

# **A State-Based Approach to the Gesture Recognition**

*Hyungu Yim*

The Graduate School  
School of Electrical and Electronic Engineering  
Yonsei University

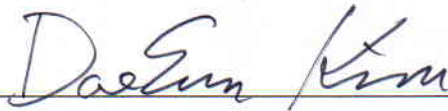
# **A State-Based Approach to the Gesture Recognition**

A Masters Thesis  
Submitted to the Department of  
Electrical and Electronic Engineering  
and the Graduate School of Yonsei University  
in partial fulfillment of the  
requirements for the degree of  
Master of engineering

*Hyungu Yim*

December 2013

This certifies that the master's thesis  
of Hyungu Yim is approved.



---

Thesis Supervisor: DaeEun Kim



---

Hong-Goo Kang



---

Euntai Kim

The Graduate School  
School of Electrical and Electronic Engineering  
Yonsei University  
December 2013

# **Abstract**

When it comes to communication between either humans or animals, the use of gestures plays the important role of expressing thoughts without sound. Gestures take many forms, such as sign languages, traffic signals, military tactical communication. In these days, direct interaction and communication with computers, referred to as Human-Computer Interaction (HCI), have become more important than ever. Gesture recognition through computers has become a big part of HCI. Inspired from these usages and importance, real-time systems for hand shape and gesture recognition are proposed in this thesis.

For hand shape recognition, a method and system that can recognize and classify the various hand shape images by means of a basic web-camera is proposed. Expensive devices, such as gloves or marking have not been used. In the case of hand gesture recognition, this dissertation proposed a state based approach. Many methods have been developed to recognize hand gestures by numerous research groups. The present study concentrates on the method that is based on the Finite State Machine (FSM) approach with probability, and compares the performances with a Hidden Markov Model (HMM) approach. The experiments have been performed with various sorts of gestures as the inputs, using either a mouse or a camera to confirm the performance of the proposed recognizer. The performance of proposed recognizer will also be compared with the other recognizers.



# Acknowledgements

연구실에 인턴으로 들어와서 공부한지가 얼마 되지 않은 것 같은데, 벌써 2년 6개월이란 시간이 지나 졸업을 맞이하게 되었습니다. 그 동안의 연구의 결실로 나온 학위 논문이 완성되기까지 항상 도움을 주시고 기도를 해주셨던 분들에게 이 글을 빌어 감사의 인사를 전하고자 합니다.

지난 2년 동안 연구 뿐만이 아니라 삶에 대해서도 아낌없는 조언으로 지도해 주신 김대은 교수님께 깊은 감사를 드립니다. 교수님의 지도를 통해 연구와 삶에 대한 저의 자세가 좀 더 성숙하게 변화되었음을 느낍니다. 그리고 더 좋은 논문이 될 수 있도록 심사과정에서 많은 부분을 지적해주시고 가르쳐주신 강홍구 교수님과 김은태 교수님께도 깊은 감사를 드립니다.

또한 저와 연구실 생활을 함께 한 선배, 친구, 후배님에게도 감사의 말을 전합니다. 삶에 대해 여러 가지 조언을 해주신 원기형, 오랜 시간 함께하며 시간을 보낸 친구 같은 동생 창민이, 옆에서 많은 도움을 준 은석이, 부족한 저를 잘 따라준 승민이, 그리고 지금은 졸업을 했지만 짧은 시간이나마 함께하였던 상욱이형, 영서형, 세준이형, 재홍이형, 동현, 중보, 서현이에게도 감사의 말을 전합니다. 이런 많은 친구들이 있었기에 연구실에서의 생활을 즐겁게 할 수 있었습니다.

또한, 언제나 저를 응원해주시고 기도로 도와주시는 부모님, 장인어른, 장모님, 상구와 민성이 에게도 몇 마디 말로는 부족하지만 깊은 감사를 드립니다.

마지막으로, 대학원 생활을 하는 동안 저보다도 많이 힘들었지만, 늘 기도를 해주었던 사랑하는 아내 민혜와 우리 아들 규민이, 딸 규원이 에게도 이 자리를 빌어 감사와 사랑을 전합니다.

## **Declaration**

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

*(Hyungu Yim)*

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Hand shape and gesture recognition . . . . .	1
1.2	Motivation and objectives . . . . .	2
1.3	Organization of dissertation . . . . .	4
<b>2</b>	<b>Background</b>	<b>6</b>
2.1	The concept of hand gesture recognition . . . . .	6
2.2	Techniques for extracting the hand gesture . . . . .	9
2.2.1	Contact based techniques . . . . .	9
2.2.2	Vision-based techniques . . . . .	10
2.3	Vision based hand gesture models . . . . .	11
2.3.1	Model-based approaches . . . . .	11
2.3.2	Appearance-based approaches . . . . .	12
2.4	Vision based hand gesture recognition techniques . . . . .	13
2.4.1	Detection . . . . .	13
2.4.2	Tracking . . . . .	16
2.4.3	Recognition . . . . .	17
2.5	Application of Hand Gesture Recognition . . . . .	24
2.6	Summary of Chapter2 . . . . .	25
<b>3</b>	<b>Hand Shape Recognition</b>	<b>26</b>
3.1	Methods . . . . .	27
3.1.1	Experiment environment . . . . .	28
3.1.2	Hand information extraction . . . . .	28
3.1.3	Hand edge extraction . . . . .	30
3.1.4	Reference data collection . . . . .	36
3.1.5	Classification . . . . .	37

3.2	Experiment . . . . .	40
3.2.1	Using one reference set of a participator . . . . .	40
3.2.2	Using large reference set of a participant . . . . .	44
3.2.3	Using several reference sets of several participant . . . . .	46
3.2.4	Estimation of orientation angle . . . . .	50
3.3	Discussion . . . . .	50
3.4	Summary of Chapter 3 . . . . .	52
<b>4</b>	<b>State based approach to hand gesture recognition</b>	<b>53</b>
4.1	Gesture extraction . . . . .	54
4.1.1	Experiment Environment . . . . .	54
4.1.2	Mouse Gesture . . . . .	55
4.1.3	Hand Gesture . . . . .	57
4.2	Finite State Machines for gesture recognition . . . . .	59
4.2.1	Training phase . . . . .	60
4.2.2	FSMs approach recognizer . . . . .	68
4.3	Performance experiments . . . . .	74
4.3.1	Parameters for HMM . . . . .	75
4.3.2	According to Number of States . . . . .	77
4.4	Summary of Chapter 4 . . . . .	81
<b>5</b>	<b>Robustness analysis</b>	<b>82</b>
5.1	Overlap problem . . . . .	83
5.2	Different time period problem . . . . .	92
5.3	Combined method for continuous recognition . . . . .	96
5.4	Comparing with proposed method and HMMs . . . . .	100
5.5	Summary of Chapter 5 . . . . .	101
<b>6</b>	<b>Conclusions</b>	<b>102</b>
6.1	Hand posture recognition . . . . .	102
6.2	Hand gesture recognition . . . . .	103
6.3	Future studies . . . . .	105
	<b>Appendix</b>	<b>107</b>
	A. Contour Distance ( $d_{threshold}$ ) Setting Method . . . . .	107
	<b>Bibliography</b>	<b>110</b>

# List of Figures

2.1	Vision-based gesture interpretation system (Reprinted from Pavlovic et al. (1997)) . . . . .	7
2.2	Body parts identified in the literature employed for making gestures (Reprinted from Karam (2006)) . . . . .	8
2.3	Whole system of Charade (Reprinted from Baudel and Beaudouin-Lafon (1993)) . . . . .	10
2.4	Examples of (a) 3D textured volumetric (b) skeleton model (Reprinted from Pavlovic et al. (1997)) . . . . .	12
2.5	Examples of (a) Binary silhouette model (b) Contour model (Reprinted from Pavlovic et al. (1997)) . . . . .	13
2.6	Vision based hand gesture recognition techniques (Reprinted from Rautaray and Agrawal (2012)) . . . . .	14
2.7	Label sequence corresponding to part of the data (Reprinted from Hong et al. (2000b)) . . . . .	23
3.1	The structure of the experiment equipment. . . . .	28
3.2	Principle of noise removal using moving window. The window which is represented the square with 9 empty space means the moving window with size 3. The red circle means the tested position. Then type 1 error is changed to be detected one which is represented by deep blue one and type 2 error is to be removed one which one is light blue. . .	30
3.3	Example of hand pixel extraction. (a) is the original image. (b) is the skin detected binary image by using RGB color value. (c) is the image after noise removal. . . . .	30
3.4	Example of edge detection by Canny Edge Filter. (a) is the original image and (b) is the edge image. The black line means the pixels on the edge. . . . .	31

3.5	Principle of Finger Removal using another moving window. First is the example image and the second is skin detected binary image. Then palm or wrist part are collected using moving window and the last one is finger-removed image which is sum of pixels which is not finger parts.	32
3.6	Examples of finger-removed images. There are four set of image. Each set has two images. The left one is the skin detected image and the right one is the finger-removed image. . . . .	32
3.7	Principle of palm extraction. R is the double of the mean value of temporary distribution which is calculated in 'Finger Part Removal' section. Red dotted line means the path of moving window and the blue translucent circle means the window. The blue x means the center and the blue arrow represent the radius. The right one shows the output of the algorithm. The yellow circle means the window size and the yellow box in the center means the center point of the palm. The other four yellow boxes in the border of the circle means the end points of window. . . . .	33
3.8	Examples of palm and center detected images. There are 7 example images with center. Yellow circle means palm position and the center of the circle is the calculated center of the hand. . . . .	34
3.9	Principle of Hand Contour Vector Set Extraction. The left one include Center of the hand, Hand Contour Vectors and the edge of the hand. The right one show the distribution of the distance between center and edge which is equal to the length of HCVs. . . . .	35
3.10	Examples of the median filter . . . . .	35
3.11	Example images of 25 classes. Images from 1st to 24th are the alphabets except 'J' and 'Z' which has title label on each image . 25th image means 'Love'. . . . .	36
3.12	Principle of Hand Contour Vector Set Matching. . . . .	38
3.13	Overview of whole process . . . . .	39
3.14	Test with same images which is used for reference set. X axis shows the class 1 25 and Y axis shows the success ratio from 0 to 1. . . . .	41
3.15	Test with various different input images of a participator who made a reference set. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together. . . . .	41
3.16	The perception ratio for each input letters . . . . .	42

3.17	The perception ratio for A, E, M, N, S, and T. The precision in last column shows the performances with grouping. . . . .	43
3.18	Test with various different input images of participators who didn't make a reference set. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together. . .	43
3.19	The perception ratio for each input letters . . . . .	44
3.20	The perception ratio for A, E, M, N, S, and T. The precision in last column shows the performances with grouping. . . . .	44
3.21	Test with various different input images of a participator who made reference sets and large reference set which is consisted of 5 images for each class. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together. . . . .	45
3.22	The perception ratio for each input letters . . . . .	45
3.23	The perception ratio for A, E, M, N, S, and T. The precision in last column shows the performances with grouping. . . . .	45
3.24	Test with various different input images of participators who made reference sets. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together. . . . .	46
3.25	Test with various different input images of participators who didn't make reference sets. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together. Mean success rate = 81.12 %(before grouping), Mean success rate = 94.45 %(after grouping) . . . . .	47
3.26	The perception ratio for each input letters . . . . .	47
3.27	The perception ratio for group 1 (A, E, M, N, S, and T), group 2 (R and U), group 3 ( G and H) and group 4 (K and V) . The precision in last column shows the performances with grouping. . . . .	48
3.28	Outputs of high performance participators. There are two participators(two rows). Left ones (a and c) are the output without grouping and the right ones (b and d) are with grouping. . . . .	48
3.29	Outputs of low performance participators. There are two participators(two rows). Left ones (a and c) are the output without grouping and the right ones(b and d) are with grouping. . . . .	49

3.30	Angle estimation experiment. X-axis is target angle and the estimated output in Y-axis. Blue line shows the ideal case and the red line shows the output. . . . .	51
4.1	The view of experiments . . . . .	55
4.2	The environment for the mouse gesture. (a) The drawing environment (b) The result after drawing the desired gesture . . . . .	56
4.3	Same gestures, (a) and (b) are drawn, but, since these two gestures are located in the different place, they are considered as different gestures even if they have same shape. Therefore, normalization is always necessary. . . . .	56
4.4	After normalization, the data has same maximum and minimum values. (a) a case of two (b) a case of six . . . . .	57
4.5	When hand is extracted by the proposed method, it doesn't matter that cuff may appear or not. . . . .	58
4.6	The process to extract and save the hand information from the video. (a) Detecting the centroid of hand. Red point. (b) The trajectory of gesture in the video. (c) The saved data from the video (d) normalized data for the experiments . . . . .	58
4.7	Trajectories of hand gestures obtained by using single camera . . . . .	60
4.8	The gestures of 'a'. (a) Distribution of data with spatial and temporal information. (b) Distribution of data only with spatial information. . .	61
4.9	After clustering the gesture information, 'a', without temporal information . . . . .	65
4.10	Mouse gesture after clustering the gesture information . . . . .	66
4.11	Finite State Machines with probability of gesture 'a'. . . . .	68
4.12	Whitening process . . . . .	71
4.13	bivariate normal distribution . . . . .	72
4.14	Contour line with 95% and 99% probability density. black line : 95%, pink line : 99% . . . . .	73
4.15	Overview of whole process . . . . .	75
4.16	The state compositions of the input data according to number of states (a) Four states (b) Five states (c) Six states (d) Seven states . . . . .	78
5.1	The gestures which have completely different distribution of spatial and temporal information . . . . .	83



5.2	The gestures which have completely same spatial distribution, but are drawn in the other direction. . . . .	84
5.3	The gestures which have completely same distribution, but different start position. . . . .	85
5.4	The gestures which have some few similarities for spatial and temporal distribution. . . . .	86
5.5	The gestures which have considerable similarities for spatial and temporal distribution. . . . .	87
5.6	The state structures of (a) and (b) in figure 5.5 . . . . .	87
5.7	The mahalanobis distance for each state and total distance. . . . .	88
5.8	The state structures of (c) and (d) in figure 5.5 . . . . .	89
5.9	The average Mahalanobis distance for one data point with various inputs in a recognizer. ★ shape means that the gestures are already rejected. 1. 'g' gesture recognizer 2. 'q' gesture recognizer 3. 'LQ' gesture recognizer 4. 'RQ' gesture recognizer . . . . .	90
5.10	The average mahalanobis distance for one data point with various inputs in a recognizer. (a) 'g' recognizer (b) 'q' recognizer (c) 'LQ' recognizer (d) 'RQ' recognizer . . . . .	91
5.11	The average mahalanobis distance for one data point with several clustering steps. (a) 'g' recognizer (b) 'q' recognizer . . . . .	91
5.12	The same input gestures that have different time periods (Uniform case)	93
5.13	The same gestures that have the different input speed . . . . .	94
5.14	The same gestures which have the various input speed (Not uniform case) . . . . .	95
5.15	The same gestures which have the various input speed in specific parts	95
5.16	The log-likelihood for each data input (a) data1 recognizer. (b) data2 recognizer. (c) data3 recognizer. . . . .	96
5.17	combined method for continuous hand gesture recognition (a) Rock posture is considered as the gesture (b) Paper posture is considered as the meaningless movement . . . . .	97
5.18	The moment to change rock from paper (a) it is classified as paper (b) it is classified as rock . . . . .	98
5.19	The operation method of median filter to solve the 'paper' hand posture fading problem . . . . .	99

# **Chapter 1**

## **Introduction**

The use of gestures is an important element of human communication, comprising the ability to express thoughts and intentions without sound. Besides general communication, gestures are used in many formalized ways, such as sign languages, traffic signals and for the military.

A sketchpad using a pen to control the objects on a tablet is considered the first form of a gesture recognition system, referred to as stroke based gestures (Sutherland, 1964). Since then, the research in this field have been intensive, and the researchers have developed various forms of recognition systems to receive the input gestures naturally, using the devices such as gloves, markers, or cameras. (Wexelblat, 1995; Bolt, 1980; Freeman and Weissman, 1995).

Direct interaction and communication with computers, referred to as Human-Computer Interaction (HCI), has become more important; thus, the gestures being recognized naturally by computers is a field of significant interest. Thus, gestures are widely held as a method that enables natural HCI in order to do various sort of task and applications. Therefore, recognition systems for hand posture and gesture are proposed in this thesis.

### **1.1 Hand shape and gesture recognition**

Pattern recognition systems based on vision have been actively researched for several purposes in various fields. There are many applications using pattern recognition, such as tele-operation (Kofman et al., 2005; Hu et al., 2003), biometric recognition (Jain

et al., 2004; Delac and Grgic, 2004), translation of sign language (Hernandez-Rebollar et al., 2002; Pugeault and Bowden, 2011), and so on. In biometric recognition, features of the face, iris, voice, hand, and so on have been used for identification (Yoruk et al., 2006). In particular, many researchers have studied hand and gesture recognition, as they show good performances for verification, and the equipment required for recognition is relatively cheap and simple (Bulatov et al., 2004).

There are two categories that can classify hand posture and gesture recognition (Munib et al., 2007). The first category is methods using devices such as gloves to obtain position, orientation, and movement of hand. Such methods show stable performance, as they can create abundant information about the hand's position and orientation as well as the shape of hand. However, such methods require data collection devices, which may be expensive, and uncomfortable to wear/use. For this reason, recent research has focused on the second category; methods using computer vision algorithms with image processing. Because it is considered as a method which enable the natural interaction with computers. Such methods are based on the vision using a digital camera, a (Microsoft) Kinect device, or others. Many techniques have been developed to obtain the proper features that match with the target information, such as Haar-like features (Viola and Jones, 2004), Histogram of Gradients (HOG) features (Dalal and Triggs, 2005), neural networks, and so on.

This paper proposed a method and system that can classify and recognize the input images or gestures obtained using a web-camera without using any expensive devices such as gloves or marking.

## 1.2 Motivation and objectives

Gesture recognition by computers is of increasing significance. Because the use of gesture is an important part to express thoughts and intentions non-verbally. For example, Thieffry (1981) said that a gesture is a physical manifestation including facial expressions, eye movements, and so on, of a mental concept. Another definition is that a gesture is a non-verbal expression of feelings and thoughts (Verma and Dev, 2009). The definition and importance of the gestures is the underlying motivation for the present study.

Another reason for studying gesture recognition is that there are many existing and

prospective applications requiring gesture-based HCI with applications spanning commercial entertainment, industrial, and medical applications. For example, there are many products being gradually commercialized on the market, such as the gesture recognition to control smartphones or televisions. This thesis sets out to devise a vision based gesture recognition system to obtain and recognize the gesture data with intuitive methods. Because a web-camera is cheaper and less awkward than other gesture data collection devices, as mentioned above.

A hand posture and gesture recognition systems based on Finite State Machines (FSMs) with probabilities is proposed here. It was motivated by work on the several FSMs systems, enabling the fast and straightforward recognition (Hong et al., 2000a; Verma and Dev, 2009). They achieve good performances for various input data. However, the present study is aimed at developing a superior hand gesture recognition system. In addition, this dissertation aimed at exhaustive analysis for the FSMs-based gesture recognition systems compared with the HMMs based systems.

One of the aims here is to develop a system that is effective, cheap, and simple. This dissertation wants to make the recognition systems which can work in real-time at frame rates using several input data such as mouse device and web-camera. For mouse device, the positions of the mouse cursor are a proxy for the gestures, and its positions are saved to the data set. In order to use a web-camera, the various vision algorithms have to be developed and used. If complex image processing algorithms are used, then the computational complexity is increased. Thus, to simplify this process, 2D points that are considered as a simple feature representing the center of the user's hands in an image from every frame of video are used for the input of gesture recognizer. These features are obtained by a color-based approach, edge detection, and vector extraction, which represents the distance between the center and the edge of the user's hands. Using these various data sets of postures and gestures obtained by mouse device or a camera, the data are trained for making the posture classifier and FSMs based recognizer systems and test the relative performance of the proposed method.

Inspired by these usage and significance, the systems for hand posture and gesture recognition based on vision and the FSMs with probability were proposed in this thesis.

### 1.3 Organization of dissertation

The organization of this dissertation is as follows. In this chapter, the reasons why hand posture and gesture recognition are important are described. The background for this research is described in Chapter 2, which introduces the various methods for gesture recognition as well as the application issues relating to hand and posture recognition.

Chapter 3 introduces the classifier systems for hand posture recognition based on vision using a cheap web-camera. This research focuses on the recognition of the American Sign Language (ASL) finger-spelling alphabet. The 25 hand shapes, including 24 letters, are used for the classes. J and Z are excluded from the classes, as they are comprised of sequences rather than a single positioning. In the proposed methods, techniques of image processing were used for obtaining an accurate center of input image for classification. Classification is done by comparing a distance distribution of a given image with data set obtained by image processing techniques. The experiment environment and methods of the algorithm are introduced in more detail; finally, the various experiments and results of the proposed methods are provided.

Chapter 4 explains the proposed method for hand gesture recognition based on FSMs with probability. Mouse gesture and hand gesture' information from a single camera are used as the inputs for the experiments. The position of mouse (x and y) using a mouse input device and the centroid of the hand from video are used. The training method and recognizer using data sets of gestures obtained by various input devices is also introduced. In this section, the proposed recognizer are explained in more detail. The experiments are performed with various sort of gesture input, using a mouse and web-camera to confirm the performance of the proposed recognizer.

In chapter 5, the experiments and findings are described. Various experiments are performed to analyze the robustness of the proposed method. To that end, the experiments are performed by using several different conditions for structure and temporal information of input data. The overlap problems were checked to confirm the effectiveness of the proposed methods. There are two kinds of overlap situations: spatial and temporal overlap. The effect of these parameters were analyzed. The experiments are also performed with the same gestures made over different time period. These experiments are designed to confirm the properties of the proposed method. Finally, the new method is explained to use the recognition systems in continuous time. This combines the methods of chapters three and four for the continuous gesture input, which is also referred

to as "live video".

Finally, Chapter 6 presents conclusions about the proposed model. In this chapter, the whole dissertation is also summarized, and future works are discussed.

# **Chapter 2**

## **Background**

Pattern recognition systems based on vision have been actively researched for several purposes in various fields. In particular, many researchers have studied hand and gesture recognition, as it shows good performance for verification, and the equipment required for recognition is relatively cheap and simple (Bulatov et al., 2004). However, making these recognition systems is not an easy task. There are many phases to be carefully designed to achieve good recognition performance, such as hand modeling, hand extraction from the images or videos, and analysis and interpretation of the input. Pavlovic et al. (1997) shows these phases as per the diagram in Figure 2.1. The information of gestures is obtained by one or more cameras. Subsequently, it is analyzed to estimate the gesture model parameters. The estimated parameters of the gesture model are used to determine the meanings of gestures.

In this chapter, an overview of the hand gesture recognition system and the application issues is given in detail.

### **2.1 The concept of hand gesture recognition**

Various human sensors are used in human-human communication, or human-computer communication (Jaimes and Sebe, 2007). For example, speaking, gaze, facial expression, gesture, and other elements are involved in this. In particular, the use of gesture is an important part to express thoughts and intentions non-verbally. Research by Hall (1959) indicated that human communication consists of 35% verbal communication and 65% non-verbal gesture communication. In other words, gestures are a major

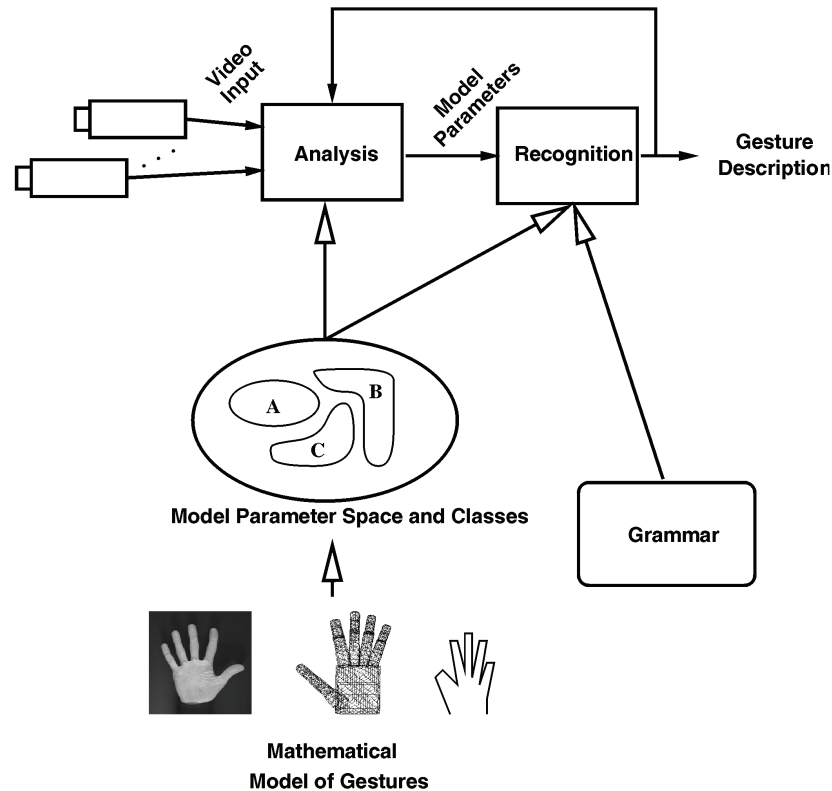


Figure 2.1: Vision-based gesture interpretation system (Reprinted from Pavlovic et al. (1997))

subset of human communication.

