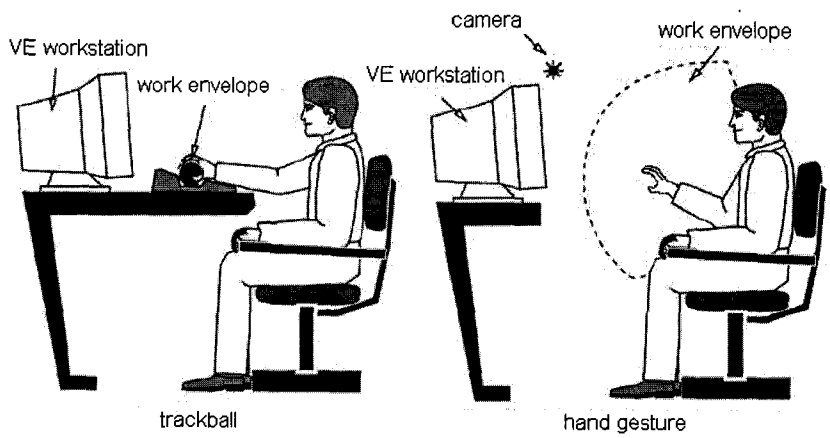


Figure 6.6: The work envelopes: trackball *vs* hand gesture (adapted from [100]).

pre-programmed with commands according to the applications.

Trackballs limit the motion freedom of the user's hand to a small work envelope close to the desk (see Figure 6.6), therefore the natural motion of the user's hand is sacrificed, and the interaction with the VE is less intuitive [100]. In order to have a larger work envelope to interact with the 3D gaming VE, it is desirable to maintain the hand freedom in terms of global hand motions and local finger motions within a certain volume. With the developed 3D hand tracking and gesture recognition system, a gesture-based navigation and manipulation interface can be applied for the 3D gaming VE to achieve a larger work space as illustrated in Figure 6.6, which can result in a more intuitive and natural user experience with greater flexibility.

6.3 Navigation with Hand Postures

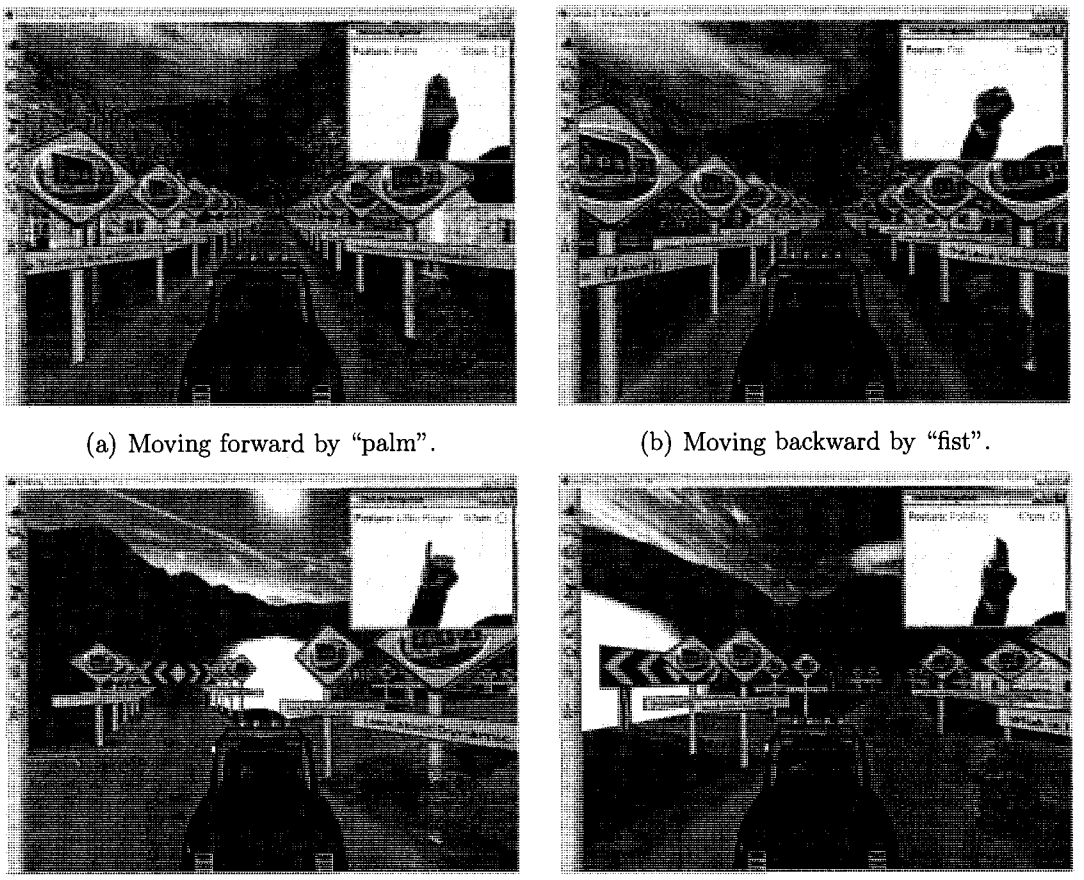
Navigation commands allow the user moving to different locations within a VE scene. In our 3D gaming VE, the user is represented by the avatar car. To move the avatar car to different locations, we use the means of manipulating the user's viewpoint within the virtual world. The avatar car can be moved forward/backward and turn left/right. To implement these functions, we map these commands to our hand postures as Table 6.1 shows. This mapping provides a simple and intuitive means of hand posture-based interface to navigate the gaming VE by driving the avatar car. Figure 6.7 shows the screen shots of navigating the car to different directions using different hand postures.