Finding the exact definition for the concept of the "gesture" is not simple, because this term is widely used in various fields with different, but similar, meanings. Thief-fry (1981) defined gestures as the physical expression of mental concept. Mitra and Acharya (2007) stated that the gestures are composed of the physical movement of the hands, arms, face, and body which convey the meaning or information. Among body parts, hands are most widely used for gesturing. Figure 2.2 demonstrates the truth of this fact, because hands are natural ways to make gestures (Karam, 2006).

There is a wide variety of hand gestures, depending on the context. When the person communicates with other people, hands play an important role to convey our thoughts (Zabulis et al., 2009). For example, a hand can indicate a direction or object non-verbally. It is also a more intuitive method for HCI than conventional methods. However, in HCI, hand gestures have to be capable in practical terms to be used for controlling a computer, regardless of the natural use of the hand (Kendon, 1986). Hence, gestures that are used in this field are defined slightly different to the real life meaning



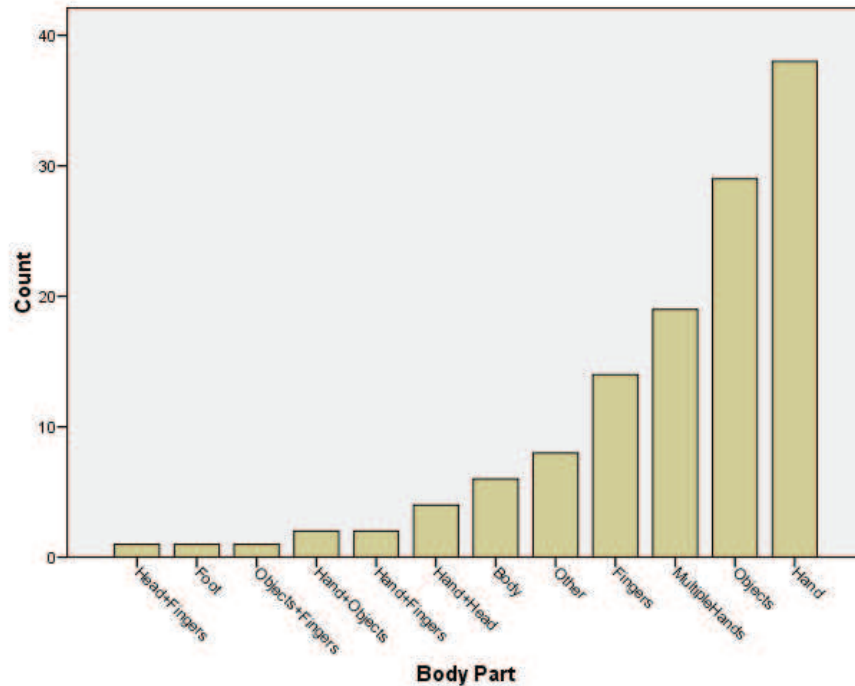


Figure 2.2: Body parts identified in the literature employed for making gestures (Reprinted from Karam (2006))

for maximizing the efficiency of the current technologies.

Gesture recognition is defined as the whole process of tracking gestures and ascribing meaning to them. Gesture recognition is presently one of the most active and vibrant research fields (Yamato et al., 1992; Lee and Kim, 1999; Yang et al., 2009).

Mitra and Acharya (2007) classified hand gestures as static and dynamic. Static hand gestures are defined as a certain pose and spatial orientation of the hand or arm without any temporal information. In other words, it is a single pose/position without movement. On the other hand, if there is movement considered, it is called dynamic hand gesture. Ottenheimer (2008) categorized dynamic hand gestures as five types; regulators, affect displays, illustrators, emblems, and adaptors. Regulators are defined as gestures that can control interaction. Illustrators are gestures emphasizing speech. Emblems are gestures that can be easily translated as short verbal communication. When person uses some unintended gestures in communication, they are called adaptors. Each of these five types of categories involve both temporal and spatial variation to represent the gesture, such as a waving hand. Therefore, in dynamic gesture recognition, information on trajectories of hands is needed to analyze the meaning.

## 2.2 Techniques for extracting the hand gesture

The feature extraction process is an important process to obtain the relevant data on the hand on the basis of an image. Huang and Pavlovic (1995) distinguish hand gesture recognition techniques according to the method of feature extraction. There are many techniques to enable HCI; however, in the present paper, two types of techniques are concerned: contact based techniques and vision-based techniques. These two techniques have been widely used for extracting the features of the hand in recognition systems, and there are described in turn in the next two subsections.

### 2.2.1 Contact based techniques

To control a computer or work with a computer using the user's hand conventionally, intermediary devices such as keyboards, mouse, and so on are required (Sturman and Zeltzer, 1994). Because movement of our hands cannot be recognized naturally by the computer, they have to be transferred to a language it can interpret. For this reason, much research has been done to develop devices that can easily transfer the user's hand information.

Contact-based techniques, which are the result of these efforts, use a cloth-made glove equipped with sensors for obtaining the gesture data. The features extracted by a glove, accelerometers, and so on are used for analyzing and interpreting the meaning of gestures. These techniques can be classified into five different methods: mechanical, haptic, ultrasonic, inertial and magnetic. Bolt (1980) used the tracking sensor called Polhemus to facilitate communication of the user's hand position to the computer. This sensor can be attached to the hand and relay information about the position and orientation of the hand in relation to the screen. Baudel and Beaudouin-Lafon (1993) developed a dataglove named Charade, that can control computer-based presentation displays using hand gestures. In this system, the user wears a VPL dataglove, which is connected physically to the serial port of the computer. The bending of each fingers, and the position and orientation of the hand in 3D space can be measured by the dataglove. Figure 2.3 shows the whole form of this system.

Cipolla et al. (1993) proposed the method computing motion parallax to obtain 3D visual interpretations of hand gestures. For this process, they used a glove with colored markers attached as input to the system. With information obtained by a glove, the

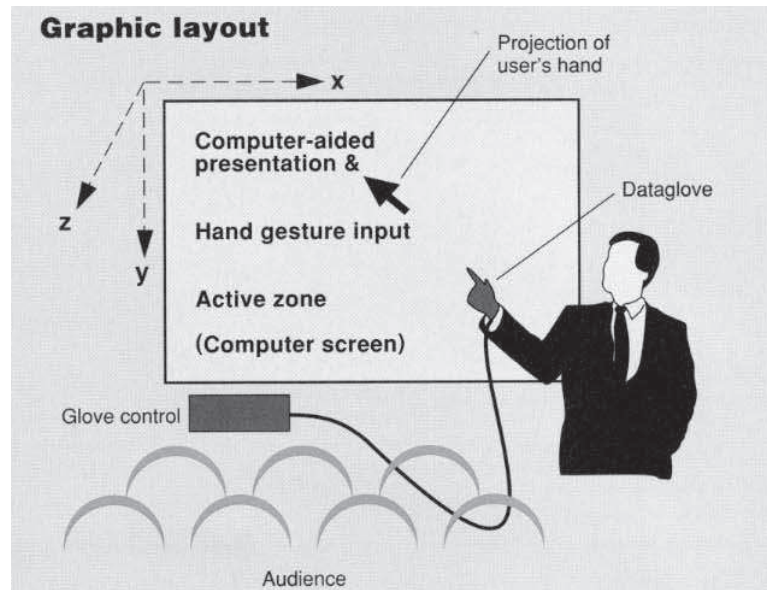


Figure 2.3: Whole system of Charade (Reprinted from Baudel and Beaudouin-Lafon (1993))

relative velocity of a nearby feature point can be measured to compute motion parallax. The Nintendo Wii-Remote is a device to detect the movement of the user's hand or arm using an accelerometer. Using contact-based techniques can easily obtain the accurate data on the movement of the hand, and its composition is simple. However, since the user has to wear the devices, it is uncomfortable. In addition, there are critical problems with these techniques, relating to health (Schultz et al., 2003). For example, Nishikawa et al. (2003) mention the risk of cancer when the user uses magnetic devices. There are also problems that can be made by devices in sustained contact with the skin, such as allergies.

## 2.2.2 Vision-based techniques

Due to the problems associated with contact-based techniques, many systems and techniques have been researched for glove-free approaches for hand gesture recognition (Mase and Suenaga, 1992; Darrell and Pentland, 1993). Mase and Suenaga (1992) proposed a system called Finger Pointer, which can recognize a pointing action in real-time. Stereo image sequences are used in the system without any glove or image processing hardware. Vision based techniques used one or more cameras to capture video sequence for detecting and analyzing the hand movements using computer vision algorithms. These techniques use visual inputs from the camera to deliver the

feature information. However, vision based techniques face the problem of the variety and non-uniformity of hand gestures; these are manifested also in the view point of the camera, such as different silhouette scales, resolution of images, and movement speed and direction. Another problem is that the process takes place in real time. For using vision based techniques, input images with image processing have to be treated at the frame rate, which can be difficult to achieve. In addition to this problem, different lighting conditions and cluttered backgrounds are also persistent problems. Therefore, vision-based techniques have suffered from configuration complexity and occlusion problems. However, these techniques are user friendly and bring no health issues, as the user just stands up in front of camera and moves their hands or arms. They offer good solutions that are cheap and practical. For that reason, they are a highly promising field of development in HCI.

## **2.3 Vision based hand gesture models**

Many researchers have proposed the model of hands or body parts to represent the features to be used in gesture analysis. There are two major categories to represent hand gestures (Derpanis, 2004).

- Model-Based Approaches
- Appearance-Based Approaches

### **2.3.1 Model-based approaches**

Model based approaches can extract the features using kinematics models (Ahmad, 1994). These approaches are categorized into three different types: textured volumetric, geometric model, and skeleton model according to the method of gesture representation (Bourke et al., 2007). Textured volumetric and skeleton model can give precise information of the hand skeleton and surface skin. Figure 2.4 shows examples of the textured volumetric model and skeleton model.

All of these approaches have to be 2D projections for extracting the kinematics parameters. This is necessary to compute the spatial description of hand poses with joint

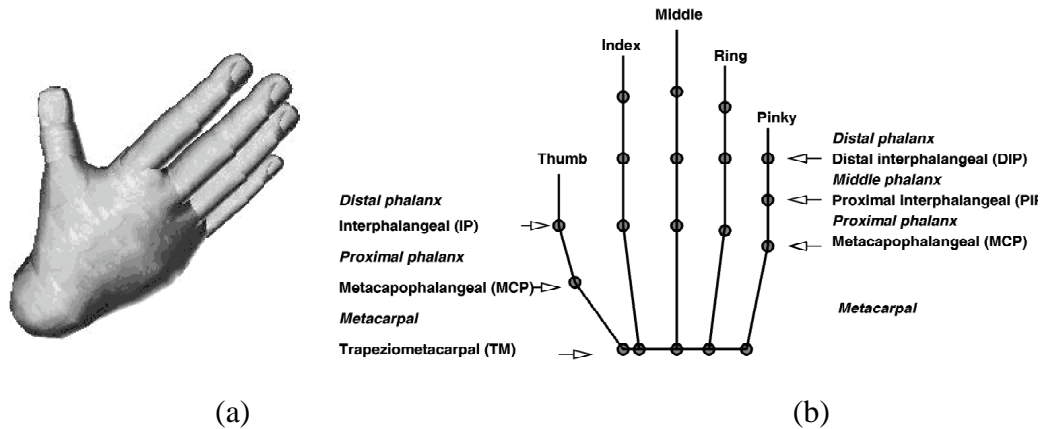


Figure 2.4: Examples of (a) 3D textured volumetric (b) skeleton model (Reprinted from Pavlovic et al. (1997))

angles and so on. Rehg and Kanade (1994b) proposed one of the very first approaches for hand tracking based on the model approach. Wu et al. (2001) proposed probabilistic methods referred as a Bayesian framework to achieve optimal estimation of the parameters. Although the parameters or features have to be in linear systems and have to follow Gaussian distributions, a Kalman filter can also be used to do this task. In addition, a condensation algorithm can also be used for estimating the parameters (Black and Jepson, 1998; Rittscher and Blake, 1999). The probabilistic methods show good performances in estimating the parameters; however, if the dimensions are high, then there are computational problems.

Model based approaches have the advantage that they can generate precise hand gesture representation, despite requiring heavy computation.

### 2.3.2 Appearance-based approaches

Appearance based approaches have been widely used to represent hand gestures as 2D intensity images related to the hand images. In other words, these approaches represent the gestures as sequences of images. Therefore, in contrast with model based approaches, these approaches cannot directly derive the parameters from the 3D spatial description of the hand. There are several 2D models that are used commonly such as the color based model, silhouette model, contour model, and motion based model. The color based model uses markers to obtain the pose or motion of hands. Bretzner et al. (2002) proposed the method using multi-scale color features with parti-

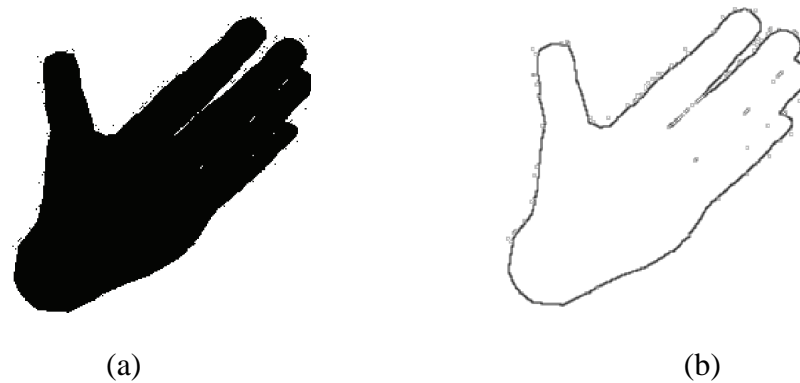


Figure 2.5: Examples of (a) Binary silhouette model (b) Contour model (Reprinted from Pavlovic et al. (1997))

cle filtering. Silhouette models use geometric properties of the hand silhouette such as surface, convexity, centroid and orientation. Contour models commonly represent the hand gestures as the edge of the hand to match this information with a template model (Cipolla and Hollinghurst, 1996; Cootes et al., 1995). Motion based models are based on the motion of hands from the image sequences. Figure 2.5 shows examples of the binary silhouette model and contour model.

## 2.4 Vision based hand gesture recognition techniques

For recognizing the hand gesture automatically, three fundamental steps are needed for vision based hand gesture recognition systems: detection, tracking, recognition. Figure 2.6 shows the three fundamental steps' architectures with commonly used techniques.

The present section introduces these three fundamental steps one by one.

### 2.4.1 Detection

The very first step to recognize gestures is hand detection from the images. If detection is not successful, then hand recognition cannot even begin, so it is the primary step in hand gesture recognition systems. Thus, much research has been done to develop efficient methods to extract features. For example, skin color, shape, and 3D model of hands can be used as the features; these are described in turn in the next three subsections.

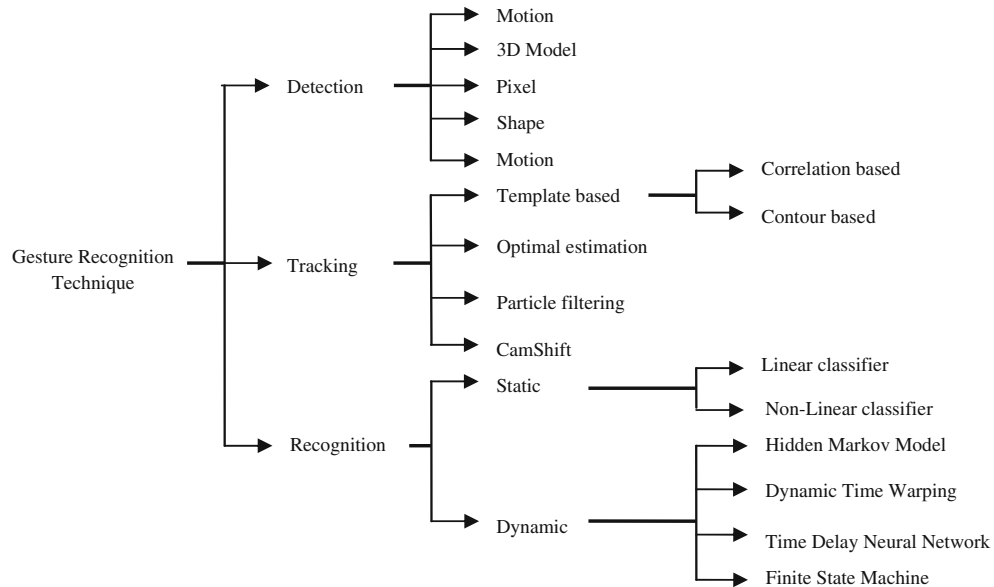


Figure 2.6: Vision based hand gesture recognition techniques (Reprinted from Rautaray and Agrawal (2012))

#### 2.4.1.1 Skin color

The most important element in skin color approaches is the decision of the color space that is used to detect hands. There are many color spaces such as RGB, HSV, YCbCr, and  $La^*b^*$ . If the proper color spaces are used according to the experimental perspective, several problems, such as different illumination and cluttered background, can be solved suitably. Bradski (1998), Kurata et al. (2001) and others have proposed many methods robust against illumination variance. See Terrillon et al. (2000) for a comparison of the performances of various skin chromaticity models.

Another problem is the cluttered or similar-color to hands backgrounds. If the color of the background is similar to that of the hands, or there are many objects in the background, then skin color approaches can be ineffective. To solve these problems, Reh and Kanade (1994a) proposed the method of background subtraction. However, it has the assumption that the camera systems in be fixed in terms of the background. There has been extensive research to overcome this problem (Blake et al., 1999).

In the present research, RGB color spaces are used to extract hands from the images and video. The research assumed that the background is static and illumination is not much changed.

#### 2.4.1.2 Shape

The shape of hands has been used to detect hands from the images. As already introduced, the contour of hands can give much information independent of camera position, skin color, and illumination. The contour line is obtained by the edge detection; however, many edges (other than contour) are induced by this method. Therefore, in order to obtain the precise contours of the hand and increase reliability, sophisticated pre-processing and post-processing are needed. Typical pre-processing is the smoothing of images, and typical post-processing is thinning.

Phadtare et al. (2012) proposed a method using 3D hand shapes obtained by Kinect. The shapes of hand are obtained by Kinect, then the context, which determines the hand shape, is compared with example shapes in the database. If shape of hands is used for detecting hands, the similar shapes have to be used in the database for comparison.

the whole shape of hands can be used, but also shapes of parts of hands, such as fingertips. Song and Takatsuka (2005) developed a finger pointing interface with stereo vision in real-time. A 3D fingertip tracking system is used; however, it requires the careful calibration and efficient localization algorithm to obtain proper performances.

The proposed method uses the contour information of hands with edge detection and skin color methods.

#### 2.4.1.3 3D model

This approach is the method of using 3D models in order to detect the hand in images. Hands move fast, and there is self-occlusion in the images, so it is difficult to detect or find the corresponding real hand motion. This problem can be solved by using 3D models that have enough degrees of freedom to represent hands. The joint angle of the hand can be obtained from a kinematic hand model, so self-occlusion problem and other problems can be solved thereby.

Many researchers have developed various 3D hand models. Shimada et al. (1998) proposed a method to estimate joint angle of a hand and refine the 3D shape of hand. When the initial approximately-3D hand shape model is given, rough pose, such as position and joint angle of hand, is obtained by using synthesis. Subsequently, these parameters are refined by covariance ellipsoid of a probability distribution. From these processes, the precise pose of the hand can be obtained. Lee and Kunii (1995) proposed



a method that can automatically analyze 3D hand posture using stereo images and a deformable model framework to fit a 3D model of the hand into the images. They can estimate 27 interacting hand parameters, such as finger joint angles, so they can reconstruct 3D hand posture using these parameters.

#### **2.4.1.4 Motion**

Motion approaches have been less used than the other approaches, as these approaches are required a complicated setup for hand motion detection alone. Cui and Weng (1996) and Freeman and Weissman (1995) proposed the method assuming that the hand motion is the only motion in the images from the video.

### **2.4.2 Tracking**

Tracking is the process to find the hands or objects from the images on a frame to-frame basis. In other words, when hands or objects are moving, tracking in image sequences indicates continuously identifying hands or objects location (Lepetit and Fua, 2005). If the detection methods that were introduced above section are enough fast to extract the hand at the desired frame rate, then this method can be used continuously to track the hands. However, this is very difficult since the hand moves very fast. Areas of research for the solution of this problem are explained in the next section.

#### **2.4.2.1 Particle filtering**

Particle filtering, such as by condensation algorithm (Isard and Blake, 1998b) or Monte Carlo filter (Doucet et al., 2001), has been used to track the position of the hand. This approach can model the location of the hand with a set of particles. It has the advantage that it can be used more generally than Kalman filtering which is restricted to use of Gaussian distributions. In addition to this advantage, this method does not require the linearization of the relation between the state and measurements. However, if this method is used for a complex model, then it requires many particles to track the hand, and thus has high computation requirements, particularly for high dimensional models. To overcome this problem, Isard and Blake (1998a) proposed a condensation algorithm which is used reduced dimensionality by modeling commonly known constraints. This

algorithm is used to track curves against cluttered backgrounds, and it operates in real time.

#### **2.4.2.2 CAMshift**

The principle of mean shift algorithm is the base of CAMshift algorithm. The term CAMshift refers to continuous adaptive mean shift. Mean-shift is a kernel based tracking method that uses density based appearance models to represent targets (Yilmaz et al., 2006). Kernel means the object shape and appearance. For instance, the kernel can be a rectangular template or an elliptical shape. By computing the motion of the kernel in frame to frame, objects can be tracked. Commonly, parametric transformation such as translation, rotation, and affine is formed for the kernel motion. The CAMshift algorithm has been widely used because of the relatively low computation complexity and simplicity (Bradski and Kaehler, 2008).

### **2.4.3 Recognition**

The previous steps, detection and tracking, are the processes for obtaining the precise features of the hand from the images. Recognition is the last step to interpret the meaning of posture and gestures. To this end, after obtaining precise features, the recognizer has to be made to use these training gestures and recognize the new input gestures.

There are many techniques to make a recognizer for vision based hand gesture recognition. For example, the template matching technique can be used for recognizing the static gestures, while hidden markov models and finite state machines can be used for the dynamic gestures; thus, the techniques such as HMMs and FSMs have to handle the temporal aspect as well as spatial aspect (Lee and Kim, 1999). These techniques are referred to as automata based methods, which are composed of a set of states and transitions. Set of states represents the static hand gestures, such as posture of hand, and set of transitions represents the temporal or probabilistic information, which represents changes to the states. Since these methods can represent temporal information with set of states, a dynamic gesture is considered as a path that starts in an initial state and ends in a final state. The computation complexity of the automata based methods is large because if there are many gestures to be recognized or many states, then its

computations for joint probabilities are to be complexed.

Some techniques for static and dynamic gesture recognition were introduced as following subsections.

#### **2.4.3.1 K-means**

The K-means algorithm is a clustering algorithm that can find the space (in the entire data distribution) where the data are most concentrated (Lloyd, 1982). The basic step of k-means clustering is simple. At first, the number of clusters and center of these clusters have to be determined. Subsequently, the distance of each object to the centroids of clusters is computed. Euclidian distance is commonly used for computing the distance. Using these distances, groups the object based on minimum distance. This process is repeated until the iteration number set already is over or there are no improvements in the distance errors.

#### **2.4.3.2 Hidden Markov model**

HMMs have been widely used for, inter alia, speech and gesture recognition. The HMM is a doubly stochastic process model, as a process that cannot be observed can be estimated by other process of observation in this model. In other words, hidden states cannot be observed directly, but can be inferred through an observed sequence. Probabilities of output symbols in each state are computed by estimating the parameter of HMMs. A number of hidden states and observed states comprise an HMM, and each hidden state has a transition probability with other hidden states. There are some different models, the ergodic model and left-right model, for the topology of hidden state transition. The ergodic model is a fully connected topology where all states are connected with each other, so each state can change the all other states with one transition. Left-right topology is the proper type for the input changing over time, and indexes of state are always increased when transition takes place. The more detailed explanation of the HMM follows.

There are many terms and notations in HMM. For convenience, the notations of dis-

crete HMM are described as follows.

$$\begin{aligned}
 O &= o_1, o_2, \dots, o_T : \text{observed symbol sequence} \\
 V &= v_1, v_2, \dots, v_M : \text{set of possible output symbols} \\
 T &= \text{length of the observation sequence} \\
 M &= \text{number of observed symbols} \\
 Q &= q_1, q_2, \dots, q_N : \text{set of hidden states} \\
 N &= \text{number of hidden states} \\
 s_t &= \text{hidden state at time } t \\
 \pi &= \pi_i | \pi_i = P(s_1 = q_i), 1 \leq i \leq N : \text{initial state probability} \\
 A &= a_{ij} | a_{ij} = P(s_{t+1} = q_j | s_t = q_i) : \text{probability of state transition from } q_i \text{ to } q_j \\
 B &= b_j(k) | b_j(k) = P(v_k | s_t = q_j) : \text{probability of output symbols } v_k \text{ at state } q_j \\
 \lambda &= (\pi, A, B) : \text{parameter set of HMM model}
 \end{aligned} \tag{2.1}$$

The Observed symbol sequence has the observation vectors during a certain period of time,  $T$ . For example, if observed output symbols are three dimensional data,  $x, y, z$ , then the observation vector,  $o_t$ , also has a  $3 \times 1$  matrix. Initial state probability shows the probability of each state at  $t = 1$ , so the sum of elements in  $\pi$  must be 1. If there are three hidden states, then  $\pi$  will be a  $3 \times 1$  matrix. Probability of state transition matrix ( $a_{ij}$ ) shows the probability of transition from state  $i$  at time  $t$  to state  $j$  at time  $t + 1$ . Hence, the sum of the row in state the transition matrix has to be 1. Probability of output symbol, ( $b_j(k)$ ), indicates the probability that  $v_k$  is the output symbol when the state is  $q_j$  at time  $t$ .

HMM performs well in recognizing the time series data. It has a discrete version and continuous version for analyzing these data. In discrete HMM, input information, which has continuous values, on the average, has to be modified by vector quantization for using HMM with this information. However, if a stage of vector quantization is not performed properly, it distorts the observations sequence and thus is a principal cause of bad recognition rates. Hence, there have been studies about the continuous HMM using Gaussian distribution to strengthen the recognition rates using the input information directly without vector quantization. For using HMM in real life or applications, three fundamental problems of HMM must be solved.

The first problem is the evaluation of the probability problem. This problem concerns when the observation sequences ( $O$ ) and various HMM models( $\lambda$ ) are given, how  $P(O|\lambda)$  will be calculated effectively, and what kind of HMM models have the great-

est probability of observation sequences. It can be solved using a forward-backward algorithm.

The second is a problem related to the optimal state sequence. The problem concerns how the best optimal state sequence ( $Q = q_1, q_2, \dots, q_n$ ) will be found when the observation sequences ( $O$ ) and a HMM model ( $\lambda$ ) are given. It can be solved using the Viterbi algorithm (Viterbi, 1967).

The third problem is the most significant. This problem is related to adjustment of model parameter, and how the model parameter ( $\pi, A, B$ ) will be estimated to maximize  $P(O|\lambda)$ . The Baum-Welch parameter re-estimation algorithm has been used to solve this problem (Dempster et al., 1977). Each solution of the problems is now briefly explained, as explained in (Rabiner, 1989).

The forward-backward algorithm is the simplest method using trellis which is composed of all possible state sequences about the observation sequences in length  $T$  to obtain  $P(O|\lambda)$  as in the equation below.

$$P(O|\lambda) = \sum_Q P(O, Q|\lambda) \quad (2.2)$$

where  $Q = q_1, q_2, \dots, q_N$ . In this case, all possible number of state sequences are  $N^T$ , and its complexity is  $O(TN^T)$ . Therefore, it cannot be used in a very simple problem. To solve these problems, forward algorithm and backward algorithm are used. These methods are simple. In the case of forward algorithm,  $P(O|\lambda)$  is easily computed using forward variable  $\alpha_t(i)$  as in the following equation.

$$\begin{aligned} \alpha_t(i) &= P(O, s_t = q_i|\lambda) = P(o_1, o_2, \dots, o_t, s_t = q_i|\lambda) \\ P(O|\lambda) &= \sum_{i=1}^N \alpha_T(i) \end{aligned} \quad (2.3)$$

In a similar way,  $P(O|\lambda)$  is easily calculated using backward variable  $\beta_t(i)$

$$\begin{aligned} \beta_t(i) &= P(o_{t+1}, o_{t+2}, \dots, o_T | s_t = q_i, \lambda) \\ P(O|\lambda) &= \sum_{i=1}^N \pi_i b_i(o_1) \beta_1(i) \end{aligned} \quad (2.4)$$

Using these two algorithms, the complexity becomes  $O(TN^2)$ .

The Viterbi algorithm has difficulty in determining the exact state sequences in the observation sequences; however, the Viterbi algorithm can identify the best optimal state

sequences. There is also some variable called delta which is defined in the equation below.

$$\delta_t(i) = \max_{s_1, s_2, \dots, s_t} P(s_1 s_2 \dots s_t = q_i, o_1 o_2 \dots o_t | \lambda) \quad (2.5)$$

This variable is that the largest probability that has a pass at time  $t$  with  $t$  observations and state  $q_i$  at time  $t$ . This variable can be represented as a formula.

$$\delta_{t+1}(j) = \max_i [\delta_t(i) a_{ij}] b_j(o_{t+1}) \quad (2.6)$$

Using these variables, the best sequential track can be obtained.

Baum-Welch re-estimation algorithm is used to obtain the  $\lambda = (A, B, \pi)$ , which makes the maximum probability  $P(O|\lambda)$ . This is the most important problem to determine the performance of HMM. This also involves the expectation maximization (EM) algorithm, using the stage of maximization and the stage of expectation. This algorithm is continuously used until  $P(O|\lambda_{t+1})$  becomes smaller than  $P(O|\lambda_t)$  or some fixed iteration number. This will be explained in more detail in Chapter 4.

HMM has been widely used in the statistical method using training data for the recognition of speeches or gestures, so it has already been used in various fields.

It has been widely used in speech recognition, and it shows a good success rate. Baker (Baker, 1975) proposed the DRAGON system, representing each of the important sources for the recognition of continuous speech; in addition, the research of many other applications on speech recognition has flourished (Bahl and Jelinek, 1975; Bahl et al., 1983; Jelinek et al., 1975; Juang and Rabiner, 1991; Rabiner and Juang, 1993).

There have been numerous studies concerning off-line and on-line handwriting recognition (Hu et al., 1996, 2000; Plamondon and Srihari, 2000). Off-line handwriting recognition means that hand-written words or sentences are recognized through the scanner or other methods. On the other hand, on-line handwriting recognition involves recognizing words hand-written with an electronic device. In on-line handwriting recognition, the information of  $x$  and  $y$  coordinates about each digit can be obtained in a real time. Motion video tracking or stock price prediction are other popular applications using HMM.

There have also been researches for designing the control structures using internal memory in various autonomous agent fields using HMMs (Yim and Kim, 2013b). This is because HMMs are sometimes referred to as Probabilistic FSMs (PFSMs). PFSMs

have the ability to move other states, in contrast with DFSMs, thus, if the PFSMs can be used for the control structures of agents, it will make the powerful performances to the noise inputs.

There are two HMM architecture relating to the methods of which emitted symbols. The first architecture is state-emission HMM, which takes the form of Moore machines. In state-emission HMM, output probabilities can be computed as  $P(v_k | s_t = q_j) = P(b_j(k))$ . In other word, the output symbols are only observed at each state. On the contrary, the second architecture takes the form of Mealy machines, and it is named arc-emission HMM. The output symbols are observed during state transition in this architecture, so its output probabilities are calculated by  $P(v_k, s_t = q_j | s_{t-1} = q_i)$ .

### 2.4.3.3 Finite state machine

Finite state machines are the methods that use finite numbers of states to train the gestures. This approach is used for various fields other than that of gestures. For example, FSMs have been used to encode internal memory in evolutionary computation, and it has shown a good performance with complex problems (Kim, 2006; Yim and Kim, 2012, 2013c).

It is a mathematical model that was developed for practical applications. FSMs are considered as a Mealy machine model (Kohavi, 1978), which is composed of states and actions and has good properties to understand simple memory structures.

FSMs are defined as  $M = (\Sigma, \Gamma, S, s_0, \delta, \omega)$ .  $\Sigma$  is a finite non empty set of input values;  $\Gamma$  is a set of output values;  $S$  is a finite set of states;  $s_0$  is the initial state, which is an element of  $S$ ;  $\delta$  is the state transition function; and  $\omega$  is the output set.

The data distribution of gesture recognition systems is represented with states,  $S$ . In these machines, only one state is taken at each time step, referred to as current state. There are state transitions, where the current state is changed to another state, which is called next state by the fixed rules. The next state and the output are a result of the current state and input.

In many cases, deterministic FSMs, also known as deterministic finite automaton, are used for solving the complex problems. The next states in DFSMs are uniquely determined. In other words, it cannot change other states except for that which is already determined. However, when the input sequences have noise, DFSMs can make wrong

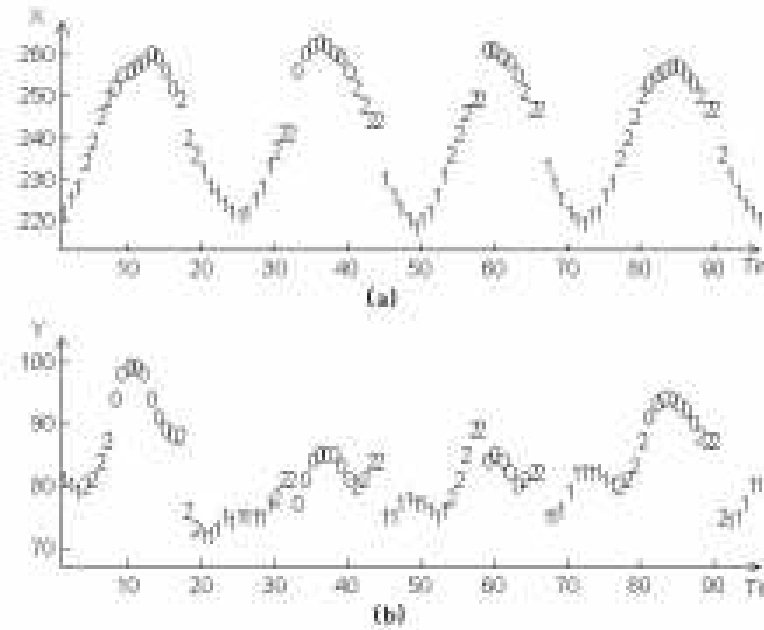


Figure 2.7: Label sequence corresponding to part of the data (Reprinted from Hong et al. (2000b))

decisions.

Several previous studies have used FSMs approaches for gesture recognition. (Hong et al., 2000b,a) proposed the method that models each gesture as an FSM. In this method, gestures are modeled as an ordered sequence of states in the FSMs approaches. Figure 2.7 shows an ordered sequence of states for a certain gesture.

The training of the gesture model is done off-line with various clustering algorithms to cluster the gestures in the spatial spaces. Through the training phase, the characteristics of each state that comprises a gesture are derived. When the clustering phase is worked, they assumed that the variance of each cluster is isotropic. Therefore, it has same cluster size. The number of states (clusters) is a key concern in achieving high recognition success rates, and it is determined according to the applications. Fewer training sets are needed in FSMs approaches than the HMMs approaches, because the trajectories of gesture in space are the most important factors in the former approach. Therefore, if there are suitable training data sets for gestures, the numbers of sets are not important factors, unlike in the HMMs approaches. After the clustering is complete, the distances between each data point and the center of clusters are computed to assign the clusters to the data point. Each data point chooses the closest cluster from among all clusters. After assigning clusters to each data point, the alignment phase be-



gins to align the data according to each assigned cluster. Each data point is reassigned to the state to which it belongs in accordance with the time. Through these training works, the gestures are redefined as the ordered sequences of states.

Subsequently, the gesture recognition can be performed online using trained FSMs. When the input gestures are entered to the FSM recognizer, if the input data is passed through all state sequences and reaches a final state, then the input gesture is recognized. Bobick and Wilson (1997) also proposed the method where the gesture is considered as a prototype trajectory in a 2D space.

Detailed explanations of properties of FSMs and recognizers based on FSMs are introduced in Chapter 4.

## 2.5 Application of Hand Gesture Recognition

There are many applications using gesture recognition, including of recognition of sign language (Starner, 1995) or recognition of face expression. For tracking or recognizing gestures, input images from the camera are analyzed. Motion video tracking or stock price prediction are another application using HMM.

An important applied area for gesture recognition techniques is communication applications - that is, interpretation of sign language. Since sign languages consist of gestures used in human-human style interaction, it came naturally for gesture recognition research to help towards this direction and develop the method for use in HCI. Starner and Pentland (1997) and Kadous (1995) proposed methods to recognize 40 words of the American sign language and 95 words of the Australian sign language, respectively. Murakami and Taguchi (1991) recognized both finger and sign words of the Japanese sign language, while Imagawa et al. (1998) implemented a bi-directional sign language translator.

Virtual reality (VR) application is a relatively new field in HCI. VR applications use gestures to manipulate the virtual objects or other people using user hands or body while navigating in virtual environments as second life (Boulos et al., 2007), 3D display interactions (Sharma et al., 1996), or Lively from Google. In addition, there are many other studies on interaction with and manipulation of objects in virtual environments using hand gestures (Boulic et al., 1996; Bryson, 1996). These can be divided

into three subcategories according to the degree of participation of user in virtual environments: non-immersed interaction (Segen and Kumar, 1998), semi-immersed interaction (Krueger et al., 1985), fully-immersed interaction (Song et al., 2000). Non-immersed interaction is that where users are not represented in the virtual environments; An avatar is used in semi-immersed interaction, while users are represented and participate inside the virtual environments in fully-immersed interaction.

Hand gestures can be also used to control robot movements by video from cameras located on the robot (Goza et al., 2004); thus, it can boost the effectiveness of human-robot interaction. This application is slightly different to fully-immersed interaction, because this application is performed in the real world. Becker et al. (1999) proposed robot platforms for manipulating and instructing robots.

In 3D modeling applications, using hand gesture can facilitate intuitive designing or manipulating of 3D objects. Users can design, create, and control 3D objects in a more natural way, as in 3D animated movies, since neither a keyboard nor mouse device is used. Zeleznik et al. (2007) and Kiyokawa et al. (1996) are examples of applications where hand gestures help in the creation of 3D models.

There are also many applications that can be used for home appliances in a house. Recently, the houses where these applications are used have been referred to as "intelligent house". User can turn on/off the light, adjust temperature, or manipulate the television using hand gestures. Freeman and Weissman (1995) proposed a system that uses these applications for manipulating television, such as adjusting volume, changing channels, and turning the television on and off.

## **2.6 Summary of Chapter2**

this chapter is concentrated on the explanation of vision based hand gestures, since it is more intuitive of HCI. The concepts of hand gesture recognition and the various techniques for detecting, tracking, and recognizing the hand gesture are examined in this chapter. Each technique has pros and cons; nevertheless, if these techniques are properly used in accordance with the intended usage, then good performance can be obtained in hand gesture recognition. The various applications are also introduced where the recognition systems for hand gestures can be used in real life.

## Chapter 3

# Hand Shape Recognition

In this chapter, a method and system is proposed that can classify and recognize the input images obtained using a web-camera without using any expensive devices such as gloves or marking. The American Sign Language (ASL) finger-spelling alphabet is used to verify the performance of systems. Sign-language recognition is a difficult problems, as there are many factors to decide the mean of the gestures, such as shape, posture, and movement. Stokoe (2005) said that there are signs and finger-spelled English words in sign language. The former distinguishes the sign language using hand posture, position change, or movements. The latter only focuses on the shape of the fingers, not on the position or movement of the hands. The finger-spelled word has a one-to-one correspondence with the letters. Various methods have been used for recognizing the gestures and poses of the hand in the former case. For instance, Cutler and Turk (1998) developed a real-time recognition method using optical flow of hand motion. They used optical flow to estimate and segment the motion blobs. Glove-based input is also used for hand tracking in several fields (Sturman and Zeltzer, 1994).

The present study focuses on the recognition of American Sign Language (ASL) finger-spelling alphabet based on vision using a cheap webcam. There are several similar studies proposing and testing methods to classify the ASL finger-spelling alphabet. Yoruk et al. (2006) proposed a method using the shape of the hand silhouette for recognizing the individual hands. Pugeault and Bowden (2011) used Kinect to use depth information of images with standard intensities of images. Van den Bergh and Van Gool (2011) used a Time-of-Flight (ToF) camera and RGB camera for gesture recognition. The images obtained by RGB camera are merged with depth images that

are obtained by ToF camera. Munib et al. (2007) proposed a method based on neural network and Hough transform with 20 different shapes of hand. They used the edge filter to obtain the contour of fingers.

In the method proposed in the present dissertation, techniques of image processing are used for obtaining an accurate center of input image for classification. There are three parts of image processing in the proposed method (see the next subsection for details).

The 25 hand shapes including 24 letters, were used for the classes. J and Z were excluded from the classes, because these letters comprise sequences. The experiments were performed with two cases: first, the classification for each class; second is the classification for grouping classes that were grouped on the basis of similarity of shape. The hand model proposed in this dissertation, Hand Contour Vector(HCV), has an ability to classify the ASL finger-spelling letters that have similar shapes such as 'A', 'E', 'M', 'N', 'S', and 'T'. This shows the better performances than with the existing methods. However, when static gestures are used for HCI contexts such as TV and smart-phone control, some of ASL finger-spelling letters have to be grouped. This because, in order to distinguish the finger-spelling alphabets that have similar forms, extra image processing is needed to extract more abundant information. These processes have high computational complexity, and, thus, are hard to work in real-time. Therefore, in these applications, the static gestures that can be distinguished with certainty are commonly used. This dissertation shows the results of both cases, and the performance of the proposed method will be compared with the methods previously studied.

$320 \times 240$  images are used for the experiments. 3000 finger-spelling alphabet images are obtained (120 images in each class). These images were obtained by four participants who are non-native to sign language.

This section is to be prepared for submission as a scientific paper (Yim and Kim, 2013a).

### 3.1 Methods

Various hand shapes are tried to classify using only visual input. There are three parts in the proposed method. First is the "Hand Extraction", which is designed to find the hand from the image for finding the center of the hand and distribution of distance between center and edge pixel. This distribution is called as Hand Contour Vector set,

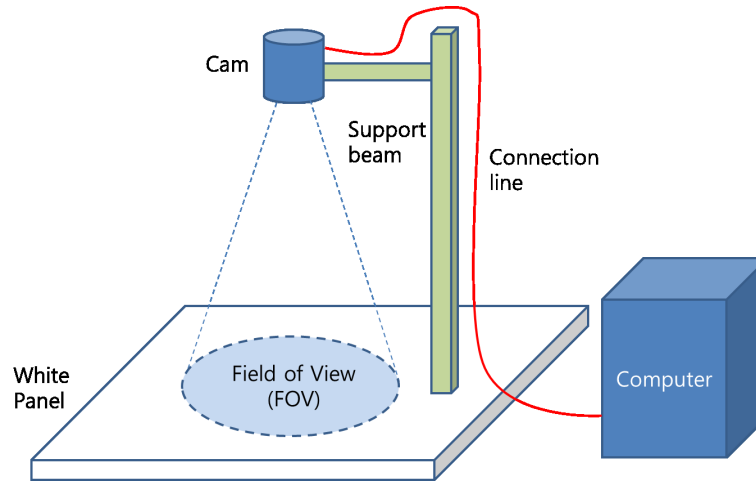


Figure 3.1: The structure of the experiment equipment.

shortly HCV. Then the images are changed into one HCV set. In the second part, the process of data collection progresses using the first one for making a reference data set. The third part is classification, where one HCV of current image is compared with HCV data set. From these processes, the system can classify the hand shape of different letters in the ASL alphabet. Then detail process of above three parts is checked.

### 3.1.1 Experiment environment

In the present experiment, only a personal computer and webcam (Logitech Webcam C160) (and human participants) are used. The computer has Intel(R) core(TM) i5-2500 CPU (3.30GHz quad core), and MATLAB 2011a is used. A white panel is used to find the hand portion. Then, the experiment is conducted without any occlusion. Figure 3.1 shows the experimental setting. In the experiment, a single right hand is positioned on the white panel, and then the region in the field of view (FOV) is checked and the images are transferred to the computer in real time. The computer then calculates the algorithm.

### 3.1.2 Hand information extraction

In this part, hand information is extracted from a single image. There are many mid-processes which are for locating the hand. The objective of this course is finding both the center of the hand and the distribution of pixels. Then, the size of images (input or reference image) are reduced to smaller ones.

### 3.1.2.1 Skin detection using color information

First of all, the pixels are collected using RGB color value. For one single picture from the camera, the RGB color value of all pixels is checked. Threshold values are set and pixels that are the candidates of pixels in the hand are identified. However, there are some two types of errors possible here. The first (type1) is the identification of other pixels which are not in hand but satisfy the threshold value. The other (type2) is pixels in the hand region that do not satisfy the threshold value. In Figure 3.3 (a) is the original image and (b) is the output of this process. There are two arrows which have some text, indicating type1 and type2 errors. Then these errors have to be compensated to improve the performance. In this part, RGB instead of YCbCr, HCV or L.a.b. color space is used, which are widely known as robust methods of skin detection. In this case, the hand from the image can be properly found without them in the restricted environment. However if there were some changes in environment and greater robustness of detection was needed, then the addition of these could be a solution.

### 3.1.2.2 Noise removal

The binary image is acquired from the first process, and moving window is used to eliminate both types of error. In this part, if the center is not detected but there are many pixels in the window, then it changes the center to be the center of the detected pixels. In addition, if the center is detected but there are few pixels in the window, then it changes the pixel to be a not-detected one. In Figure 3.2, the principle of this process can be shown. The window to all pixels is moved except for the border of the picture. Then the noise-removed images can be obtained.

Figure 3.3(c) shows that these errors are removed. There are entire hand pixels without any hole and the errors in the outside of the hand region, including the wrist part. The wrist can be eliminated by some using restriction on the environment, like wearing a long sleeve. However, since long sleeves cannot be assumed, the model which can be operated with or without the presence of a long sleeve delineating the wrists.

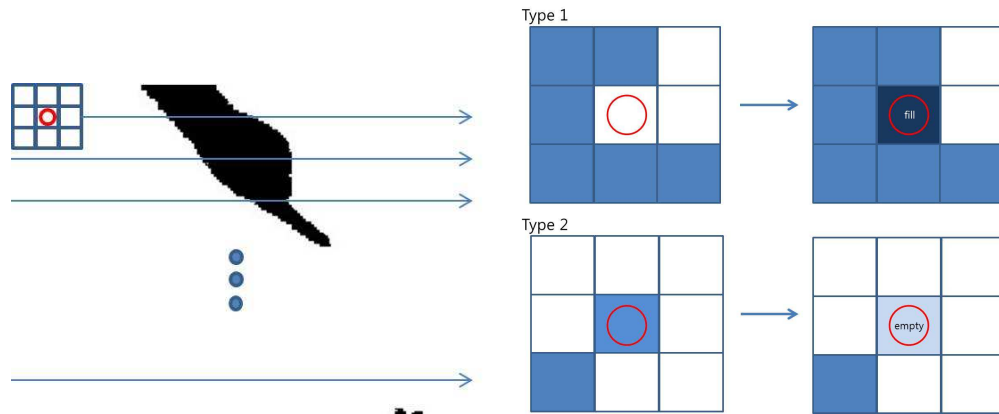


Figure 3.2: Principle of noise removal using moving window. The window which is represented the square with 9 empty space means the moving window with size 3. The red circle means the tested position. Then type 1 error is changed to be detected one which is represented by deep blue one and type 2 error is to be removed one which one is light blue.