Table 6.1: Navigation commands performed by hand postures.

Move Forward	Move Backward	Turn Right	Turn Left
			
Palm	Fist	Little Finger	Two Fingers

6.4 Manipulation with Hand Gestures

Manipulation commands allow the user selecting and manipulating the virtual object he is interested in. For our 3D gaming VE, the manipulation commands including selecting a traffic sign and open an explorer window to display the correspondent information of



(a) Moving forward by "palm".

(b) Moving backward by "fist".

(c) Turning right by "little finger".

(d) Turning left by "two fingers".

Figure 6.7: Navigating the avatar car by hand postures.

the learning object. We map these manipulation commands to our hand gestures as Table 6.2 shows. To select a traffic sign, the user first drives the avatar car close to the target. As both sides of the virtual highway are aligned with traffic signs, the user needs to specify which side the targeted traffic sign is located. For example, as Figure 6.8 shows, if the user is interested in the traffic sign with a PDF mark on the right side , he just needs to perform the "J" gesture, and this traffic sign will be highlighted with an attribute window showing its attributes. Figure 6.9 shows the example of selecting the left traffic sign with the "Quote" gesture. If the user find the left traffic sign is what he is looking for according to its attributes, he can access this learning object using the "Grasp" gesture, and an explorer window will be displayed with the detailed information of this learning object (see Figure 6.10).

The system has been demonstrated to and used by different people. Most users find

Table 6.2: Manipulation commands performed by hand gestures.

Select Right Traffic Sign	Select Left Traffic Sign	Open Learning Object
		
J	Quote	Grasp

the gesture-based interface intuitive to use and can control the application by a short instruction. Feedbacks from users are positive and many users are interested and excited about being able to interact with the 3D gaming VE by just moving their hand into the work envelope and the navigation and manipulation would commence. Figure 6.11 shows one user navigating the 3D gaming VE and manipulating virtual objects with his hand.

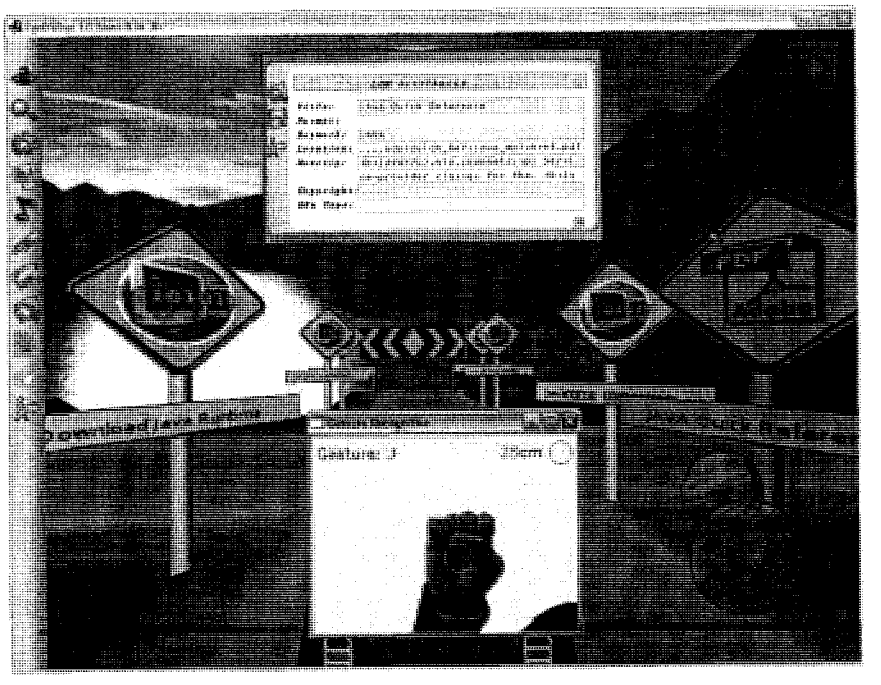


Figure 6.8: Selecting the right traffic sign with the "J" gesture.