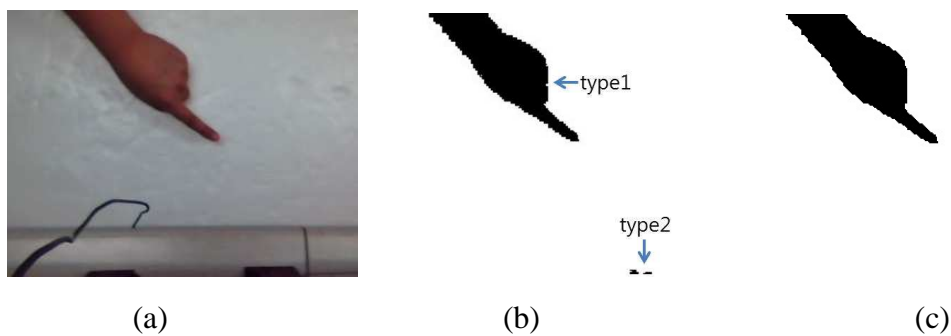


Figure 3.3: Example of hand pixel extraction. (a) is the original image. (b) is the skin detected binary image by using RGB color value. (c) is the image after noise removal.

### 3.1.3 Hand edge extraction

In this paper, the distribution of the outer edge of the hand is used. The edge of the hand is found by using a Canny Edge Filter. With proper coefficients, the edge can be found. To classify various hand shapes, this edge information is crucial. Therefore this edge information is saved for future processes. In Figure 3.4, the output of edge detection can be checked. (a) is the original image and (b) is the detected edge image.

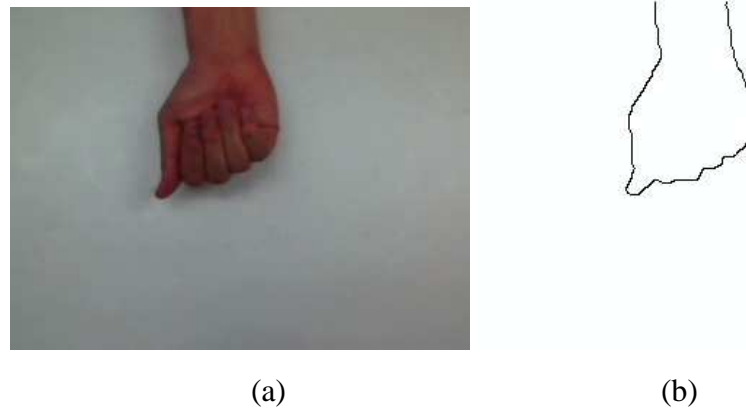


Figure 3.4: Example of edge detection by Canny Edge Filter. (a) is the original image and (b) is the edge image. The black line means the pixels on the edge.

#### 3.1.3.1 Making temporary center

The center of the hand have to be found to find the distribution. There are two steps in finding the center point. First is the traditional method, which has been used in other research. In this step, the center point is found by calculating the mean position of skin pixels. Then the distribution of the distances is calculated between temporary center and edge. Next, the mean of these distributions is saved into memory. The center point using mean of hand pixel is easily changeable by shape or the pixels on the wrist; therefore, the center point is updated using the mean value of temporary distribution (between edge image and temporary center) in the second step. This part is explained in the palm extraction subsection below.

#### 3.1.3.2 Finger part removal

The hand shape is largely determined by the finger gesture and these shapes affect the center point of the hand. Therefore, a method which can remove the finger part is used. In this method, another moving window is applied. This window is square with a hollow point in the center of it. In the Figure 3.5, there are some blue transparent hollow squares. This is the moving window. Then, the pixel on the center point of the mask is the target pixel, and the number of the edge pixels is counted in the blue region, which is the outside of the window without an inner section. If the target pixel is in the skin region and the number of the edge pixel is over the threshold value, then the target is decided in the finger part. In addition, if the target pixel is not in the skin region or if the target pixel is in skin region and there are few edge pixels, then it is



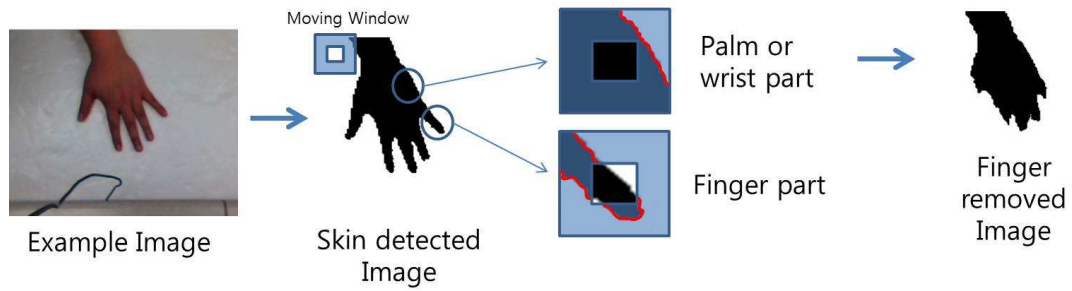


Figure 3.5: Principle of Finger Removal using another moving window. First is the example image and the second is skin detected binary image. Then palm or wrist part are collected using moving window and the last one is finger-removed image which is sum of pixels which is not finger parts.



Figure 3.6: Examples of finger-removed images. There are four set of image. Each set has two images. The left one is the skin detected image and the right one is the finger-removed image.

passed. Then the pixels can be cutout in the finger part.

Figure 3.5 shows the principle of this process and the finger-removed image in the right end of the figure. The finger is not perfectly removed, but the output does not include a significant part of the finger. Then some examples of finger removed images are shown as in Figure 3.6. The figure shows the output of the finger part removal. The upper two cases show successful finger removal, and lower two cases show some fails. However, the majority of the finger part is deleted.

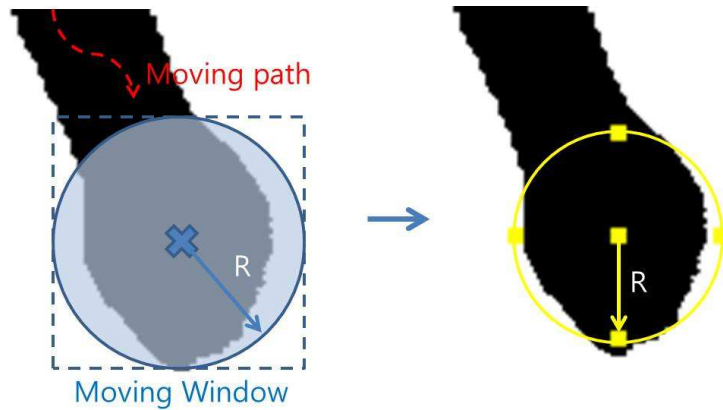


Figure 3.7: Principle of palm extraction.  $R$  is the double of the mean value of temporary distribution which is calculated in 'Finger Part Removal' section. Red dotted line means the path of moving window and the blue translucent circle means the window. The blue  $x$  means the center and the blue arrow represent the radius. The right one shows the output of the algorithm. The yellow circle means the window size and the yellow box in the center means the center point of the palm. The other four yellow boxes in the border of the circle means the end points of window.

### 3.1.3.3 Palm extraction and finding center of the hand

There are two information sets here: one is with finger and the other without finger. In either case, the palm have to be found to find the robust center of the hand. Therefore, the finger part is removed as in the preceding subsection. Then, the patterns can be observed in examples, as in Figure 3.6. As the wrist is thin, the finger-removed image has a relatively large palm attached to the wrist. Then, the palm is found using the appropriate large circle. Next, the mean value of temporary distribution is used, which is calculated in 'Finger Part Removal'. The radius is determined as double this value.

Figure 3.7 shows the principle of this part and its output using an example image. As mentioned in the caption of the figure, two facts can be shown. The first is that the size of the circle is good. The double of the mean value is operated as an appropriate radius value. The yellow circle includes almost part of palm. The other is the center point. The center point can be found using this circle, and the output is good. Therefore, the center point of other hand shapes is compared. In Figure 3.8, the center points are well-detected. In particular, the second case of this part represents the letter B in sign language. Then, as the four fingers are attached to each other, they are not eliminated by finger removal. However, it can also find the center position. In addition, the fifth

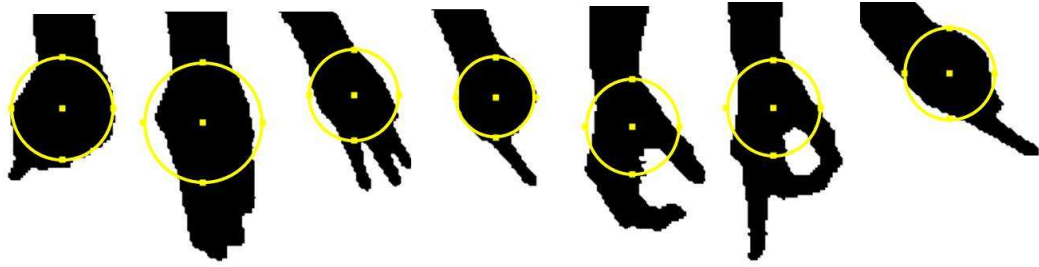


Figure 3.8: Examples of palm and center detected images. There are 7 example images with center. Yellow circle means palm position and the center of the circle is the calculated center of the hand.

and sixth are special cases in which the palm does not face the camera. However, they also operate well as center points in classification. In addition, the case is checked without wrist by using some clothes, in which the center is also found with ease.

#### 3.1.3.4 Hand Contour Vector(HCV) set extraction

Above this part, the center of the hand can be found. Returning to the whole edge image, the new center can be added and the new distribution can be calculated as the distance between center and edge. In this case, the distance is referred to as the Hand Contour Vector(HCV), and the distribution as the HCV Set. From the center, it tries to find the nearest pixel from the center at each angle. After this calculation, the HCVs can be collected in all angles to make the HCV set. Then these sets are used as classifiers. The image can be reduced into 360 numbers of HCV (one HCV set). If there is no pixel in some  $i$ -th angle, then the zero value is put in the  $i$ -th HCV.

Figure 3.9 shows the principle of this method and the HCV information. The first figure shows the extraction HCVs in each angle. 1-degree angle resolution is used. Then the 360 HCV numbers can be gotten, as in the second Figure. Finally, one hand image is changed into an omni-directional 360 number of HCVs.

#### 3.1.3.5 Median filter

Figure 3.9 shows that there are so many zero values between neighboring angles. They are generated by the angle resolution and detection. When the observation is zero, there are many omitted values. This is called the zero-back error. Then, the median filter is

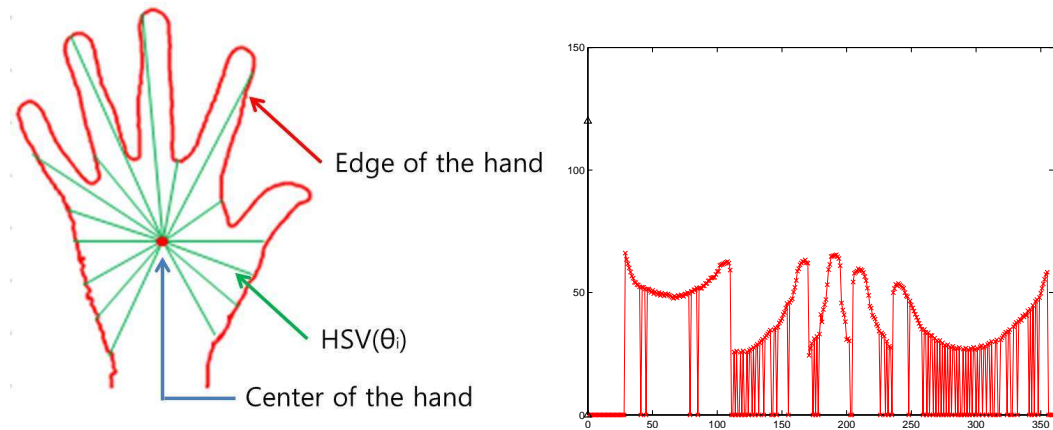


Figure 3.9: Principle of Hand Contour Vector Set Extraction. The left one include Center of the hand, Hand Contour Vectors and the edge of the hand. The right one show the distribution of the distance between center and edge which is equal to the length of HCVs.

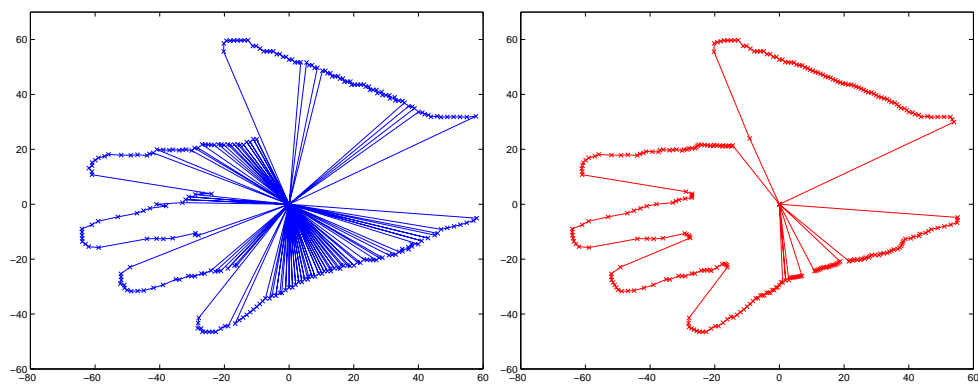


Figure 3.10: Examples of the median filter

used in the HCV set data. The median filter operates to pick up the middle value of neighboring values, except for itself, using a moving window that is 4pixels/1degree (2 pixels in left and 2 pixels in right). Then, the single zero-back errors can be modified. Figure 3.10 shows the effect of the median filter. The HCV set is rebuilt in 2D using both length and angle. There are many zero-back errors in some finger and outer palm in the left one. However, after using the median filter, there are some advances in this part. However, it can retain the wrist part which is true zero. Then, the modified HCV set can be collected, which has more stable distribution than the one without median filter from one image.

The courses, as mentioned above, are the single processes applied to single images. The single image from the camera is changed into the single HCV set.



Figure 3.11: Example images of 25 classes. Images from 1st to 24th are the alphabets except 'J' and 'Z' which has title label on each image . 25th image means 'Love'.

### 3.1.4 Reference data collection

Prior to classification, reference sets have to be made, which are used as classes in classification. The classes can be made with many different hand shapes. In the experiment, 24 sign language letters and one single word, "love" are used. As letters 'J' and 'Z' requires a sequence, they are not used. However, there are some differences between the images that are in equal classes. Repetitions of the same shape can have different configurations, and different people have different hand shapes. Therefore, an efficiently large reference set of each class have to be made to get high performance.

After getting images of all classes, they are changed into HCV sets and only remember HCV set information without images.

### 3.1.5 Classification

Having the reference HCV set, the current input is classified. Therefore, the current input is changed into an HCV set. Then, the proper class is found by matching it with the reference data.

#### 3.1.5.1 Normalization

After changing the current input, it has to be normalized. The input data is normalized with each reference datum. The mean value of each reference HCV set is counted, and the whole current distribution is changed with the ratio between two mean values.

$$\begin{aligned}
 HCVs_i &= \{(L_{1^\circ}, \theta_{1^\circ}), (L_{2^\circ}, \theta_{2^\circ}), \dots, (L_{359^\circ}, \theta_{359^\circ}), (L_{360^\circ}, \theta_{360^\circ})\} \\
 M_i &= \frac{\sum_{a=1^\circ}^{360^\circ} L_a}{\sum_{a=1^\circ}^{360^\circ} 1} \\
 HCVs'_i &= \{(L'_{1^\circ}, \theta'_{1^\circ}), (L'_{2^\circ}, \theta'_{2^\circ}), \dots, (L'_{359^\circ}, \theta'_{359^\circ}), (L'_{360^\circ}, \theta'_{360^\circ})\} \\
 M'_i &= \frac{\sum_{a=1^\circ}^{360^\circ} L'_a}{\sum_{a=1^\circ}^{360^\circ} 1} \\
 \text{Normalized\_Current\_HCV\_set} \\
 nHCVs'_i &= \frac{M_i}{M'_i} \times \{(L'_{1^\circ}, \theta'_{1^\circ}), (L'_{2^\circ}, \theta'_{2^\circ}), \dots, (L'_{359^\circ}, \theta'_{359^\circ}), (L'_{360^\circ}, \theta'_{360^\circ})\}
 \end{aligned}$$

Numerical formulas represent the normalization process. Each HCV set has 360 numbers of HCVs. Then, the mean of the length value (L) is calculated. Ration between two mean values is multiplied to current HCV set. Then the output is the solution of this course. Then, each input can be compared with reference HCV sets after normalization.

#### 3.1.5.2 HCV Set matching

Through the above steps, the HCV Sets can be collected as reference sets. In this step, the current data is tried to be matched with these reference sets. Briefly, the one current vector set is matched to the most similar one in the reference. In addition, it can be compared with the reference image set using all pixels in the image. However, this is not robust in either speed or performance. HCV model automatically extracts the hand information, and outputs for maintaining the information of hand are small. Each

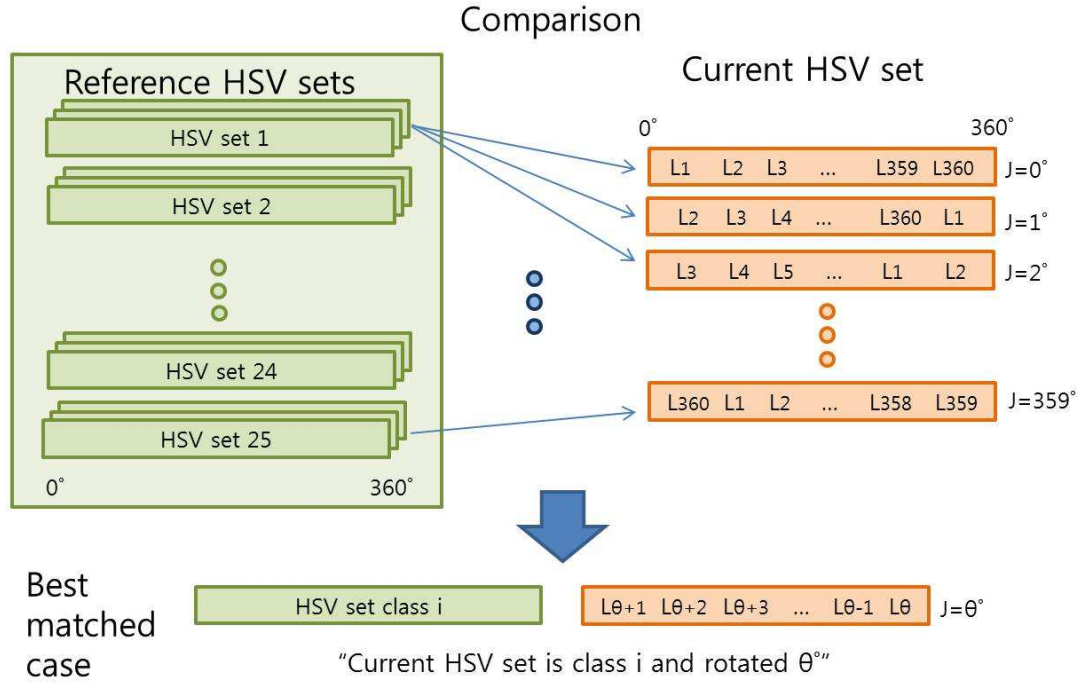


Figure 3.12: Principle of Hand Contour Vector Set Matching.

vector in the HCV set has both angle and length, and there are 360 vectors in one hand shape when 1-degree resolution is used. In a normal situation, if there are  $M$  classes and  $N$  reference in each class, the comparisons are done  $N \times M$  times. However, the angle variations have to be thought as the angle of the hand is not known and there are no processes to find it. Therefore, the comparison is done 360 times, at each angle variation, to find the angle. Then the full calculation is  $360 \times N \times M$ .

In each comparison, the sum of difference between two vectors in all angles is counted. In the first 360 times of calculation, two vectors that have the same angle are simply compared. Next, one angle of all vectors in the current set is changed and the comparison is done again. After finishing the calculation in all angles, then 350 differences are obtained. The most small difference case is the score of the matching in this class. This can identify the most matching score with the most matching angle.

$$nHCVs'_{i,j} = \frac{M_i}{M'_i} \times \{(L'_{1^\circ}, \theta'_{1^\circ} + j), (L'_{2^\circ}, \theta'_{2^\circ} + j), \dots, (L'_{359^\circ}, \theta'_{359^\circ} + j), (L'_{360^\circ}, \theta'_{360^\circ} + j)\}$$

$$\Delta j = 1^\circ \gg \theta'_{1^\circ} + j = \theta_{1^\circ + j}$$

$$MatchingScore_{i,j} = \sum_{i=1^\circ}^{360^\circ} \{L_i - \frac{M_i}{M'_i} \times L'_{i+j}\}$$

$$MatchingScore_i = \min(MatchingScore_{i,j})$$

In the formula,  $j$  is the angle variation and ' $j$ ' is changed from 1 degree to 360 degrees.

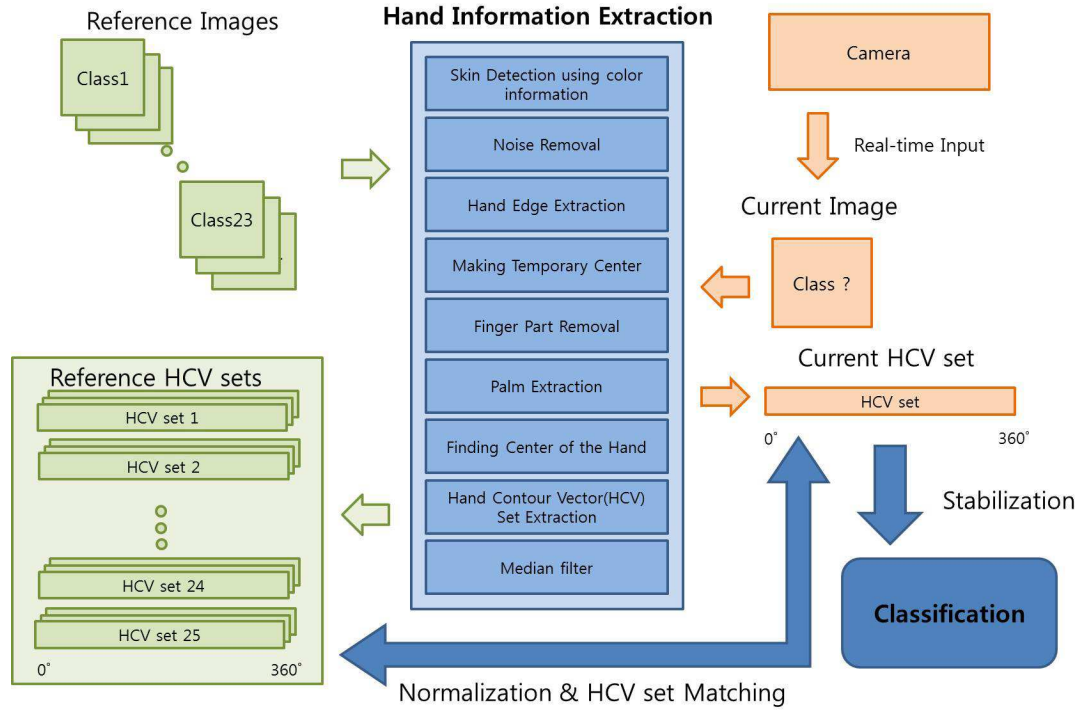


Figure 3.13: Overview of whole process

The smallest case ( $\text{MatchingScore}_i$ ) can be found as matching score of the class 'i'. Then, M number of Matching Scores can be collected with each optimal angle j, and the smallest one among them is found. This is the answer with both class i and angle j. Similarly, in Figure 3.12, this process can be checked. The right scarlet squares are the current HCV sets with angle variation (j degree), and they are matched with reference HCV sets. "Comparison" is matching process between current and reference. Finally the optimal matched case with class i and rotation angle j are obtained.

### 3.1.5.3 Stabilization

The proposed model operates in real-time. Thus, the data is obtained from the camera and the output is shown at each frame. However, there are some spark errors in the output sequence. Therefore, some 1st order low pass filters with 0.9 alpha value are added, which means that some 0.1 weighted number to the past calculated angle are added. Thus, some spark errors are prevented in the output sequence. After that, the past angle is updated using the current output.

$$\begin{aligned}\theta &= \alpha * \theta_{old} + (1 - \alpha) * \theta_{new} \\ \theta_{old} &= \theta\end{aligned}\tag{3.1}$$



In the above formula,  $\theta_{new}$  means the calculated mean and  $\theta_{old}$  is the past one. Then, the final output is the  $\theta$ , and the old one is updated using the final one.

## 3.2 Experiment

Experiments are performed to check the proposed model with finger-spelling alphabets. A lot of variations were tested with both reference sets and test sets: there were 4 participants with 25 classes and 120 images in each class (total 3000 images).