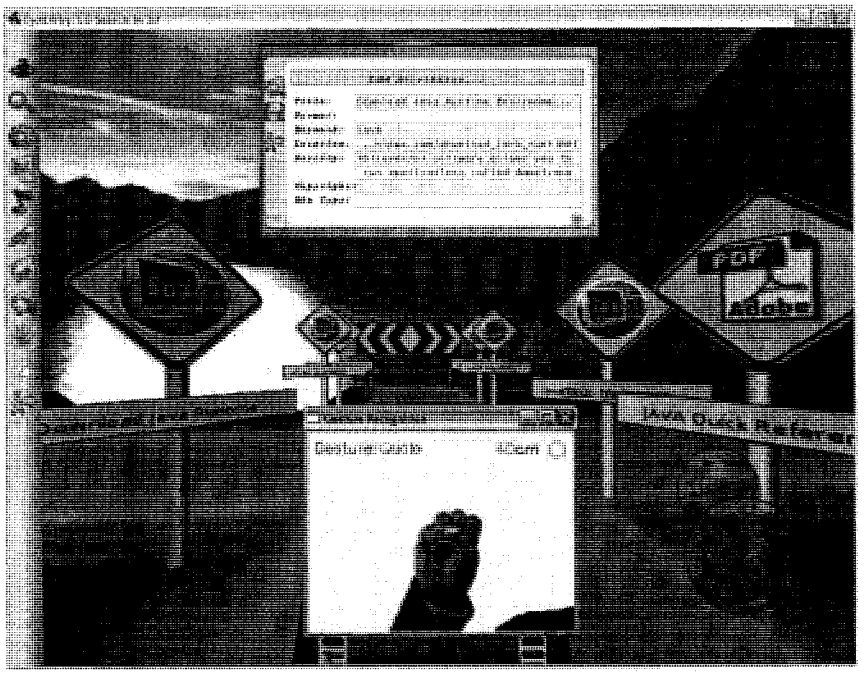


Figure 6.9: Selecting the left traffic sign with the "Quote" gesture.

Figure 6.10: Opening an explorer window with the "Grasp" gesture.

6.5 Summary

This chapter presents an application of a gesture interface for interaction with a 3D gaming VE. With this system, the user can navigate the 3D gaming world by driving the avatar car with a set of hand postures. When the user wants to manipulate the virtual objects, he can use a set of hand gestures to select the target traffic sign and open a window to check the information of the correspondent learning object. This application demonstrates the gesture-based interface can achieve an improved interaction, which are more intuitive and flexible for the user.

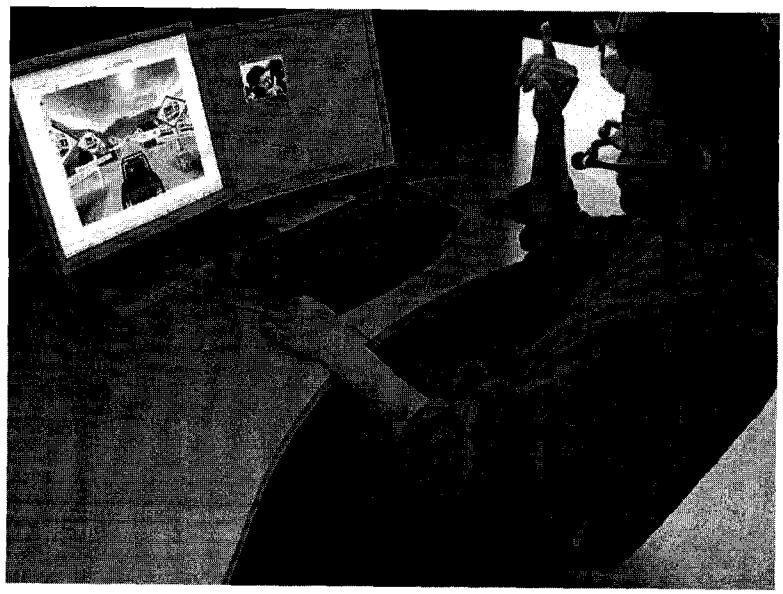


Figure 6.11: Interacting with the 3D gaming VE with hand gestures.