In the proposed method, the training phase is the same process as that used for making HCVs sets from the reference images. The processes to obtain HCVs sets from webcam input images and to compare these sets with reference HCVs set are test phases. Making a HCVs set process from an image, that is the training phase, takes 0.336911 seconds. To classify one input image from the 25 classes in the test phase takes 0.490244 seconds. Therefore, it can be considered that it can classify the static gestures in real time.

### 3.2.1 Using one reference set of a participator

In this part, the model is tested using only one reference set, which includes 25 classes with only one image in each class. Briefly, a reference set (finger-spelling alphabet data set) consists of one image for each class in 25 classes.

#### 3.2.1.1 Experiment with reference set

In this section, the algorithm is tested by using only a reference set. The images in the reference set are matched with reference set itself. In other words, a training set is used as a test set to check that the system works well.

Figure 3.14 shows the performance histogram of the experiment. The figure shows 100% success rates for all letters. It is obviously a reasonable output, because the algorithm is compared with the reference set and test set, and it computes the difference of each image. Therefore, if same image as the reference image is used, then it has perfect matching, so there is no difference between reference image and test image.

From this experiment, it can be confirmed that the proposed system works correctly.

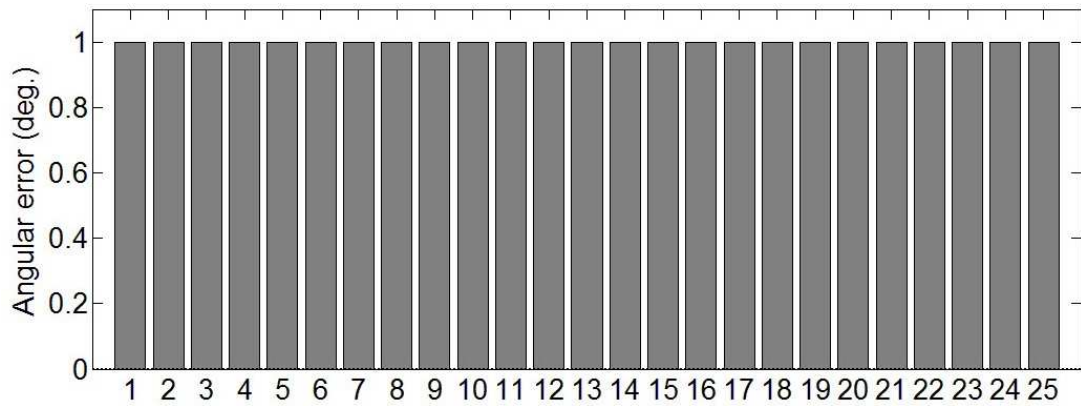


Figure 3.14: Test with same images which is used for reference set. X axis shows the class 1 25 and Y axis shows the success ratio from 0 to 1.

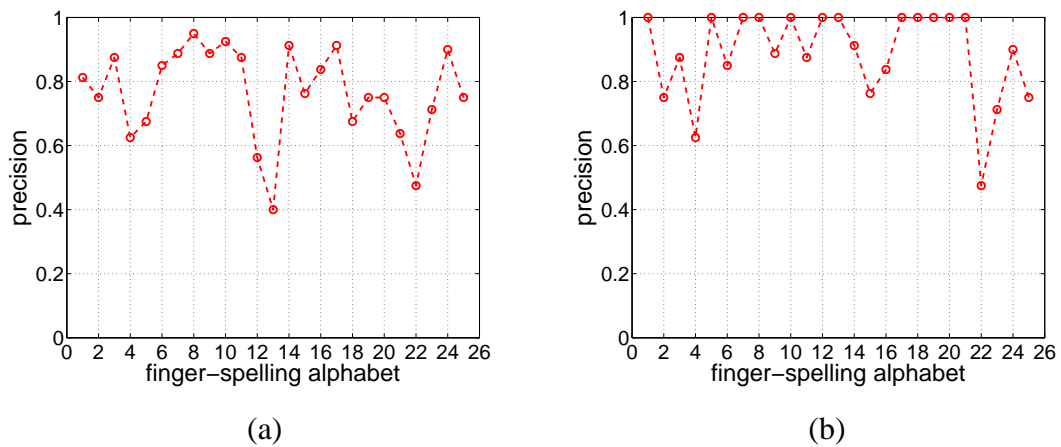


Figure 3.15: Test with various different input images of a participator who made a reference set. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together.

### 3.2.1.2 Experiment with various images of a participator

In this part, the output performances are checked using various inputs of the person who makes the reference set. The reference images used in this part are the same as the reference set used in above experiment. In other words, 25 reference images, which include one image for each class, are used, and 2000 images that are made by the person who made the reference set are used to test the algorithm.

Figure 3.15(a) shows the precision rate of this experiment. The letters that have the distinct shape features of the hand show good performances. There are 11 letters that show more than 80% recognition rates: A, C, F, G, H, I, K, O, Q, R, and Y. Mean

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Love
A	0.8125	0	0	0	0	0	0	0	0.0125	0	0	0	0	0	0	0	0	0.0875	0.0625	0	0	0	0.025	0	0
B	0	0.75	0	0	0	0	0.1875	0	0	0	0	0	0	0	0	0	0.0375	0	0	0.025	0	0	0	0	0
C	0	0	0.875	0.025	0	0	0	0	0	0	0	0	0.0125	0.0875	0	0	0	0	0	0	0	0	0	0	0
D	0.0125	0	0	0.625	0.0125	0	0	0	0	0	0	0	0	0.2875	0	0	0	0.0125	0.0125	0	0	0	0.0375	0	0
E	0.0125	0	0	0	0.675	0	0	0	0.025	0	0	0.15	0.05	0	0	0	0.0375	0.025	0	0	0	0	0.025	0	0
F	0	0.1375	0	0	0	0.85	0	0	0.0125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0.8875	0.1125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0.05	0.95	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0.0125	0	0	0	0.05	0	0	0	0.8875	0	0	0.0125	0	0	0	0	0	0.0375	0	0	0	0	0	0	0
K	0	0	0	0	0	0	0	0	0	0.925	0	0.0125	0.0125	0	0	0	0	0	0	0.0375	0.0125	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	0.875	0	0	0	0	0	0	0	0	0	0	0	0	0	0.125
M	0	0.0125	0	0	0.175	0	0	0	0	0	0.5625	0.125	0	0	0	0	0.05	0.0375	0	0	0	0.0375	0	0	0
N	0	0	0	0	0.1125	0	0	0	0	0	0	0.25	0.4	0.9125	0	0	0	0.025	0.1375	0	0	0	0.075	0	0
O	0	0	0	0.05	0.025	0	0	0	0	0	0.0125	0	0.0125	0	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0.025	0.0125	0.1	0	0	0	0	0	0	0	0.7625	0.1	0	0	0	0	0	0	0	0
Q	0	0.025	0	0	0	0	0.0875	0	0	0	0	0	0	0.05	0.8375	0	0	0	0	0	0	0	0	0	0
R	0	0.0375	0	0	0	0	0	0	0	0	0	0.0125	0	0	0	0	0.9125	0	0	0.025	0	0	0.0125	0	0
S	0.1625	0	0	0	0	0	0	0.025	0	0	0.0125	0.025	0	0	0	0	0.875	0.075	0	0	0	0.025	0	0	0
T	0.1	0	0	0	0	0	0	0	0	0	0	0.05	0	0	0	0	0.0875	0.75	0	0	0	0.0125	0	0	0
U	0	0.0125	0	0	0	0.0125	0	0	0	0.0125	0	0	0.0375	0	0	0.125	0	0.0125	0.75	0.025	0	0.0125	0	0	0
V	0	0.0125	0	0.0125	0	0	0	0	0	0.1125	0	0.0125	0.0125	0	0	0	0.0125	0.0125	0.025	0.0875	0.6375	0.0375	0.025	0	0
W	0.0125	0.025	0	0.025	0.0375	0	0	0	0	0.0625	0	0.0625	0	0	0	0.0125	0.075	0	0.05	0.15	0.475	0.0125	0	0	0
X	0.05	0	0	0	0	0	0	0	0	0	0.0625	0	0	0	0	0	0.0625	0.1	0	0.0125	0	0.7125	0	0	0
Y	0.025	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0125	0	0	0	0	0.9	0.0625	0
Love	0.0125	0	0	0	0	0	0	0	0	0	0.0875	0	0	0	0	0	0	0	0.0125	0	0	0	0.1375	0.75	0

Figure 3.16: The perception ratio for each input letters

success rate is 76.6 %. Figure 3.16 shows the perception ratio for each input letter. For instance, the letter 'A' could be given as input, while the system may occasionally perceive it as some other letter. The diagonal values of Figure 3.16 indicate the correct recognition of a finger-spelling letter. Other values except for diagonal terms, denote the misclassification. However, there are many hand shapes that have the similar forms to each other. For example, A, E, M, N, S, and T are similar to each other. All of these letters' shapes are the form of fist. Careful observation is required to distinguish these alphabets. It mainly depends on the position of the thumb. These differences are hard to distinguish, even by a person. Therefore, it is tried to group together those that have a similar shape of hand. Figure 3.17 shows the precision rate of these letters. If these letters are grouped together, then it shows more than 95% recognition rates, so A, E, M, N, S, and T, and (R,U), (G,H), (K,V) were grouped. Figure 3.15(b) shows the recognition rate when some classes are grouped together. In this case, if r was estimated as u, it is also recognized. When r and u are grouped, the recognition rate of r is 93.75% and u is 87.5%. In case of g and h, both recognition rates are 100%. The recognition rate of k is 96.25%, and v is 75%. The mean precision rate of this experiment with grouping is 88.85%.

### 3.2.1.3 Experiment with input images of several participants

In this part, the output is checked using the input images of three people with a reference image that is equal to the one used in the above part. The experiments are performed whether the proposed model can be used for other person who did not make the reference set. The algorithms do comparison reference image and test image directly, so it can be presumed that if there are not any reference set of the participant

	A	E	M	N	S	T	precision
A	0.8125	0	0	0	0.0875	0.0625	0.9625
E	0.0125	0.675	0.15	0.05	0.0375	0.025	0.95
M	0	0.175	0.5625	0.125	0.05	0.0375	0.95
N	0	0.1125	0.25	0.4	0.025	0.1375	0.925
S	0.1625	0	0.0125	0.025	0.675	0.075	0.95
T	0.1	0	0	0.05	0.0875	0.75	0.9875

Figure 3.17: The perception ratio for A, E, M, N, S, and T. The precision in last column shows the performances with grouping.

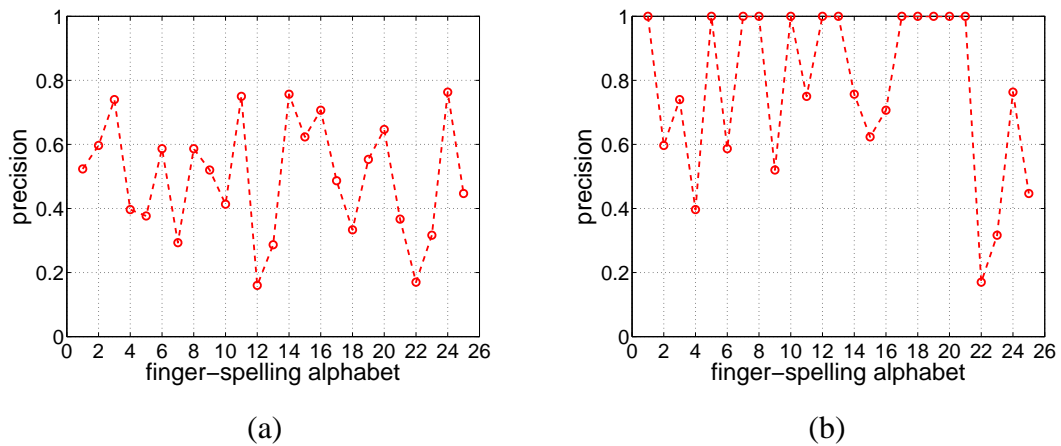


Figure 3.18: Test with various different input images of participators who didn't make a reference set. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together.

making the test images, then it shows worse performance than the above experiments.

Figure 3.18(a) shows the result of this experiment. As shown the figure, the experiment with test set of a participant who makes the reference set performs worse. Its mean success rate is 49.6%. This is because peoples hand dimensions and finger-spelling styles differ. However, if similar shapes of alphabets are grouped, then success rates are also increased, as in Figure 3.18(b). In this case, mean rate of recognition is 77.49%. Figure 3.20 shows the recognition rate of A, E, M, N, S, and T, and it almost has 90% precision rate. Even if there is no reference set of the person who makes input images as a test set, the performance of this experiment is not bad. This means that the systems are effective for classifying the shape of hand.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Love
A	0.52333	0	0	0	0.00333	0	0	0	0.00333	0	0	0	0	0	0	0	0	0	0.09333	0.32333	0	0	0	0.05	0.00333	0
B	0	0.59667	0	0	0.00667	0.22	0	0	0	0	0	0	0	0.03333	0	0	0	0.05	0	0	0.09	0	0	0.00333	0	0
C	0	0	0.74	0.00667	0.00333	0	0.04667	0	0	0	0	0.00333	0.02667	0.11	0	0	0.02	0.00333	0.00667	0.02333	0.00667	0	0.00333	0	0	0
D	0.02333	0	0	0.39667	0.06	0	0	0	0	0	0	0.01667	0.02667	0.25667	0	0	0	0.04333	0.07333	0	0	0	0.10333	0	0	0
E	0.01	0.01333	0	0	0.37667	0.00333	0	0	0.00667	0	0	0.09333	0.11333	0.01333	0	0	0	0.13333	0.18667	0	0	0	0.05	0	0	0
F	0.00667	0.28667	0	0	0.1	0.58667	0	0	0.00667	0	0	0.00333	0.02333	0	0	0	0.06667	0.00333	0	0.00667	0	0	0	0	0	0
G	0	0	0	0	0.00667	0	0.29333	0.40667	0.02	0	0	0.03333	0.10667	0.01	0.09667	0	0.01667	0.00333	0.00667	0	0	0	0	0	0	0
H	0	0	0.01	0	0	0	0.30667	0.58667	0.02333	0	0	0	0	0.03333	0.05	0.02	0	0	0	0	0	0	0	0	0	0
I	0.02	0	0	0	0.06333	0	0.00333	0.01	0.52	0	0	0.03333	0.00667	0.00333	0	0	0.12667	0.10333	0	0	0	0	0.11	0	0	0
J	0.00333	0.00667	0	0	0.05	0	0	0	0.41333	0	0.07333	0.05	0	0	0	0	0.02333	0	0.03667	0.11	0.13	0.01667	0.08667	0	0	0
K	0.02333	0	0	0	0.00333	0.00333	0	0	0	0	0.75	0	0.01333	0	0	0	0	0	0.00667	0.00667	0	0	0.02333	0.05333	0.11667	0
L	0.01	0.01667	0	0	0.24333	0	0	0	0.00667	0	0	0.16	0.16667	0	0	0	0	0.16667	0.22	0	0	0	0.07	0	0	0
M	0.00333	0.02333	0	0	0.11	0.01	0	0	0	0	0	0.11333	0.28667	0	0	0	0.04667	0.37	0.00667	0	0	0.03	0	0	0	0
N	0	0	0	0.09	0.04333	0	0	0	0	0	0	0.02	0.02667	0.75667	0	0	0	0.04667	0.16667	0	0	0	0	0	0	0
O	0.00667	0.00667	0.00333	0	0	0.01333	0.13667	0.06667	0.00333	0	0	0	0	0.02	0.62333	0.11333	0	0	0.00333	0	0	0	0	0.00333	0	0
P	0	0.01667	0	0	0	0.05333	0.07	0.00667	0	0	0	0	0	0.02667	0.10667	0.70667	0	0	0.00333	0	0	0	0.01	0	0	0
Q	0	0.05667	0	0	0	0.10667	0	0	0	0.00333	0	0	0.07	0	0	0	0	0.48667	0	0.02667	0.24	0	0	0.01	0	0
R	0.07	0	0	0.00333	0.08667	0	0	0	0.02	0	0	0.01	0.02333	0.09667	0	0	0	0	0.33333	0.39667	0	0	0	0.08	0	0
S	0.13333	0	0	0	0.02333	0	0	0	0	0	0.00667	0.11333	0	0	0	0	0.07667	0.55333	0	0	0	0	0.09333	0	0	0
T	0	0.05667	0	0	0.01333	0.00333	0	0	0	0.02	0	0	0.01333	0	0	0	0.16333	0	0.01333	0.46667	0.01	0.01	0.05	0	0	0
U	0	0.00667	0	0	0	0	0	0	0	0.24	0	0.03667	0.03	0	0	0	0.01667	0	0.04333	0.12	0.36667	0.02667	0.11333	0	0	0
V	0.01333	0.09333	0	0.01667	0.05333	0.04	0	0	0	0	0.09333	0	0.03667	0.05	0	0	0.01333	0.14	0.11333	0.03333	0.09333	0.17	0.04	0	0	0
W	0.06667	0	0	0.02667	0	0	0	0	0.00333	0.00333	0	0.00667	0.08667	0	0	0	0.00333	0.09333	0.38333	0.00667	0.00333	0	0.31667	0	0	0
X	0.08333	0	0	0	0	0	0	0	0	0	0.02667	0	0	0	0	0	0	0.01	0.01	0	0	0	0	0.76333	0.10667	0
Y	0.02333	0	0	0	0.00333	0.00333	0	0	0.01333	0.00667	0.22333	0.00333	0	0	0	0	0	0.00667	0.00667	0.01	0	0	0.01	0.24333	0.44667	0

Figure 3.19: The perception ratio for each input letters

	A	E	M	N	S	T	precision
A	0.523333	0.003333	0	0	0.093333	0.323333	0.943333
E	0.01	0.376667	0.093333	0.113333	0.133333	0.186667	0.913333
M	0.01	0.243333	0.16	0.166667	0.106667	0.22	0.906667
N	0.003333	0.11	0.113333	0.286667	0.046667	0.37	0.93
S	0.07	0.086667	0.01	0.023333	0.333333	0.336667	0.86
T	0.133333	0.023333	0.006667	0.113333	0.076667	0.553333	0.906667

Figure 3.20: The perception ratio for A, E, M, N, S, and T. The precision in last column shows the performances with grouping.

### 3.2.2 Using large reference set of a participant

In this experiment, larger reference sets are made than in the first experiments by taking five images for each class. The reference sets are made by one participant. It is expected that large reference set will improve performance with letters in same class but with slightly different shapes. Therefore each of the five different images in each class have slightly different forms of hand.

#### 3.2.2.1 Experiment with input images of a participant

The proposed model is tested using a large reference set with many test images from one participant who makes the reference set. It is a similar experiments to that in Subsection 3.2.1.2, however there are many images in each class for reference images. 2500 images made by a person who made the reference set are used to test the algorithm.

Figure 3.21(a) shows the recognition rate of this experiment. Mean success rate is

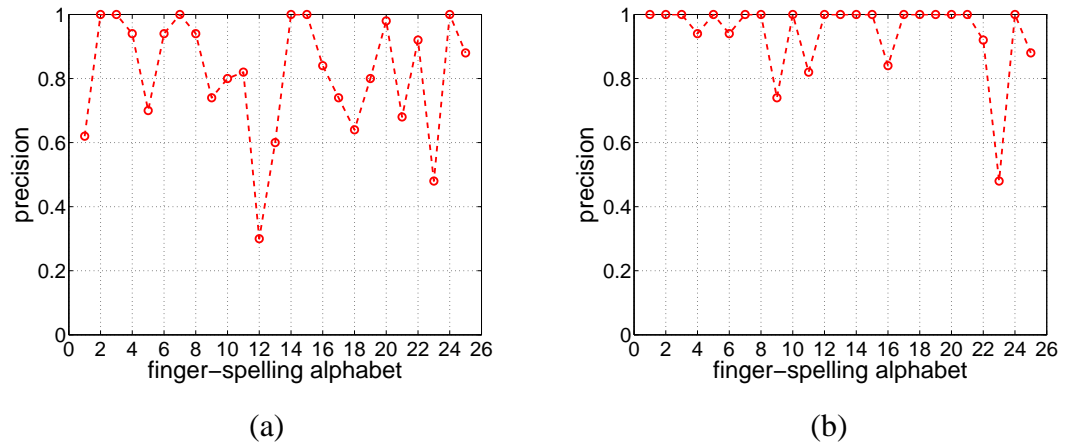


Figure 3.21: Test with various different input images of a participant who made reference sets and large reference set which is consisted of 5 images for each class. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together.

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Love
A	0.54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.24	0.18	0	0	0	0.04	0	0
B	0	0.94	0	0	0	0.03	0	0	0	0	0	0	0	0	0	0	0	0.03	0	0	0	0	0	0	0
C	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	0.97	0	0	0	0	0	0	0	0	0	0	0.03	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0.7	0	0	0	0	0	0	0	0.13	0.03	0	0	0	0.1	0.04	0	0	0	0	0	0
F	0	0.03	0	0	0	0.85	0	0	0.02	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0.5	0.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0.02	0.97	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0
I	0.02	0	0	0	0.02	0	0	0	0.86	0	0	0	0	0	0	0	0	0.05	0.03	0	0	0	0	0.02	0
K	0	0.01	0	0.01	0	0	0	0	0	0.46	0	0.02	0.07	0	0	0	0.16	0	0	0	0.25	0.02	0	0	0
L	0.01	0	0	0	0	0	0	0	0	0.8	0.01	0	0	0	0	0	0	0.05	0.01	0	0	0	0.1	0	0.02
M	0	0	0	0.32	0	0	0	0	0	0.37	0.25	0	0	0	0	0	0.01	0.05	0.01	0	0	0	0	0	0
N	0	0	0	0	0.12	0	0	0	0	0	0.12	0.62	0	0	0	0	0.01	0.09	0	0	0	0.04	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0.99	0	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.08	0.92	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0.07	0	0	0	0.87	0.02	0	0.04	0	0	0	0	0
S	0.01	0	0	0.07	0	0	0	0.07	0	0	0.07	0.01	0	0	0	0	0	0.59	0.18	0	0	0	0	0	0
T	0.04	0	0	0	0	0	0	0	0	0.02	0.21	0	0	0	0	0	0.08	0.62	0	0	0	0.03	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.22	0	0.77	0.01	0	0	0	0	0
V	0	0	0	0	0	0	0	0	0.14	0	0.06	0.01	0	0	0	0	0.13	0	0.09	0.57	0	0	0	0	0
W	0	0	0	0.01	0	0	0	0	0	0	0	0	0.01	0	0	0	0.01	0	0.01	0.03	0.93	0	0	0	0
X	0	0	0	0.01	0	0	0	0	0	0	0	0.02	0.37	0	0	0	0	0.03	0.21	0	0	0	0.36	0	0
Y	0.05	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05	0	0	0	0	0.89	0.01	0
Love	0	0	0	0	0	0	0	0	0	0	0.16	0	0	0	0	0	0	0.02	0	0	0	0	0.1	0.72	0

Figure 3.22: The perception ratio for each input letters

	A	E	M	N	S	T	precision
A	0.54	0	0	0	0.24	0.18	0.96
E	0	0.7	0.13	0.03	0.1	0.04	1
M	0	0.32	0.37	0.25	0.01	0.05	1
N	0	0.12	0.12	0.62	0.01	0.09	0.96
S	0.01	0.07	0.07	0.01	0.59	0.18	0.93
T	0.04	0	0.02	0.21	0.08	0.62	0.97

Figure 3.23: The perception ratio for A, E, M, N, S, and T. The precision in last column shows the performances with grouping.

75.24%. Its mean performances are not better than the performance in Subsection 3.2.1.2; however, there are many more letters that show more than 85% success rate. In addition, when the letters with similar shapes are grouped together, then the recognition

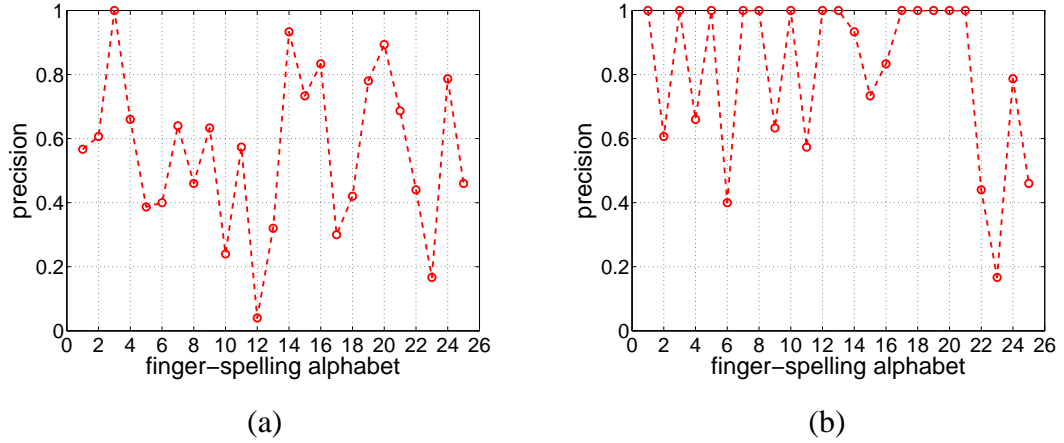


Figure 3.24: Test with various different input images of participators who made reference sets. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together.

rate is increased to 92.92%. In particular, when A, E, M, N, S, and T are grouped, the recognition rates are more than 96%, as shown in Figure 3.23. It also shows better performances than the performances when single reference set is used. When R and U are grouped, the recognition rate of R is 91% and U is 99%. In case of G and H, recognition rate of G is 100%, and H is 99%.