Chapter 7

Conclusions

This thesis presents a two-level architecture to recognize hand gestures in real-time with one camera as the input device. This architecture considers the hierarchical composite properties of hand gestures and combines the advantages of statistical and syntactic approaches to achieve real-time vision-based hand tracking and gesture recognition.

The low-level of the architecture focuses on hand posture detection and tracking with Haar-like features and the AdaBoost learning algorithm. The Haar-like features can effectively catch the appearance properties of the selected hand postures. Meanwhile, the Haar-like features also display very good robustness against different lighting conditions and a certain degree of robustness against image rotations. The AdaBoost learning algorithm can significantly speed up the performance and construct strong cascades of classifiers by combining a sequence of weak classifiers. A parallel cascades structure is implemented to classify different hand postures. The experiment results show this structure can achieve satisfactory performance for hand posture detection and tracking. The 3D position of the hand is recovered according to the camera's perspective projection. To achieve the robustness against cluttered backgrounds, background subtraction and noise removal are applied.

For the high-level hand gesture recognition and motion analysis, we use a SCFG to analyze the syntactic structure of the hand gestures based on the input strings converted from the postures detected by the low-level of the architecture. Based on the similarity measurement and the probability associated with each production rule, given an input string, the corresponding hand gesture can be identified by looking for the production rule that has the greatest probability to generate it. For the hand motion analysis, we use SCFGs to analyze two structured hand gestures with different trajectory patterns: the

rectangle gesture and the diamond gesture. Based on the probabilities associated with the production rules, the SCFGs can effectively disambiguate the distorted trajectories and identify them correctly.

The major contributions of this thesis are:

1. A two-level system architecture is implemented, which combines the advantages of statistical and syntactic pattern recognition approaches effectively, and achieves real-time, accurate and robust hand tracking and gesture recognition with one camera as the input device.

2. A parallel cascade structure for the architecture's low-level is implemented using the AdaBoost learning algorithm and a set of Haar-like features. This structure can correctly extract a set of hand postures track the hand motion in 3D in real-time.

3. The hand gestures are analyzed base on a SCFG, which defines the composite properties based on the constituent hand postures. The assignment of the probability to each production rule of the SCFG can be used to control the "wanted" gestures and the "unwanted" gestures. Smaller probability could be assigned to the "unwanted" gestures while greater value could be assigned to "wanted" gestures so that the resulting SCFG would generate the "wanted" gestures with higher probabilities.

4. For hand motion analysis, with the uncertainty of hand trajectories, the ambiguous versions can be identified by looking for the SCFG that has the higher probability to generate the input string. The motion patterns can be controlled by adjusting the probabilities associated with the production rules so that the resulting SCFG would generate the standard motion patterns with higher probabilities.

To demonstrate the effectiveness of our gesture recognition system for human-computer interaction, an application of gesture-based interaction with a 3D gaming VE is implemented. With this system, the user can navigate the 3D gaming world by driving the avatar car with a set of hand postures. When the user wants to manipulate the virtual objects, he can use a set of hand gestures to select the target traffic sign and open a window to check the information of the correspondent learning object.

For the future work, there are many possible improvements that can extend this work. First of all, more diversified hand samples from different people can be used in the training process so that the classifiers will be more user independent. The second improvement could be context-awareness for the gesture recognition system. The same

gesture performed within different contexts and environments can have different semantic meanings. For example, with the background extracted from the video, if there is a computer detected, we can say that a pointing gesture means turning on the computer in an office. However, if there is a stove detected from the background, we can be pretty much sure that the user is in a kitchen and the pointing gesture probably means turning on the stove. To analyze the context information, the syntactic approach again can be a good candidate. All of the relationships between the gesture and the context can be described by defining a grammar for the primitives (*e.g.* pointing gesture, computer, stove *etc.*).

Another possible improvement is to track and recognize multiple objects such as human faces, eye gaze and hand gestures at the same time. With this multimodel-based tracking and recognition strategy, the relationships and interactions among these tracked objects can be defined and assigned with different semantic meanings so that a richer command set can be covered. By integrating this richer command set with other communication modalities such as speech recognition and haptic feedback, the human-computer interaction experience can be enriched greatly and be much more interesting.