### 3.2.2.2 Experiment with input images of several participant

The model is tested using a large reference set with many test images from the other three participant who did not make the reference.

Figure 3.24(a) shows the performance of this experiment. The mean performance is 55.84%. When the similar letters are grouped together, then the mean performance is 80.91%. This experiment is similar with the experiment of section 3.2.1.3. Therefore, it also shows similar results; however, its performances are improved, because it has more reference sets to compare with the input images.

### 3.2.3 Using several reference sets of several participant

In this part, several reference sets are made with four persons. 500 reference images are made that consist of 20 images total for each class. There are four participants, so there are five images from each participant for each class. After that, the 2500

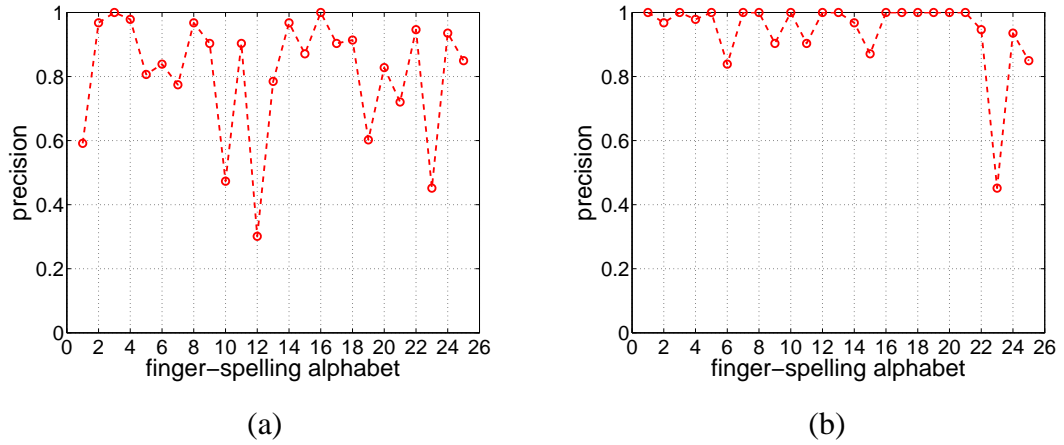


Figure 3.25: Test with various different input images of participators who didn't make reference sets. (a) The recognition ratio of each class. (b) The recognition ratio when similar shapes of hand are grouped together. Mean success rate = 81.12 % (before grouping), Mean success rate = 94.45 % (after grouping)

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Love
A	0.5914	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.39785	0.01075	0	0	0	0	0	0
B	0	0.96774	0	0	0	0.03226	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	0.97849	0	0	0	0	0	0	0	0	0	0.02151	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0.80645	0	0	0	0	0	0	0	0.04301	0	0.02151	0	0	0	0.11828	0.01075	0	0	0	0	0
F	0	0.05376	0	0	0.03226	0.83871	0	0	0.05376	0	0	0	0	0	0	0	0.02151	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0.77419	0.22581	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0.02151	0.96774	0	0	0	0	0	0	0.01075	0	0	0	0	0	0	0	0	0	0
I	0.01075	0	0	0	0	0	0	0	0.90323	0	0	0	0	0	0	0	0	0.08602	0	0	0	0	0	0	0
K	0	0.01075	0	0.01075	0	0	0	0	0	0.47312	0	0	0.03226	0	0	0	0.04301	0	0	0.08602	0.32258	0.01075	0.01075	0	0
L	0	0	0	0	0	0	0	0	0	0	0.90323	0	0	0	0	0	0.01075	0.03226	0	0	0	0	0.05376	0	0
M	0	0	0	0	0.30108	0	0	0	0	0	0	0.30108	0.31183	0.03226	0	0	0	0.05376	0	0	0	0	0	0	0
N	0	0	0	0	0.05376	0	0	0	0	0	0	0.06452	0.78495	0	0	0	0	0	0.09677	0	0	0	0	0	0
O	0	0	0.02151	0.01075	0	0	0	0	0	0	0	0	0	0.96774	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0.05376	0.06452	0	0	0	0	0	0	0.87097	0.01075	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0.01075	0	0	0	0	0	0	0.06452	0	0	0	0.90323	0.02151	0	0	0	0	0	0	0
S	0.04301	0	0	0	0.01075	0	0	0	0	0	0	0	0.03226	0	0	0	0	0.91398	0	0	0	0	0	0	0
T	0.04301	0	0	0	0	0	0	0	0	0	0	0	0.15054	0	0	0	0	0.2043	0.60215	0	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.16129	0	0	0.82796	0.01075	0	0	0	0
V	0	0	0	0.01075	0	0	0	0	0	0.08602	0	0.01075	0	0	0	0	0.04301	0	0.12903	0.72043	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0.01075	0	0	0	0	0	0	0	0	0.01075	0.03226	0.94624	0	0	0	0
X	0	0	0	0	0	0	0	0	0	0	0	0	0.41935	0	0	0	0	0.07527	0.05376	0	0	0	0.45161	0	0
Y	0.05376	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01075	0	0	0	0	0	0.93548	0
Love	0	0	0	0	0	0	0	0	0	0	0.09677	0	0	0	0	0	0	0.02151	0	0	0	0	0.03226	0.84946	0

Figure 3.26: The perception ratio for each input letters

different input images of four participants who made the reference sets for the test set were used. These images are slightly different from each other. Mean success ratio is 87.44%. This shows better performances than the performances with only the reference set. When the hand shapes that have similar forms are grouped together, the precision ratio is increased to 96.72%. Therefore, it is confirmed that the proposed method works well when the participants save their hand images to reference sets. If there are more reference images from the same participant, then its precision ratio will be increased.

In this experiment, there are some problems with specific persons. There are some good cases with high performances and other cases with low performances. Thus, the output of each person is shown.



	A	E	M	N	S	T	precision
A	0.591398	0	0	0	0.397849	0.010753	1
E	0	0.806452	0.043011	0	0.11828	0.010753	0.978495
M	0	0.301075	0.301075	0.311828	0.053763	0	0.967742
N	0	0.053763	0.064516	0.784946	0	0.096774	1
S	0.043011	0.010753	0	0.032258	0.913978	0	1
T	0.043011	0	0	0.150538	0.204301	0.602151	1

	R	U	precision
R	0.903226	0	0.903226
U	0.16129	0.827957	0.989247

	G	H	precision
G	0.774194	0.225806	1
H	0.021505	0.967742	0.989247

	K	V	precision
K	0.473118	0.322581	0.795699
V	0.086022	0.72043	0.806452

Figure 3.27: The perception ratio for group 1 (A, E, M, N, S, and T), group 2 (R and U), group 3 (G and H) and group 4 (K and V). The precision in last column shows the performances with grouping.

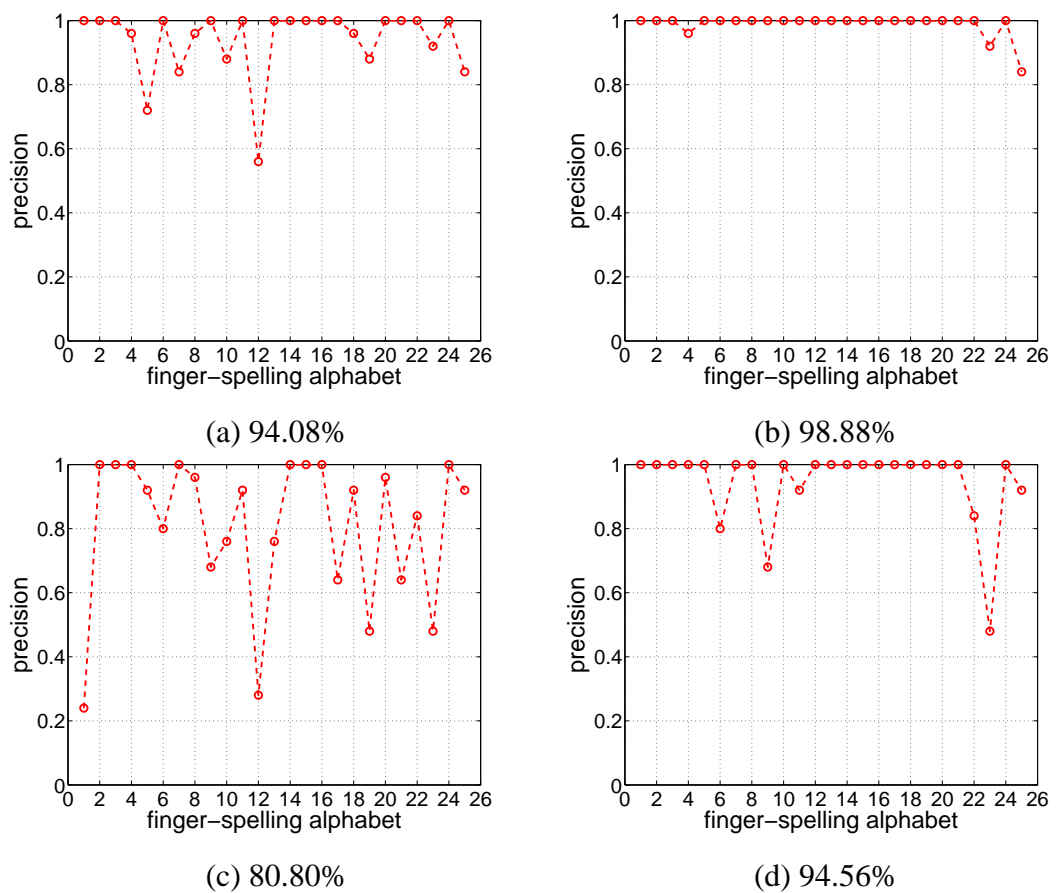


Figure 3.28: Outputs of high performance participants. There are two participants (two rows). Left ones (a and c) are the output without grouping and the right ones (b and d) are with grouping.

This dissertation shows the 81.12% success rates for the test sets when all 25 classes as static gestures in groups of similar shapes are used. If the success rates include the training sets, the performances are improved to 87.44% as shown in the results.

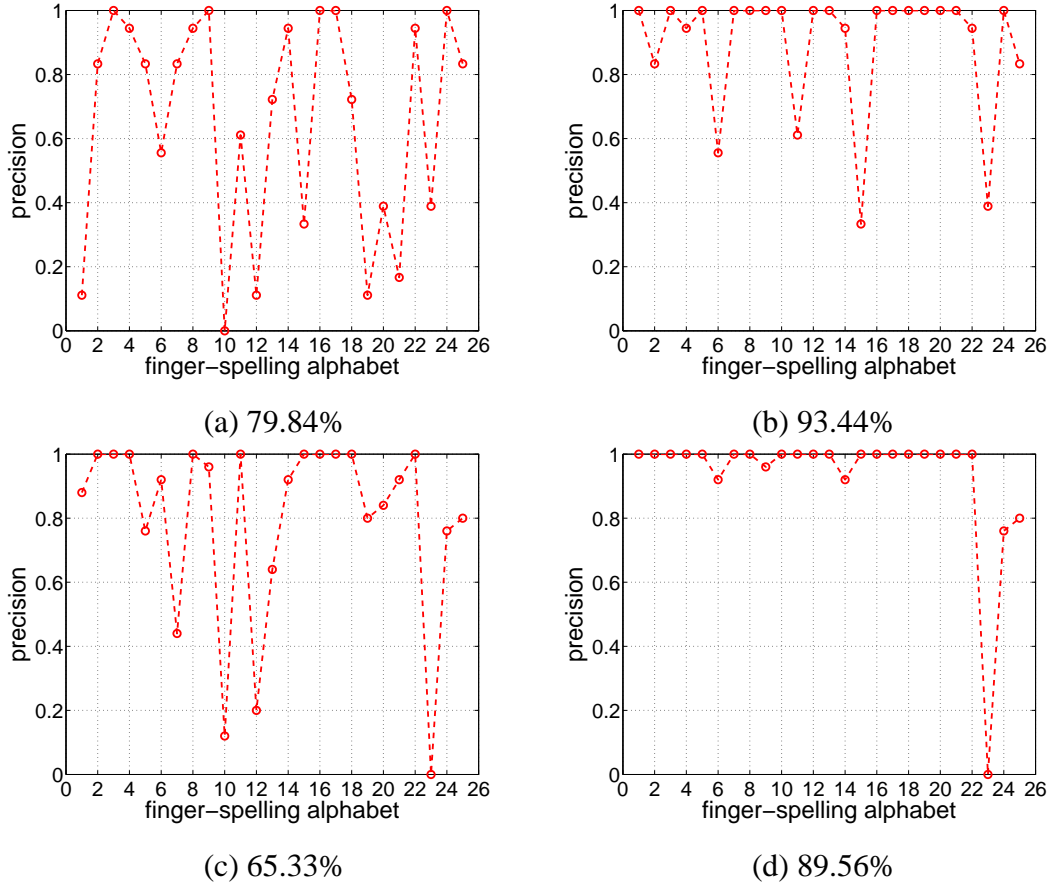


Figure 3.29: Outputs of low performance participants. There are two participants (two rows). Left ones (a and c) are the output without grouping and the right ones (b and d) are with grouping.

Pugeault and Bowden (2011) shows the 75% success rates to classify the 24 ASL finger-spelling letters without 'J' and 'Z'. They used Kinect to use depth information of images with standard intensities of images. As using Kinect, the hand information can be extracted in cluttered background in contrast with the proposed method. However, the proposed method shows the better performances. It also shows the better results to classify similar shaped gestures, such as 'A', 'E', 'M', 'N', 'S', and 'T'.

When some of the ASL finger-spelling letters are grouped for using the real-life applications, the proposed method shows the 94.45% success rates for the test sets. This is improved to 96.72% with training sets. Van den Bergh and Van Gool (2011) shows the 99.54% success rates; however, they only used the six key hand postures such as open hand, fist, pointing up, L-shape, pointing at the camera, and thumb up with RGB camera and ToF camera which can obtain the depth information. Munib et al. (2007) shows the 80% success rates with test sets, and it is improved to 92.33% with training

sets. They used Hough transform to obtain the silhouette of the hands including the inside of hands. Using Hough transform, they can classify the similar shaped gestures; however, they only used 14 ASL finger-spelling letters from the 20 available. From comparison with several studies, it can be confirmed that the proposed method in this dissertation shows good performance using only a cheap webcam in real-time.

Table 3.1: The performance comparison with other methods

	Precision	Notes
The proposed method	81.12% (total 87.44%)	25 ASL finger-spelling alphabets without 'J' and 'Z'
Pugeault and Bowden (2011)	75%	24 ASL finger-spelling alphabets without 'J' and 'Z'
The proposed method	94.45% (total 96.72%)	17 static gestures with grouping
Van den Bergh and Van Gool (2011)	99.54%	Six hand postures
Munib et al. (2007)	80% (total 92.33%)	14 ASL finger-spelling alphabets among 20 ASL signs

### 3.2.4 Estimation of orientation angle

Unlike other hand recognition methods, the model can estimate the orientation angle of the hand. Therefore, the performance of angle estimation is tested. The environment is set with a protractor for angle measurement, and the hand is moved from 0-180 degree. Photos are taken at 20-degree intervals and the hand's location is recorded. Then, the estimated angle can be found. Figure 3.30 shows the output of this experiment. The blue line shows the ideal case and the red line is the experiment output. Output is well-matched to ideal case; thus, the algorithm can estimate both angle and type of hand.

## 3.3 Discussion

There are similar letters which can result in mistakes in classification like R and U. There are four groups. Group 1 has A, E, M, N, S, and T, which are fist shapes. They look similar, although there are some little differences of thumb positions. Therefore, the proposed model, which uses contour of hand, should be weak in these parts and show poor performance in the results. However, if the hand shape is made strictly the same, then the right class using one hand can be found. Group 2 has K and V. They are confused by the similar location of the thumb. Group 3 has G and H, Group 4 has R and U. They are similar in boundary. Thus, if they are used in a real situation, some signs are decided, which are not in the same group and they can be easily classified over 90%.

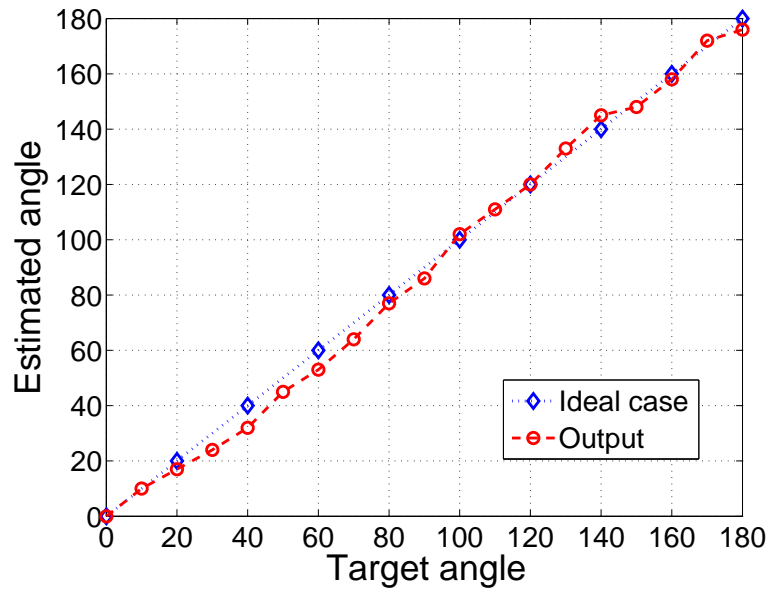


Figure 3.30: Angle estimation experiment. X-axis is target angle and the estimated output in Y-axis. Blue line shows the ideal case and the red line shows the output.

In addition, the systems show good accuracy for different people, even if the shapes are similar; however, the recognition ratio of letter 'X' is poor in every experiment. The reasons are analyzed, and it is confirmed that it is recognized as other letters that have the form of a fist, such as A, M, N, S, and T in many cases. Most often, it is considered same as letters T or N. In the result parts, X was not grouped with other letters that have the form of fist, because it is not considered as a similar shape. However, if X is grouped with other fist shaped letters, then its recognition rate is increased to almost 98%. If X is grouped with T and N, its recognition rate is increased to about average, 87.995%.

The better result can be obtained than the others. In this part, a large reference set is made using participants who are testers. Next, the algorithm is tested using both new image input and references. The output is different with each person's characteristics, and the best case shows 94.08% without grouping and 98.88% with grouping. Outputs of participants are shown and the individual differences are observed.

It is confirmed that the systems can classify well when the shapes of classes are clearly different. Sign language is used as tools of the experiment. Therefore, the successful classification can be done using some designated signs (i.e., those that are good in the above experiments). To successfully distinguish the similar hand shapes, much more image processing is required. This will be a subject for further research.

### 3.4 Summary of Chapter 3

The present proposed recognition systems that can classify the shape of hand. The outputs of the systems are checked using the proposed algorithm in the experiment part. In this part, 81.12% success is obtained when all 25 classes are used and 94.45% success is obtained when the grouping method is used. Therefore, the proposed model shows high success with grouping. Thus, it can be used for HCI contexts such as TV control, which uses some designated commands by matching some gestures with commands. In the proposed method, the user should take pictures of their hand for the reference images. This process is just a simple process to save the images. Subsequently, the user can use these systems to control or do something with their hands. Relatively few commands are necessary in this context, so if the user saves the distinct different images for the classification, accuracy will be high.

## Chapter 4

# State based approach to hand gesture recognition

In this chapter, a method and system is proposed, which can recognize the input gestures that are obtained by using a mouse device and single web-camera. Dynamic gestures that express the user's intended meaning are recognized by the proposed systems.

The hand gesture recognition system based on Finite State Machines (FSMs) with probabilities is proposed. Several previous studies have used FSMs approaches for gesture recognition. Gestures are modeled as an ordered sequence of states in the FSMs approaches. Bobick and Wilson (1997) proposed the method where the gesture is considered as a prototype trajectory in a 2D space. The training of the gesture model is done off-line with various clustering algorithms to cluster the gestures in the spatial spaces. The number of states (clusters) is a key concern in achieving high recognition success rates, and it is determined according to the applications.

Fewer training sets are needed in FSMs approaches than the HMMs approaches, because the trajectories of gesture in space are the most important factors in this approach. Therefore, if there are suitable training data sets for gestures, the numbers of sets are not important factors, unlike in the HMMs approaches.

Letters and numbers (0 ~ 9) are used as the input gesture to verify the performance of the proposed systems. These inputs are suitable for confirming the performance of recognition, they have various similar shapes; (for instance, g and q).

In order to obtain the visual input from the camera, the same methods with chapter three are used. After obtaining the centroid of the hand in every frame from several image processing episodes, these were used as the input gestures. In case of the mouse device, the positions of mouse cursor became gestures; thus, its positions are saved to the data set. These input data are trained by two phases: spatial clustering and temporal alignment. After training, several sorts of gestures are tested to the recognizers, which are composed of the ordered sequence of states. This content is prepared to be published in a journal (Yim and Kim, 2013d).

## **4.1 Gesture extraction**

For training or recognizing the gesture of hands, data sets have to be made. Data sets are made using two different methods. First, the mouse device is used to make a data set. Positions of mouse cursor became gestures, so the positions are saved to the data set. The second method is using a camera and vision algorithm. In other words, hand gestures are extracted by a camera. Skin-color detection using RGB color space is used to detect the hand in an image. Then, the center positions of the hand are obtained by using several image processing methods. These positions are used as the data set for the experiments.

### **4.1.1 Experiment Environment**

In the experiment, A personal computer, webcam (Logitech Webcam C160), and the human participants are only used. The computer has Intel(R) core(TM) i5-2500 CPU (3.30GHz quad core), and MATLAB 2011a is used. White panel is used to find hand portions. Then an experiment is performed without any occlusion. Figure 4.1 shows the setting of the experiments. There are webcam, computer, connection line and other structure. In the experiment, a single right hand is moved in front of the white panel, then the images are transferred to the computer in real time. After that, the computer calculates the algorithm.



Figure 4.1: The view of experiments

#### 4.1.2 Mouse Gesture

Mouse gestures are used as the input for the experiments. The position of mouse (x and y) is obtained using the mouse input device. The data of mouse gesture is easy to extract, and noise is relatively small in this case, so it is a good input sources to confirm the performance of the recognizer. The position of the mouse can be obtained using the "windowbuttondownfcn" function in Matlab. Using this function, numerous data is obtained with various time steps. When the mouse is moving while holding down the mouse button, the position of the mouse starts to be stored in an array. After drawing the desired gesture, when the mouse button is released, and the data until the mouse button is released is stored in a new variable. Figure 4.2 shows the drawing environment and the result after drawing the desired gesture. This is an example of a gesture that represents "zero".