The system developed in this work can be extended to many other research topics in the field of computer vision. We hope this research could trigger more investigations to make computers see and think better.

Bibliography

[1] M. Turk, *Handbook of Virtual Environment Technology.* Lawrence Erlbaum Associates, Inc., 2001.

[2] V. Pavlovic, R. Sharma, and T. Huang, "Gestural interface to a visual computing environment for molecular biologists," in *Proc. IEEE Second International Conference on Atuomatic Face and Gesture Recognition*, 1996, pp. 30–35.

[3] T. Baudel and M. Beaudouin-Lafon, "Charade: remote control of objects using free-hand gestures," *Communications of the ACM*, vol. 36, no. 7, pp. 28–35, 1993.

[4] A. Wexelblatt, "An approach to natural gesture in virtual environments," *ACM Trans. on Computer-Human Interaction*, vol. 2, no. 3, pp. 179–200, 1995.

[5] T. Kirishima, K. Sato, and K. Chihara, "Real-time gesture recognition by learning and selective control of visual interest points," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 351–364, 2005.

[6] Y. Wu and T. S. Huang, "Hand modeling analysis and recgonition for vision-based human computer interaction," *IEEE Signal Processing Magazine, Special Issue on Immersive Interactive Technology*, vol. 18, no. 3, pp. 51–60, 2001.

[7] "Immersion Corporation, Cyberglove II Datasheet." [Online]. Available: http://www.immersion.com/3d/docs/cybergloveII_dec07v4-lr.pdf

[8] Q. Chen, A. El-Sawah, C. Joslin, and N. D. Georganas, "A dynamic gesture interface for virtual environments based on Hidden Markov Models," in *Proc. IEEE International Workshop on Haptic, Audio and Visual Environments and their Applications*, 2005, pp. 110–115.

[9] T. Metais and N. D. Georganas, "A glove gesture interface," in *Proc. Bienneal Symposium on Communication*, 2004.

[10] J. Yang, Y. Xu, and C. S. Chen, "Gesture interface: Modeling and learning," in *Proc. IEEE International Conference on Robotics and Automation*, vol. 2, 1994, pp. 1747–1752.

[11] D. Geer, "Will gesture-recognition technology point the way," *IEEE Computer*, pp. 20–23, 2004.

[12] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti, "Surround-screen projection-based virtual reality: The design and implementation of the CAVE," in *Proc. ACM SIGGRAPH*, 1993, pp. 135–142.

[13] V. Pavlovic, R. Sharma, and T. Huang, "Gestural interface to a visual computing environment for molecular biologists," in *Proc. Second International Conference on Atuomatic Face and Gesture Recognition*, 1996, pp. 30–35.

[14] S. Shafer, J. Krumm, B. Brumitt, B. Meyers, M. Czerwinski, and D. Robbins, "The new easyliving project at Microsoft Research," in *Proc. Joint DARPA/NIST Smart Spaces Workshop*, 1998.

[15] C. Joslin, A. El-Sawah, Q. Chen, and N. D. Georganas, "Dynamic gesture recognition," in *Proc. IEEE Instrumentation and Measurement Technology Conference*, 2005, pp. 1706–1711.

[16] C. Keskin, A. Erkan, and L. Akarun, "Real time hand tracking and gesture recognition for interactive interfaces using HMM," in *Proc. ICANN/ICONIP*, 2003.

[17] H. Zhou and T. S. Huang, "Tracking articulated hand motion with Eigen dynamics analysis," in *Proc. of International Conference on Computer Vision*, vol. 2, 2003, pp. 1102–1109.

[18] J. Napier, *Hands.* New York: Pantheon Books, 1980.

[19] A. Corradini, "Real-time gesture recognition by means of hybrid recognizers," *Lecture Notes in Computer Science, Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction*, vol. 2298, pp. 34–46, 2001.

[20] R. H. Liang and M. Ouhyoung, "A real-time continuous gesture recognition system for sign language," in *Proc. 3rd International Conference on Automatic Face and Gesture Recognition*, 1998, pp. 558–565.

[21] J. Lin, Y. Wu, and T. S. Huang, "Modeling the constraints of human hand motion," in *Proc. IEEE Workshop on Human Motion*, 2000, pp. 121–126.

[22] "American Sign Language online dictionary." [Online]. Available: http://www.lifeprint.com/

[23] S. Lenman, L. Bretzner, and B. Thuresson, "Using marking menus to develop command sets for computer vision based hand gesture interfaces," in *Proc. of the Second Nordic Conference on Human-Computer Interaction*, 2002, pp. 239–242.

[24] A. L. C. Barczak, F. Dadgostar, and M. Johnson, "Real-time hand tracking using the Viola and Jones method," in *Proc. International Conference on Image and Signal Processing*, 2005, pp. 336–441.

[25] H. Zhou and T. Huang, "Okapi-Chamfer matching for articulate object recognition," in *Proc. International Conference on Computer Vision*, 2005, pp. 1026–1033.

[26] K. G. Derpanis, R. P. Wildes, and J. K. Tsotsos, "Hand gesture recognition within a linguistics-based framework," in *Proc. European Conference on Computer Vision*, 2004, pp. 282–296.

[27] J. Y. Lin, Y. Wu, and T. S. Huang, "3D model-based hand tracking using stochastic direct search method," in *Proc. IEEE International Conference on Automatic Face and Gesture Recognition*, 2004, pp. 693–698.

[28] G. Ye, J. J. Corso, and G. D. Hager, "Gesture recognition using 3d appearance and motion features," in *Proc. CVPR Workshop on Real-Time Vision for Human-Computer Interaction*, 2004, pp. 160–166.

[29] M. Kölsch and M. Turk, "Fast 2D hand tracking with flocks of features and multi-cue integration," in *Proc. Conference on Computer Vision and Pattern Recognition Workshop*, vol. 10, 2004, pp. 158–165.

[30] R. Bowden, D. Windridge, T. Kadir, A. Zisserman, and M. Brady, "A linguistic feature vector for the visual interpretation of sign language," in *Proc. European Conference on Computer Vision*, 2004, pp. 391–401.

[31] C. Tomasi, S. Petrov, and A. Sastry, "3D tracking = classification + interpolation," in *Proc. International Conference on Computer Vision*, 2003, pp. 1441–1448.

[32] Y. Wu and T. S. Huang, "Non-stationary color tracking for vision-based human computer interaction," *IEEE Trans. on Neural Networks*, vol. 13, no. 4, pp. 948–960, 2002.

[33] C. Manresa, J. Varona, R. Mas, and F. J. Perales, "Real time hand tracking and gesture recognition for human-computer interaction," *Electronic Letters on Computer Vision and Image Analysis*, pp. 1–7, 2000.

[34] S. Mckenna and K. Morrison, "A comparison of skin history and trajectory-based representation schemes for the recognition of user- specific gestures," *Pattern Recognition*, vol. 37, pp. 999–1009, 2004.

[35] J. Yao and J. R. Cooperstock, "Arm gesture detection in a classroom environment," in *Proc. IEEE Workshop on Applications of Computer Vision*, 2002, pp. 153–157.

[36] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering," in *Proc. 5th IEEE International Conference on Automatic Face and Gesture Recognition*, 2002, pp. 405–410.

[37] S. Jung, Y. Ho-Sub, W. Min, and B. W. Min, "Locating hands in complex images using color analysis," in *Proc. IEEE International Conference on Systems, Man, and Cybernetics*, vol. 3, 1997, pp. 2142–2146.

[38] K. Imagawa, H. Matsuo, R. Taniguchi, D. Arita, S. Lu, and S. Igi, "Recognition of local features for camera-based sign language recognition system," in *Proc. International Conference on Pattern Recognition*, vol. 4, 2000, pp. 849–853.

[39] Y. Cui and J. Weng, "Appearance-based hand sign recognition from intensity image sequences," *Computer Vision Image Understanding*, vol. 78, no. 2, pp. 157–176, 2000.

[40] A. Ramamoorthy, N. Vaswani, S. Chaudhury, and S. Banerjee, "Recognition of dynamic hand gestures," *Pattern Recognition*, vol. 36, pp. 2069–2081, 2003.

[41] E. Ong and R. Bowden, "Detection and segmentation of hand shapes using boosted classifiers," in *Proc. IEEE 6th International Conference on Automatic Face and Gesture Recognition*, 2004, pp. 889–894.

[42] F. Chen, C. Fu, and C. Huang, "Hand gesture recognition using a real-time tracking method and Hidden Markov Models," *Image and Vision Computing*, vol. 21, no. 8, pp. 745–758, 2003.

[43] C. W. Ng and S. Ranganath, "Gesture recognition via pose classification," in *Proc. 15th International Conference on Pattern Recognition*, vol. 3, 2000, pp. 699–704.