If the data obtained by the above process are used without any preprocessing, some problems occur, as shown in Figure 4.3. Figure 4.3 (a) and (b) show the same gesture drawn in different position. If any preprocessing is not done, these two gestures are considered as different gestures. To avoid these problems, it has to be normalized. All input data is normalized for the experiments. First, minimum values of data on each axis (x and y axes) are found. Then, the input data is subtracted by the minimum values found in the above process. The minimum values of input data on each axis will be zero. After that, the maximum values of input data on each axis are found. Using these



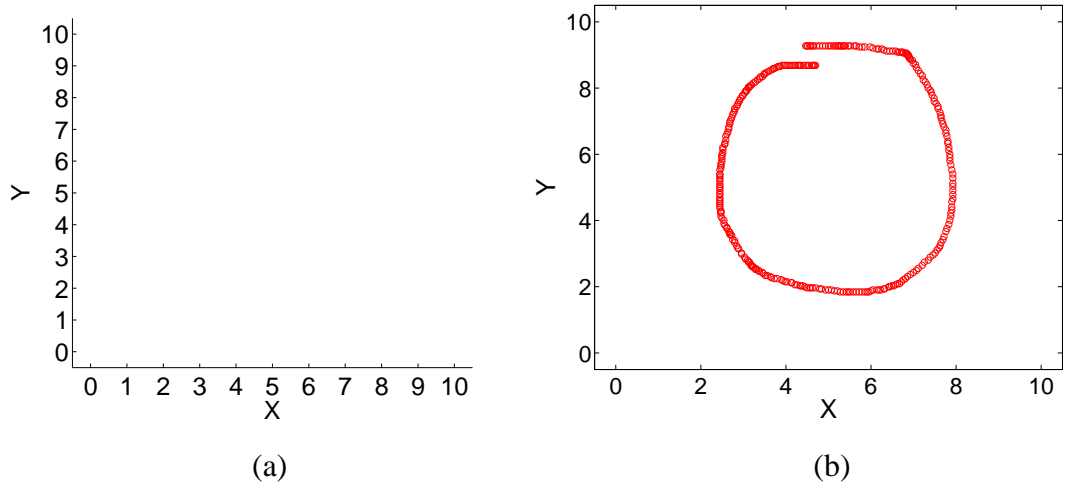


Figure 4.2: The environment for the mouse gesture. (a) The drawing environment (b) The result after drawing the desired gesture

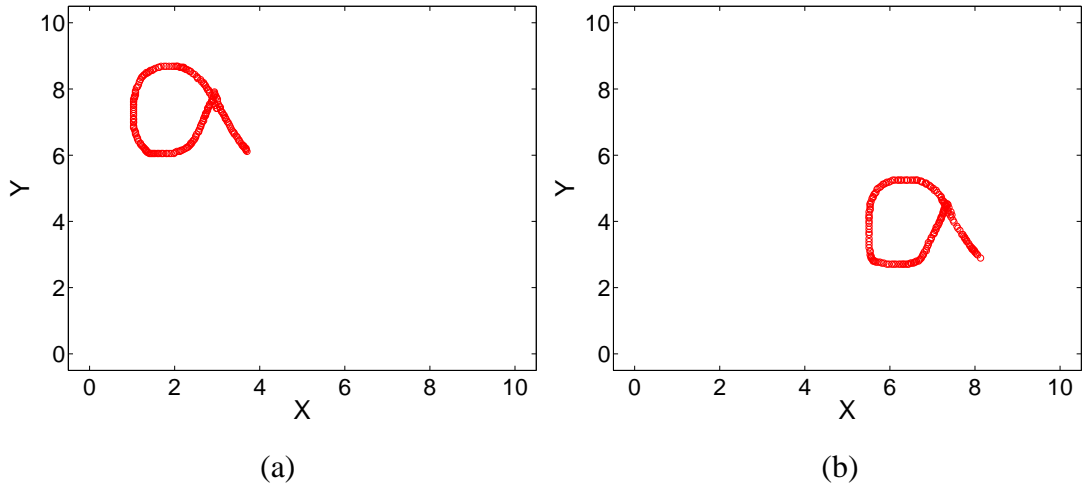


Figure 4.3: Same gestures, (a) and (b) are drawn, but, since these two gestures are located in the different place, they are considered as different gestures even if they have same shape. Therefore, normalization is always necessary.

values, the size of input data can be freely adjusted. The following equation shows the detail of this process.

$$\begin{aligned}
 Min_{x,y} &= \min(input(x,y)) \\
 input(x,y)_{allx,y} &= input(x,y) - Min_{x,y} \\
 Max_{x,y} &= \max(input(x,y)) \\
 input(x)_{allx} &= input(x)_{allx} * width / Max_x \\
 input(y)_{ally} &= input(y)_{ally} * height / Max_y
 \end{aligned} \tag{4.1}$$

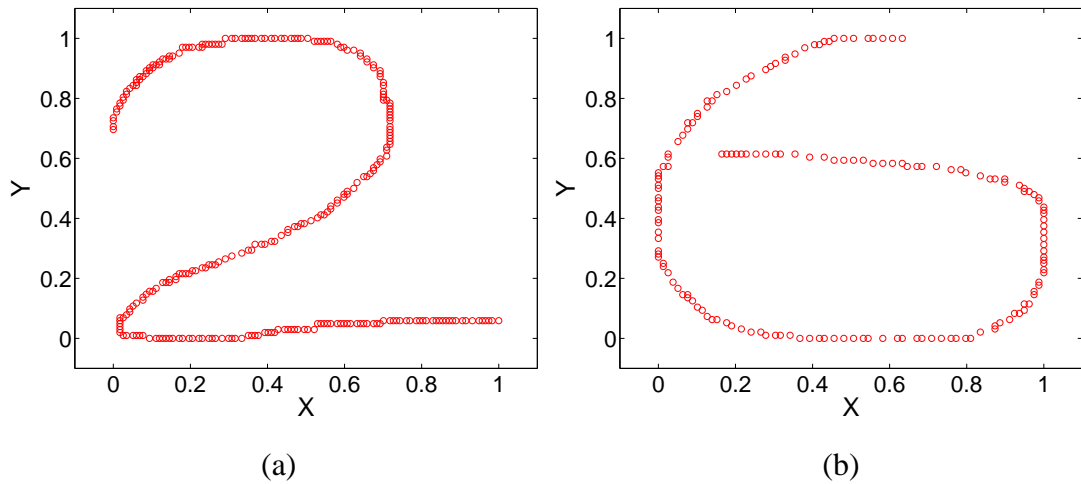


Figure 4.4: After normalization, the data has same maximum and minimum values. (a) a case of two (b) a case of six

After normalization, the data that has the same maximum (width and height) and minimum (zero) values is obtained, as shown in Figure 4.4.

Each variable storing the information of desired gesture has approximately  $100 \sim 200$  sample points. Ten sets for each letter of the alphabet and the numbers are collected; five sets among these data are used for learning, and the remainders are used for the test.

### 4.1.3 Hand Gesture

Hand gestures' information from a single camera is also used for the input. The video is recorded to obtain the information as  $240 \times 320$  size. In this experiment, discontinuous hand gestures are used for the input. In other words, each gesture is saved to a different name. For example, if gestures from zero to nine (10 gestures) are used for the experiment, then there are ten different variables saved into different names. To obtain the useful information for the gesture recognition, it is necessary to extract hand information from the image. Extracting the hand is done by using a similar method to that in Chapter 3. Skin detection using color information is used to the process.

The proposed method can detect and extract the hand well, so the user toned not hide the wrist by wearing a long sleeve. Figure 4.5 shows the properties. Although the user's wrist appears, the method can extract the hand position well.

The objective of this course is finding the center of the hand for the hand position

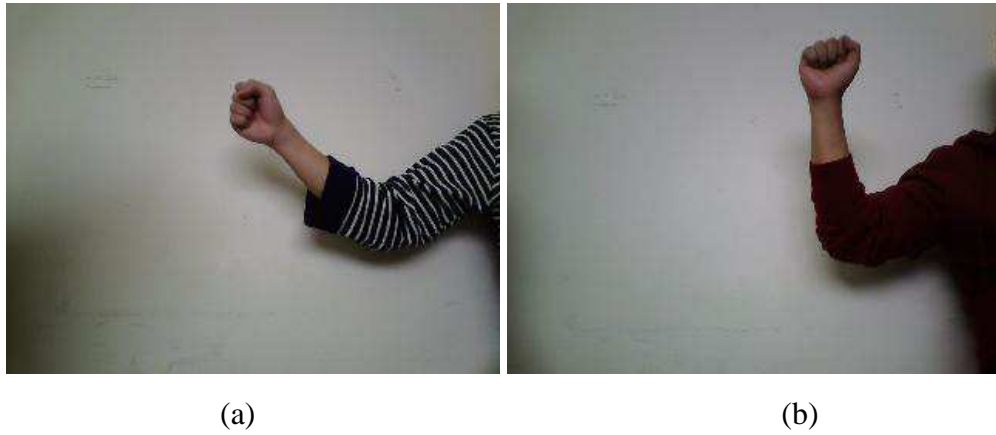


Figure 4.5: When hand is extracted by the proposed method, it doesn't matter that cuff may appear or not.

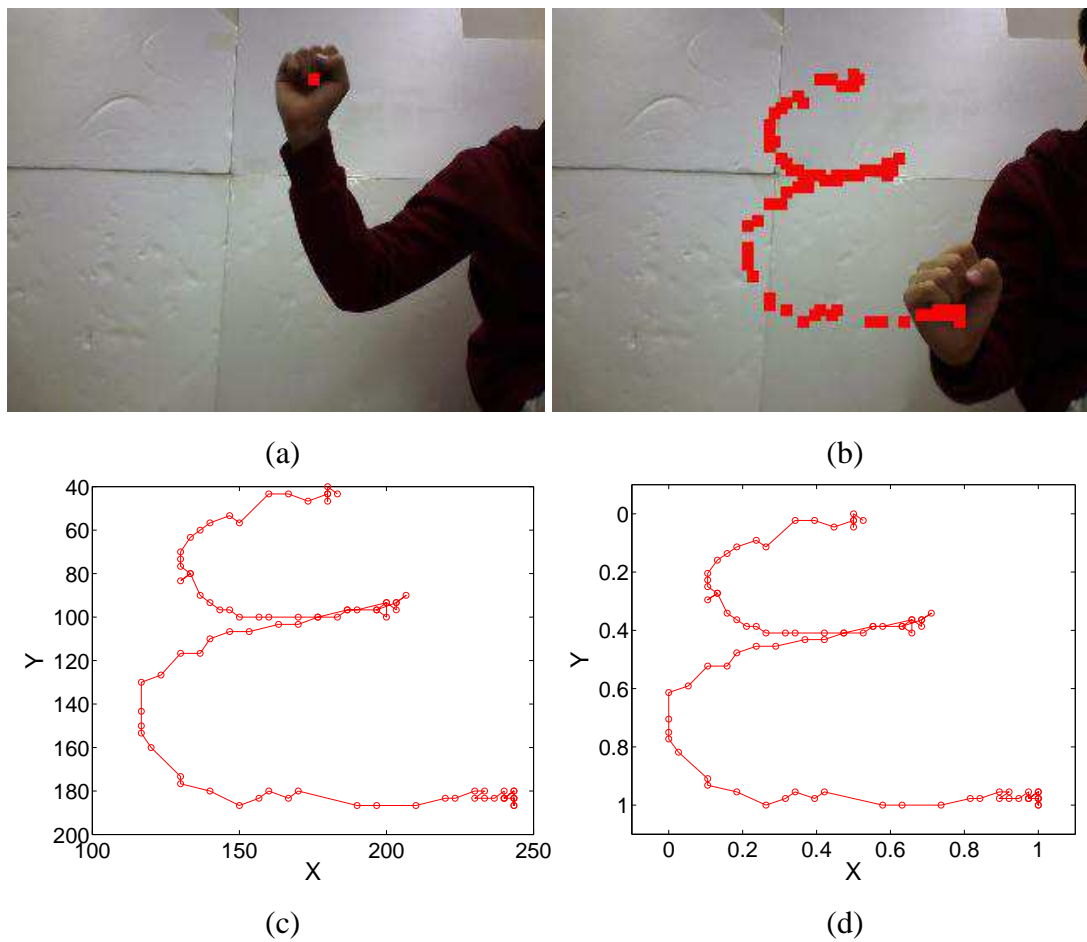


Figure 4.6: The process to extract and save the hand information from the video. (a) Detecting the centroid of hand. Red point. (b) The trajectory of gesture in the video. (c) The saved data from the video (d) normalized data for the experiments

information. The center of gravity for the region detected as the hand is considered as the center of the hand and is saved as such to the gesture information. Figure 4.6 (a) illustrates the centroid value of the hand on the input image. The center of gravity of the hand is saved while the desired gesture is drawing. After drawing the gesture, if the centers of gravity of the hand that have been already saved are drawn on the video, then the Figure 4.6 (b) can be shown. Figure 4.6 (c) shows the extracted data from this process. It is also necessary to normalize data for the same reason as with mouse gesture. Figure 4.6 (d) shows the normalized data.

The center of the hand can be found from this part. The center of the hand is used for the location of 2D image positions of the hands of the user and it is used as the input for the experiments. It is very noisy data and there are many variables to be different, even if the users draw the same gestures. Hence, the information that can be easily influenced by the environments such as the speed of hands or direction of movements is not used. The position data are used for the gesture data.

The number gestures from zero to nine are made. Twenty sets for each numbers are collected, and ten sets among these data are used for learning, and the remainders are used for the test. Figure 4.7 shows a set of number data representing the whole trajectory on the images.

## 4.2 Finite State Machines for gesture recognition

Using data sets of gestures obtained by mouse device and camera, the data is trained and tested the performance of various recognizers. In this section, the proposed recognizer is explained in more detail, which is a state-based approach with probabilities for the hand gesture recognition. Before going to the recognizer, the data set should be trained to make the proper forms. Training is divided into two steps: spatial clustering and temporal alignments. From this process, some parameters to be used to make the recognizer model can be derived. Through the training phase, each gesture is also re-defined as the sequences of states in spatial and temporal space. These redefined data sets are used to recognize the gestures with FSMs with probabilities.

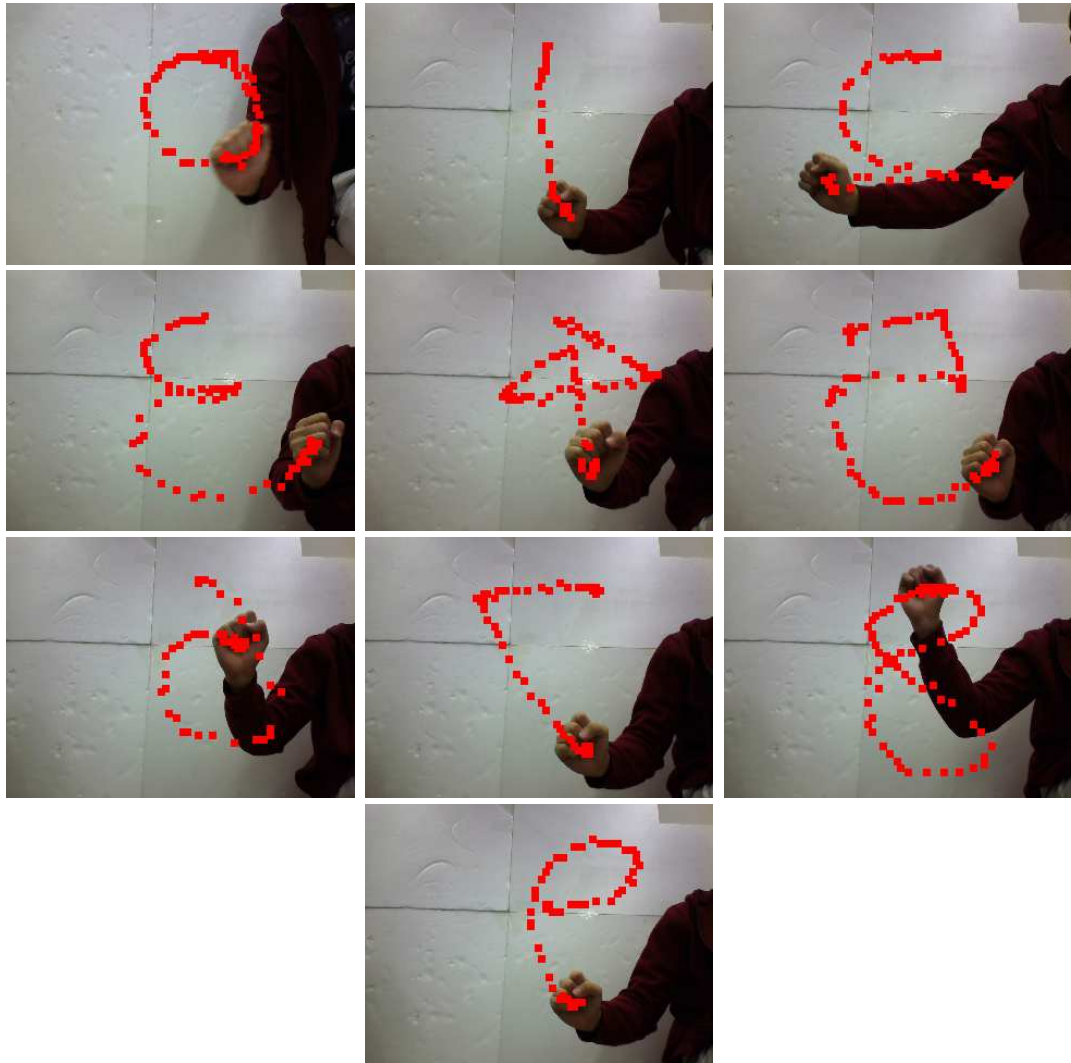


Figure 4.7: Trajectories of hand gestures obtained by using single camera

#### 4.2.1 Training phase

Gestures are data with spatial and temporal information, so there are two spaces: spatial space and temporal space. Figure 4.8 (a) shows these two spaces clearly. The Z-axis in Figure 4.8 (a) indicates time series, and the X and Y axes show the center position of the mouse or hand. If the data that are to be trained by the proposed method are not properly aligned by preprocessing, it is difficult to train the spatial and temporal information at the same time. It is desirable to do this process easily, even if the data are not aligned in time. Thus, these two spaces are decomposed for ease of training. In other words, the information on spatial and temporal domains is separated for the training. It is a similar process to that in the method proposed by (Hong et al., 2000b,a).

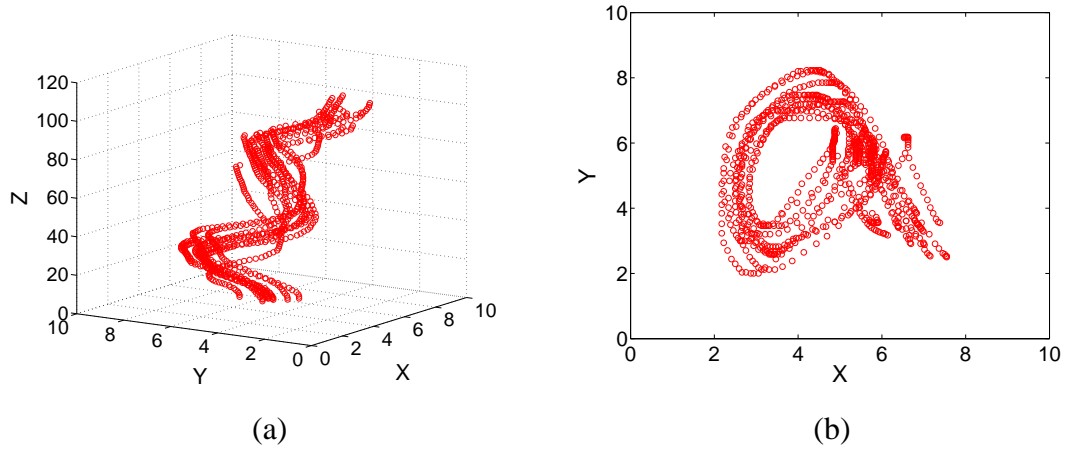


Figure 4.8: The gestures of 'a'. (a) Distribution of data with spatial and temporal information. (b) Distribution of data only with spatial information.

However, a new process is proposed to obtain better performances and the parameters that can be efficiently used in the recognizer. This method is also analyzed, comparing it with the existing method in multiple ways in more detail. There are two phases for training to recognize the gesture using the FSMs approach. In the first phase, data sets of gestures are dispersed in space ignoring the temporal information and clustered with several states, also known as the clusters. These states can express the gesture data most fully to the sequences of states. The second phase is to align the data using temporal information using the sequences of states. The decomposition of these two spaces (spatial and temporal spaces) for training make training the data easy. The training with decomposition of spatial and temporal information is effective because the spatial information shows the structures of the gesture data well, and the temporal information shows the movement of gestures well.

This training process is a completely different process to that in the HMM, which have been widely used in speech recognition. HMMs are often used for modeling the sequences of data in time, showing highly successful performances. They are doubly stochastic process models modeling via symbols representable clearly for stochastic structure of the pattern, which is difficult to model. In HMMs, the states of pattern can be represented by joint probability distribution of states and observations based on the before and after time. Therefore, symbol sets for modeling must be a type that contains the context with respect to the time change and the unique features of the pattern. First of all, the states of pattern have to be represented about time  $t$ ,  $s_t$ , to express the relation with the pattern of data and time. And then, for modeling the

process of being deployed, the states in time  $t + 1$ ,  $s_{t+1}$ , are represented by the equation related with states in time  $t$ . Therefore, when  $s_t$  equals  $q_i$  which can be also expressed as  $i$ th hidden states, the probability that  $s_{t+1}$  become  $q_j$  can be expressed as  $a_{ij}$  called transition probability,  $A$ , as in Equation 1. If there are states sequences of a model called  $\lambda$  of  $a_{ij}$ , and Full-time called  $T$ , the probability that the model  $\lambda$  generates these states sequences is to be represented as the product of the transition probability between the state of a series.

To model the pattern using representable symbol approaches to indicate the probability of symbols to occur in certain states, the symbol in time  $t$  is represented as  $v_k$ . The probability that the symbol,  $v_k$ , is generated to the state  $s_t = q_i$  can be represented as  $P(v_k | s_t = q_i) = b_i(k)$ . In other words,  $b_i(k)$  indicates the probability of output symbols  $v_k$  at state  $q_i$ , so, when there are the set of possible output symbols in total-time  $T$ , the whole sequences of states can be modeled by finding states,  $s_t = q_i$ , which have a greater occurrence possibility of the  $v_k$  in time  $t$ . The above two parameters,  $a_{ij}$  and  $b_i(k)$ , have to satisfy the following equation :

$$\begin{aligned} \sum_j a_{ij} &= 1 \quad \text{for all } i \\ \sum_k b_i(k) &= 1 \quad \text{for all } i \end{aligned} \tag{4.2}$$

With these properties, the parameters for model  $\lambda$  are optimized by using the Baum-Welch parameter re-estimation algorithm or other algorithms. Baum-Welch parameter re-estimation algorithm, which is also known as EM algorithm, using the stage of maximization and the stage of expectation, is the method to obtain the  $\lambda = (A, B, \pi)$ , which makes the maximum probability  $P(O | \lambda)$ . It is the most important problem to determine the performance of HMM. Baum-Welch parameter re-estimation algorithm is briefly explained below (Rabiner, 1989).

Assume that there are  $N$  number of states and  $M$  number of possible output symbols. To obtain the optimized  $\lambda$ , a new variable  $\xi_t(i, j)$  has to be defined. In given model parameters, which are not optimized, and observation sequences (symbols), it indicates that the probability of being in state  $s_t = q_i$  and  $s_{t+1} = q_j$ .

$$\begin{aligned} \xi_t(i, j) &= P(s_t = q_i, s_{t+1} = q_j | O, \lambda) \\ &= \frac{P(s_t = q_i, s_{t+1} = q_j, O | \lambda)}{P(O | \lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \end{aligned} \tag{4.3}$$

where forward variable  $\alpha_t(i)$  and backward variable  $\beta_t(i)$  are obtained by forward algorithm and backward algorithm, respectively. Using  $\xi_t(i, j)$ , the probability of being  $s_t = q_i$ ,  $\gamma_t(i)$ , can be defined with the following equation:

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i, j) \quad (4.4)$$

Using these two formulas the parameters of HMM,  $\lambda = (A, B, \pi)$ , can be re-estimated with following equation:

$$\pi_i = \gamma_1(i) \quad (4.5)$$

$$a_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad (4.6)$$

$$b_j(k) = \frac{\sum_{t=1, O_t=v_k}^T \gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)} \quad (4.7)$$

where  $\pi_i$  means the probability of being in  $s_1 = q_i$ . This re-estimation algorithm is continuously repeated until  $P(O|\lambda_{t+1})$  becomes smaller than  $P(O|\lambda_t)$  or some fixed iteration number. The input data can be recognized by several HMMs that have the parameters trained by this process. It was proved that they have a high success rate for modeling and recognizing the data sequences, such as gestures or speech. To obtain high performance in HMMs, encoding data sequences is important in its structures. This process seems too complex. Although HMMs have superiority modeling of the data sequences, many data sets are needed to train the data sequences. In other words, since the observation data with high probability statistically at a certain state have to be found, in order to increase the accuracy, relatively many data sets are required. In addition, there are some missing quantities when proper parameters are obtained through training. It also requires a large amount of calculation, so sometimes the covariance matrix is defined as diagonal matrix to reduce the computational complexity problem.

The FSMs approach method with probability is proposed to solve the problems that occur in training of HMMs. First phase of training only uses the spatial information out of all gestures data. Figure 4.8 (b) shows the spatial information of gesture, 'a', ignoring the temporal information. When there is only spatial information, the distribution of data is determined while clustering the data using a clustering algorithm.