[44] K. Oka, Y. Sato, and H. Koike, "Real-time fingertip tracking and gesture recognition," in *Proc. IEEE Computer Graphics and Applications*, vol. 22, no. 6, 2002, pp. 64–71.

[45] Z. Zhang, Y. Wu, Y. Shan, and S. Shafer, "Visual panel: Virtual mouse keyboard and 3D controller with an ordinary piece of paper," in *Proc. Workshop on Perceptive User Interfaces*, 2001.

[46] S. Malik, C. McDonald, and G. Roth, "Hand tracking for interactive pattern-based augmented reality," in *Proc. International Symposium on Mixed and Augmented Reality*, 2002.

[47] C. Huang and S. Jeng, "A model-based hand gesture recognition system," *Machine Vision and Application*, vol. 12, no. 5, pp. 243–258, 2001.

[48] R. Cutler and M. Turk, "View-based interpretation of real-time optical flow for gesture recognition," in *Proc. Third IEEE Conference on Face and Gesture Recognition*, 1998.

[49] S. Lu, D. Metaxas, D. Samaras, and J. Oliensis, "Using multiple cues for hand tracking and model refinement," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2003, pp. 443–450.

[50] Q. Chen, E. M. Petriu, and X. Yang, "A comparative study of fourier descriptors and Hu's seven moments for image recognition," in *Proc. IEEE Canadian Conference on Electrical and Computer Engineering*, vol. 1, 2004, pp. 103–106.

[51] B. Horn and B. Schunk, "Determining optical flow," *Artifical Intelligence*, vol. 17, pp. 185–204, 1981.

[52] Q. Yuan, S. Sclaroff, and V. Athitsos, "Automatic 2D hand tracking in video sequences," in *Proc. IEEE Workshops on Application of Computer Vision*, 2005, pp. 250–256.

[53] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Computer Vision and Pattern Recognition*, vol. 2, no. 2, 2000, pp. 142–149.

[54] G. Bradski, "Real time face and object tracking as a component of a perceptual user interface," in *Proc. IEEE Workshop on Applications of Computer Vision*, 1998, pp. 214–219.

[55] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[56] H. Zhou, D. J. Lin, and T. S. Huang, "Static hand gesture recognition based on local orientation histogram feature distribution model," in *Proc. Conference on Computer Vision and Pattern Recognition Workshop*, vol. 10, 2004, pp. 161–169.

[57] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2001, pp. 511–518.

[58] ——, "Robust real-time object detection," *Cambridge Research Laboratory Technical Report Series CRL2001/01*, pp. 1–24, 2001.

[59] M. Kölsch and M. Turk, "Robust hand detection," in *Proc. 6th IEEE Conference on Automatic Face and Gesture Recognition*, 2004.

[60] ——, "Analysis of rotational robustness of hand detection with a Viola-Jones detector," in *Proc. International Conference on Pattern Recognition*, vol. 3, 2004, pp. 107–110.

[61] M. Kölsch, M. Turk, and T. Höllerer, "HandVu vision-based hand gesture interface." [Online]. Available: http://ilab.cs.ucsb.edu/projects/mathias/handvu_ilab.html

[62] M. Kölsch, "Vision based hand gesture interfaces for wearable computing and virtual environments," Ph.D. dissertation, University of California, Santa Barbara, 2004. [Online]. Available: http://www.movesinstitute.org/~kolsch/publications.html

[63] Y. Wu, J. Lin, and T. S. Huang, "Analyzing and capturing articulated hand motion in image sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, pp. 1910–1922, 2005.

[64] V. Athitsos and S. Sclaroff, "Estimating 3D hand pose from a cluttered image," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2003.

[65] A. Imai, N. Shimada, and Y. Shirai, "3-D hand posture recognition by training contour variation," in *Proc. 6th IEEE International Conference on Automatic Face and Gesture Recognition*, 2004, pp. 895–900.

[66] K. S. Fu and A. Rosenfeld, "Pattern recognition and image processing," *IEEE Computer*, vol. 17, no. 10, pp. 274–282, 1984.

[67] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[68] P. N. Belhumeur, J. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[69] R. E. Schapire, "The boosting approach to machine learning: An overview," *Nonlinear Estimation and Classification, Springer*, 2003.

[70] K. S. Fu, *Syntactic Pattern Recognition and Applications.* New Jersey: Prentice-Hall, 1982.

[71] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision.* PWS Publishing, 1999.

[72] N. Chomsky, *Syntactic Structures.* The Hague: Mouton, 1966.

[73] D. Grune and C. J. H. Jacobs, *Parsing Techniques: A Practical Guide.* Ellis Horwood, 1991.

[74] A. C. Shaw, "A formal picture description scheme as a basis for picture processing systems," *Information and Control*, vol. 14, no. 1, pp. 9–52, 1969.

[75] C. Hand, I. Sexton, and M. Mullan, "A linguistic approach to the recognition of hand gestures," in *IEE Ergonomics Society Proc. Designing Future Interaction Conference*, 1994.

[76] M. Jones, R. Doyle, and P. ONeill, "The gesture interface module," *Technical Report, GLAD-IN-ART Deliverable 3.3.2, Trinity College, Dublin, Ireland*, 1993.

[77] Y. Ivanov and A. Bobick, "Recognition of visual activities and interactions by stochastic parsing," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 852–872, 2000.

[78] M. S. Ryoo and J. K. Aggarwal, "Recognition of composite human activities through context-free grammar based representation," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1709–1718.

[79] D. Moore and I. Essa, "Recognizing multitasked activities using stochastic context-free grammar," in *Proc. IEEE CVPR Workshop on Models vs Exemplars*, 2001.

[80] D. Minnen, I. Essa, and T. Starner, "Expectation grammars: Leveraging high-level expectations for activity recognition," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. 626–632.

[81] M. Yamamoto, H. Mitomi, F. Fujiwara, and T. Sato, "Bayesian classification of task-oriented actions based on stochastic context-free grammar," in *Proc. 7th International Conference on Automatic Face and Gesture Recognition*, 2006.

[82] D. D. Morris and J. M. Rehg, "Singularity analysis for articulated object tracking," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1998, pp. 289–296.

[83] K. S. Fu, "A step towards unification of syntactic and statistical pattern recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 8, no. 3, pp. 398–404, 1986.

[84] "Chinese Sign Language." [Online]. Available: http://www.cndeaf.com/sign/china/china_index.htm

[85] F. Quek, "Toward a vision-based hand gesture interface," in *Proc. 7th Virtual Reality Software and Technology Conference*, 1994, pp. 17–31.

[86] G. Bradski, A. Kaehler, and V. Pisarersky, "Learning-based computer vision with Intel's open source computer vision library," *Intel Technology Journal*, vol. 9, no. 2, pp. 119–130, 2005.

[87] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in *Proc. IEEE International Conference on Image Processing*, vol. 1, 2002, pp. 900–903.

[88] Y. Freund and R. E. Schapire, "A short introduction to boosting," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771–780, 1999.

[89] ——, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[90] "Open Source Computer Vision Library." [Online]. Available: http://www.intel.com/technology/computing/opencv/

[91] A. Kuranov, R. Lienhart, and V. Pisarevsky, "An empirical analysis of boosting algorithms for rapid objects with an extended set of haar-like features," *Intel Technical Report MRL-TR-July-02-01*, 2002.

[92] "Matlab Image Processing Toolbox." [Online]. Available: http://www.mathworks.com/access/helpdesk/help/toolbox/images/

[93] A. Stolcke, "Bayesian learning of probabilistic language models," Ph.D. dissertation, University of California, Berkeley, 1994.

[94] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in Information Visualization: Using Vision to Think.* New York: Morgan Kaufmann Publishers, 1999.

[95] G. Wesche, J. Wind, and M. Göbel, "Visualization on the responsive workbench," *IEEE Computer Graphics and Applications*, vol. 17, no. 4, pp. 10–12, 1997.

[96] M. Schulz, T. Rending, and T. Ertl, "Analysing engineering simulations in a virtual environment," *IEEE Computer Graphics and Applications*, vol. 18, no. 6, pp. 46–52, 1998.

[97] J. Klerkx, E. Duval, and M. Meire, "Visualization for accessing learning object repositories," in *Proc. 8th International Conference on Information Visualization*, vol. 14, no. 5, 2004, pp. 465–470.

[98] M. Bauer and P. Johnson-Laird, "How diagrams can improve reasoning," *Psychological Science*, vol. 4, no. 6, pp. 372–378, 1993.

[99] J. Larkin and H. Simon, "Why a diagram is (sometimes) worth ten thousand words," *Cognitive Science*, vol. 11, no. 1, pp. 65–99, 1987.

[100] G. C. Burdea and P. Coiffet, *Virtual Reality Technology.* New Jersey: John Wiley & Sons, 2003.

[101] "Logitech SpaceExplorer Datasheet." [Online]. Available: http://www.3dconnexion.com/docs/SpaceExplorer_Datasheet.pdf