There are several algorithms used for data clustering. These clustering algorithms are commonly classified as the unsupervised learning (Wagstaff et al., 2001). Expectation Maximization (EM) and K-means are two standard algorithms for doing this process (Alldrin et al., 2003). K-means algorithm is an algorithm to find the  $k$  number of clusters (Linde et al., 1980; Verevka and Buchanan, 1995). At first, the initial positions of  $k$  clusters are set randomly. The distances between every data point and center of clusters are computed to assign the clusters to the data point. Each data point chooses the closest cluster from among all clusters. After that, each cluster center position is updated by using all data points that are assigned in that cluster. It is repeated until some conditions are satisfied, such as there are no more reassigned data points or fixed iteration number. On the other hand, EM algorithm use a mixture of Gaussians to find the clusters given data point. There are mean and covariance matrixes of each Gaussian, and the covariance matrix has forms, such as diagonal, spherical, and full. It depends on the organization of the data set. The initial parameters (mean and covariance) that are trained by using EM are initialized randomly or the result of the K-means algorithm. It is also repeated to update the means and covariances of Gaussian.

These two algorithms have a lot of versions that differ slightly. EM algorithm, especially GMM, is used in the method, and K-means algorithm is also used to set the initial parameters of EM. Each cluster is called a state, and each datum which is located in spatial spaces is marked with class in the state that is the closest to the position of each data point. The centroid and covariance matrix of each state are represented as the following equation. The distances between the centroid of each state and data point are calculated by using Mahalanobis distance, which is the distance between data point and the center of mass in the state divided by the width of the ellipsoid in the direction of the data point, because the Euclidean distance cannot represent adequately if the data set is spread out over a large range or a small range. To represent this information, standard deviation of the distance between data point and the center of mass in the state is needed. For this process, Mahalanobis distance is used.

$$\mu_i = E(X) \quad (4.8)$$

$$\Sigma_i = E((X - \mu_i)(X - \mu_i)^T) \quad (4.9)$$

$$Distance(X, \mu_i) = \sqrt{(X - \mu_i)^T \Sigma_i^{-1} (X - \mu_i)} \quad (4.10)$$

where  $X$  means  $(x, y)^T$  and  $i$  indicates  $i$ th state. Figure 4.9 shows the result after training the gesture, 'a', using the distribution of data in Figure 4.8 (b).

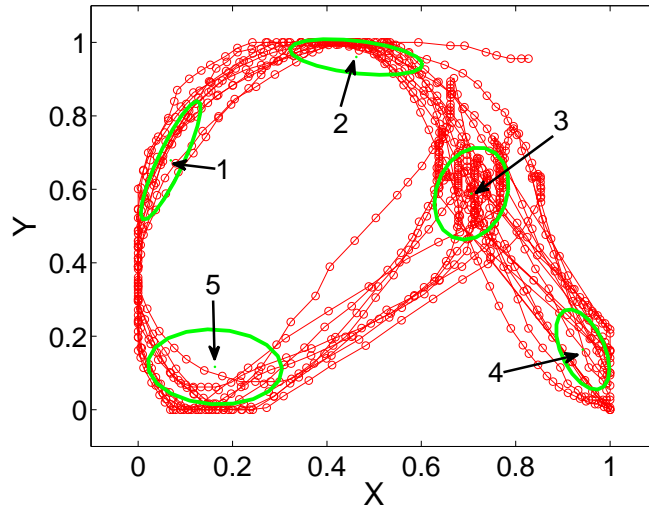


Figure 4.9: After clustering the gesture information, 'a', without temporal information

For clustering data, the number of states is an important factor to affect the performance of systems. In HMMs, it is fixed to train at the initial step; thus, in order to verify the number of states that shows the best performance, the performance have to be checked by increasing the number of states one by one. To solve these inconveniences, in the proposed method, the number of states is decided by comparing with a fixed threshold and average distances between each data point and its state as in the following equation.

$$SumDist = \frac{\sqrt{\sum_i (X_i - \mu_i)^2}}{N} \quad (4.11)$$

where  $X_i$  indicates the data point that is assigned to the  $i$ th states and  $N$  is the number of states. If the value of  $SumDist$  is larger than fixed threshold, a state is added to the spatial spaces to represent more efficiently the composition of data. For example, in the case in Figure 4.9 case, there are two states at first. By using the algorithm, the number of states is increased to five states. The figure shows that the spatial information of data are represented well with these five states.

Figure 4.10 shows the states position and distribution of each number gesture using the mouse device after spatial clustering. Each number has a different number of states, explaining each gesture better than with a fixed number of states.

After training the spatial information to assign states, the second phase begins to align the data using temporal information. Using temporal information, each data point is reassigned to the state to which it belongs in accordance with the time. In Figure 4.9,

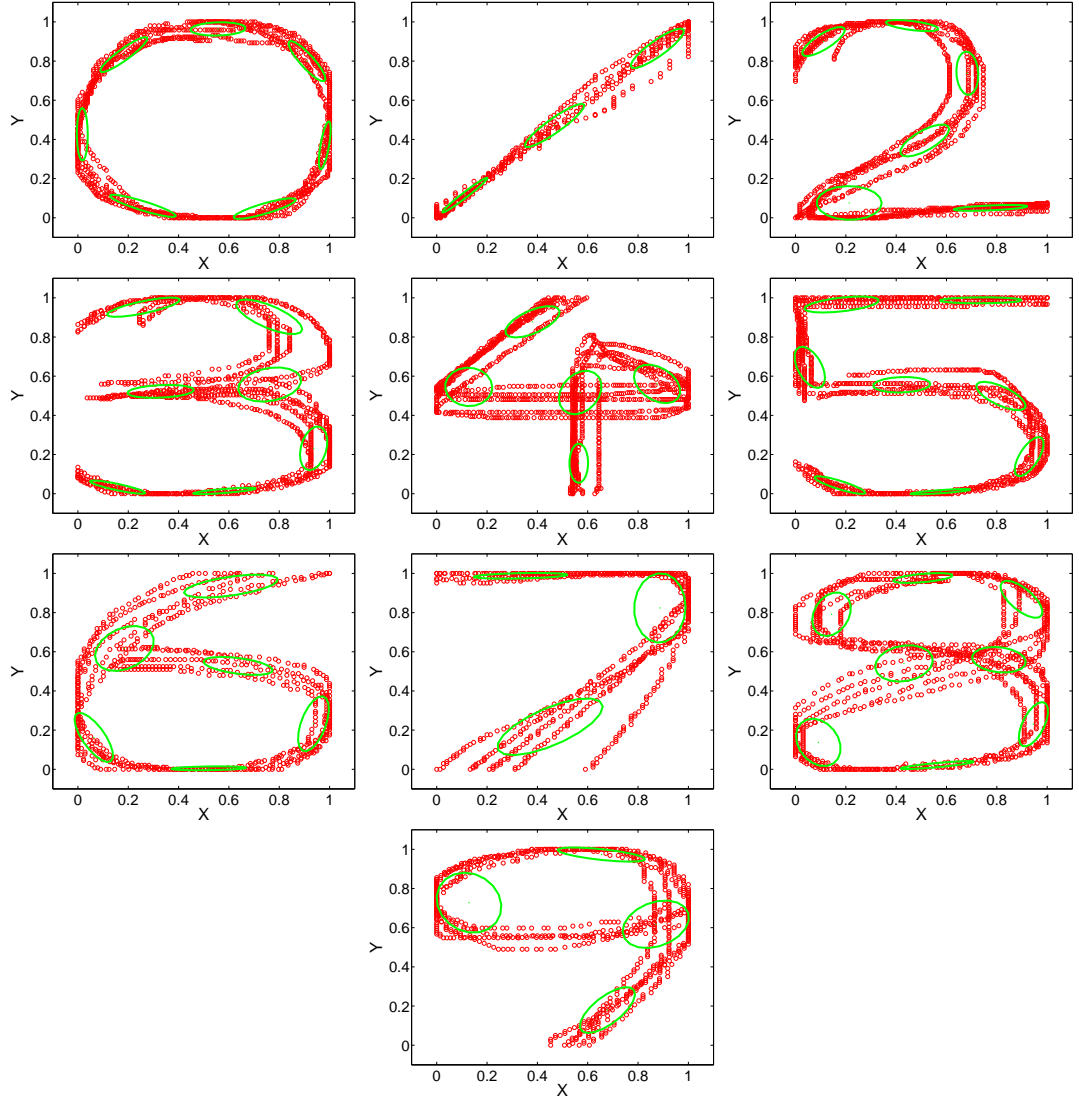


Figure 4.10: Mouse gesture after clustering the gesture information

since each data point of the "a" gesture is assigned to the state in which it belongs in accordance with the time, it can be represented as follows:

333...333222...222111...111555...555333...333444...444.

In the sequences of data, first state is only chosen to redefine the gesture data among the states that are repeated. Therefore, the "a" gesture is redefined by this process as  $3 \rightarrow 2 \rightarrow 1 \rightarrow 5 \rightarrow 3 \rightarrow 4$ . However, all data sequences in the same gesture are not always the same. There are many identical gesture sets for training in Figure 4.9; however, each gesture has some possibility to have different start positions or noise. For example, in order to draw the correct gesture in Figure 4.9, the gesture should start at state 3, but sometimes the gesture starts at state 2. In this case, the "a" gesture is

redefined as  $2 \rightarrow 1 \rightarrow 5 \rightarrow 3 \rightarrow 4$ . It is a little bit different to the initial state; however, the tendency of gesture is the same. To reduce misrecognition the probability of initial states is computed that is similar with  $\pi$  in HMMs.

It can be also recognized when a state transits to the other state using the trajectories of gesture. Through this process, the transition matrix of training data can be computed. This process can be represented as the following equations using similar notation to that of Baum-Welch algorithms.

$$S(t) = \arg \max_i (b_i(O_t)) \quad (4.12)$$

$$\xi_t(i, j) = p(S(t) = q_i, S(t+1) = q_j) \quad (4.13)$$

$$\xi(i, j) = \sum_{t=1}^{T-1} \xi_t(i, j) \quad (4.14)$$

$$\gamma(i) = \sum_{j=1}^N \xi(i, j) \quad (4.15)$$

$$a_{ij} = \frac{\xi(i, j)}{\gamma(i)} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{j=1}^N \xi(i, j)} \quad (4.16)$$

$S(t)$  means the closest state where a data point belongs at time  $t$ .  $\xi_t(i, j)$  indicates the probability of being in state  $S(t) = q_i, S(t+1) = q_j$ . It has a different meaning to the  $\xi_t(i, j)$  in HMMs: the current data point and next data point are only considered to calculate the values of  $\xi_t(i, j)$ .  $\xi(i, j)$  means the number of data points of being in state  $S(t) = q_i, S(t+1) = q_j$  throughout the whole time.  $\gamma(i)$  means the number of data points of being in state  $S(t) = q_i$  throughout the whole time. Through these parameters, the transition probability matrix,  $a_{ij}$ , is computed.

Following equation shows the example of transition matrix,  $A$ , of 'a' gesture.

$$A = \begin{bmatrix} 0.9206 & 0 & 0 & 0 & 0.0794 \\ 0.0556 & 0.9333 & 0.0111 & 0 & 0 \\ 0 & 0.0270 & 0.9459 & 0.0270 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0.0543 & 0 & 0.9457 \end{bmatrix} \quad (4.17)$$

These transition matrix and initial state probability will be used useful to solve the noise input problems in the recognizer system.

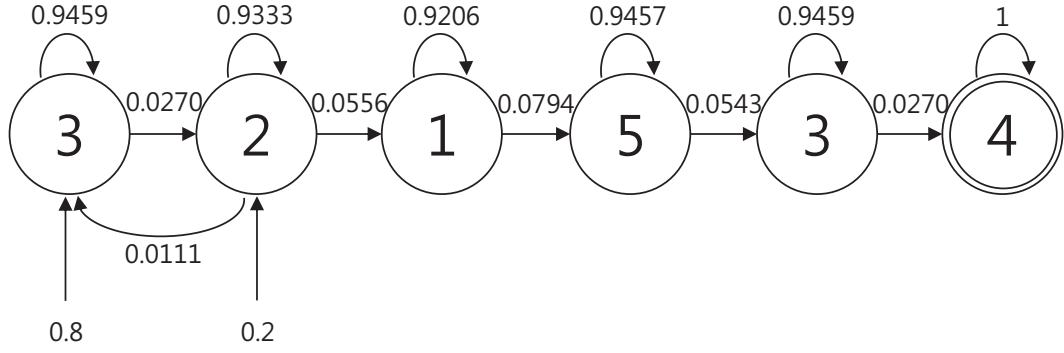


Figure 4.11: Finite State Machines with probability of gesture 'a'.

#### 4.2.2 FSMs approach recognizer

Through the training phase using spatial and temporal information, the gestures are redefined as the ordered sequences of state. For instance, the "a" gesture is redefined as  $3 \rightarrow 2 \rightarrow 1 \rightarrow 5 \rightarrow 3 \rightarrow 4$ . Now, a recognizer has to be made, which can distinguish gestures using the ordered sequences of state for each gesture. The FSMs with probabilities were proposed as a recognizer. The FSMs is a machine that is composed of a finite number of states. They receive the input data and work as the transition to another or the same state from the current state in response to the input. Using these properties of FSMs, the redefined gestures will be represented as the form of FSMs with probabilities.

With transition matrix,  $A$ , and initial state probability,  $\pi$ , the recognizer can be represented as shown in figure 4.11.

As Figure 4.11 shows, there are six states that are named by their number. It is same with the ordered sequences of state obtained by the training phase. There are two input states, 3 and 2, because the drawing method of each person can be different. Number 4 state is an acceptor. Therefore, this machine will finish in a number 4 state, called accept state, if all states in the gesture recognizer are traversed. If that happens, the input gesture is considered as an "a" gesture. That is, if data points of a gesture passed through every states of gesture recognizer, then the "a" gesture is recognized.

For finding the optimal sequence of states, there are two methods. The first method is to find a sequence of individually most likely states at each time, even if it is not guaranteed that a sequence is not always the most likely state sequence given an observation sequence. This method is also called the forward-backward algorithm. The

second method is to find the most likely state sequence by maximizing  $P(q|Q, \lambda)$ , the Viterbi algorithm. In the method, a sequence of individually most likely states at each time is important to assign a state to the data point at time  $t$ . Therefore, the recognizer process is similar with forward-backward algorithm except that forward-backward algorithm is needed to whole observation sequences,  $1 \leq t \leq T$ . The recognizer does not require the whole observation sequences to recognize. For this reason, the algorithm is more similar with variable  $\alpha$ , which is used to compute  $P(O|\lambda)$  in the forward algorithm.  $\alpha_t(i)$  is called the forward variable, and it indicates  $P(o_1, o_2, \dots, o_t, s_t = q_i | \lambda)$ , which is already introduced in equation 2.3. The method can be induced by using this algorithm with some modification and assumptions. At time  $t = 1$ , variable  $\alpha_1(i)$  is used as in the following equation:

$$\alpha_1(i) = \pi_i b_i(o_1) \quad (4.18)$$

where  $\pi_i$  means the initial probability in state  $i$ . In the forward algorithm, when each state is represented as a single multivariate Gaussian distribution, the variable  $b_i(o_t)$  has the probability as follows:

$$b_i(o_t) = \frac{1}{\sqrt{(2\pi)^K |\Sigma_i|}} \exp\left[-\frac{1}{2}(o_t - \mu_i)^T \Sigma_i^{-1} (o_t - \mu_i)\right] \quad (4.19)$$

where  $K$  refers to the dimensionality of the observation space. However, in the recognizer, the Mahalanobis distance is used, which is already defined in equation 4.10, instead of equation 4.19. The Mahalanobis distance can be used instead of the observation probability,  $b_i(o_t)$ . Because the square value of the Mahalanobis distance is in inverse proportion to the observation probabilities. It is proved as the following equation.

$$f = \frac{1}{\sqrt{(2\pi)^2 |\Sigma|}} \exp\left[-\frac{1}{2}(X - \mu)^T \Sigma^{-1} (X - \mu)\right] \quad (4.20)$$

$$\ln f = -\ln 2\pi - \frac{1}{2} \ln |\Sigma| - \frac{1}{2} (X - \mu)^T \Sigma^{-1} (X - \mu) \quad (4.21)$$

$$\ln f \propto -d^2 \quad (4.22)$$

where  $d^2 = (X - \mu)^T \Sigma^{-1} (X - \mu)$ , and  $\ln 2\pi$  and  $\ln |\Sigma|$  are constant values. Since if the square of Mahalanobis distance,  $d^2$ , is increased, then the probability,  $f$ , will be decreased, the minimum sum of the squares of Mahalanobis distance have to be found for all data points. The Mahalanobis distance can be also used rather than its square. Therefore, equation 4.18 can be redefined as follows :

$$\alpha_1(i) = \pi_i \text{Distance}(X_1, \mu_i) \quad (4.23)$$

After initialization, the recurrence formula of the forward algorithm is redefined as follows:

$$\alpha_{t+1}(j) = \left[ \sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1}) \quad (4.24)$$

However, in the proposed method, summing the other states at same time is not needed, as the most likely state is only chosen at each time. Therefore, the proposed method is similar with the recurrence formula of the Viterbi algorithm in this part. In the Viterbi algorithm, variable  $\delta_t(i)$  which means the largest probability of a path which is in state  $i$  at time  $t$  with  $t$  observation sequences, is used as in the following equation:

$$\delta_{t+1}(j) = \max_{1 \leq i \leq N} [\delta_t(i) a_{ij}] b_j(O_{t+1}) \quad (4.25)$$

where  $2 \leq t \leq T$ . On the other hand, in the proposed method, the smaller the distance values are, the better, so the minimum values of distance have to be found. In addition, it is the goal to find a minimum distance path, so the product of maximum value and observation probability have to be changed in the summation. This equation is changed as follows:

$$\alpha_{t+1}(j) = \min_{1 \leq i \leq N} [\alpha_t(i) a_{ij}] + \text{Distance}(X_1, \mu_i) \quad (4.26)$$

Unlike in Viterbi algorithm, in the proposed method, the back-tracking process is not needed, so the state which has the minimum distance is the  $i$  state, which can minimize the variable  $\alpha_t$ .

$$s_t^* = \arg \min_{1 \leq i \leq N} [\alpha_t(j)] \quad (4.27)$$

where  $s_t$  means the state at time  $t$ . Transition matrix,  $a_{ij}$ , is used a little bit differently in the method. FSMs of Figure 4.11 have transition probabilities for each state-to-state transition. These transition probabilities are used differently depending on the cases. First, transition (from  $i$ th state to  $i + 1$  state) occurs when it is satisfied with the following conditions:

$$\text{Distance}(X, \mu_{i+1}) \leq \text{Distance}(X, \mu_i) \quad (4.28)$$

$$\text{Distance}(X, \mu_{i+1}) \leq d_{\text{threshold}} \quad (4.29)$$

$$A(i, i + 1) \neq 0 \quad (4.30)$$

where  $d_{\text{threshold}}$  is a fixed value which can be determined using Mahalanobis distance equation:

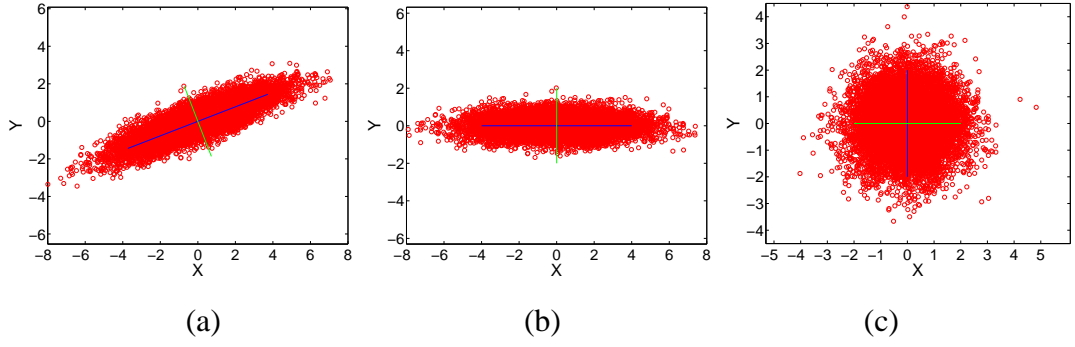


Figure 4.12: Whitening process

This fixed distance,  $d_{threshold}$ , defines a probability ellipsoid.

The process changing the data which has certain mean and variance to normal distribution is needed to determine the  $d_{threshold}$ . Whitening process is used to do this task. Figure 4.12 shows the whitening process.

Through the product of data  $X$  and eigen vector matrix  $\Phi$  of  $X$ , the main component of data becomes to the principal axis as like Figure 4.12 (a) to (b).

$$Y = \Phi^T X \quad (4.31)$$

The data distributions of Figure 4.12 (a) and (b) are the same since it is assumed that the data distribution is Gaussian. This process is proved as in the following equations (Hansen, 2005).

$$N(\mu, \Sigma) \sim \mu + N(0, \Sigma) \quad (4.32)$$

$$N(0, \Sigma) \sim \Phi N(0, \Lambda) \quad (4.33)$$

$$\Phi^T N(0, \Sigma) \sim N(0, \Lambda) \quad (4.34)$$

Therefore, (b) distribution has the eigenvalue variances.

If the changed data is multiplied by the inverse of the squared root of the eigenvalue matrix, the normalized data can be obtained with variance of 1.

$$Z = \Lambda^{-1/2} Y \quad (4.35)$$

This part is a process in Figure 4.12 (c). This process can be also proved.

$$N(0, \Lambda) \sim \Lambda^{1/2} N(0, I) \quad (4.36)$$

$$\Lambda^{-1/2} N(0, \Lambda) \sim N(0, I) \quad (4.37)$$



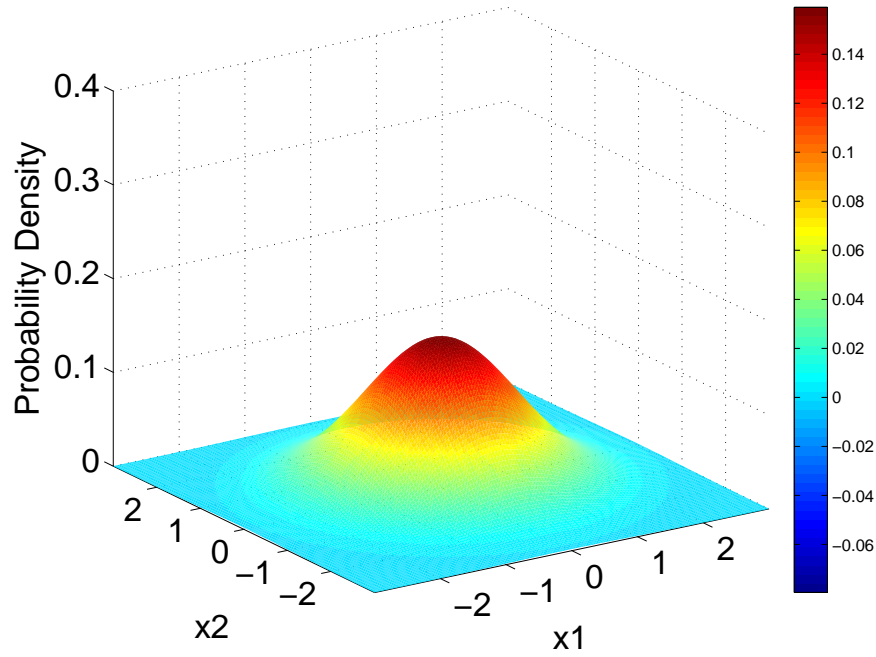


Figure 4.13: bivariate normal distribution

Cumulative distribution function for bivariate distribution is used to obtain the contour line which represents 95% or 99%. For obtaining this line, complicated procedure of double integration is commonly used, but approximation process is used for convenience. The standard bivariate density function is like below equation.

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left[-\frac{1}{2(1-\rho^2)}\left(\frac{(x-\mu_x)^2}{\sigma_x^2} + \frac{(y-\mu_y)^2}{\sigma_y^2} - \frac{2\rho(x-\mu_x)(y-\mu_y)}{\sigma_x\sigma_y}\right)\right] \quad (4.38)$$

Using this equation, Figure 4.13 can be obtained.

The volume of Figure 4.13 is 1. To get points which have 0.95 or 0.99 volume is the purpose by using this. For convenience, it is assumed that the mean is zero. At first, when a point is same with mean value, maximum probability,  $Z_{max} = f(0,0)$ , is obtained. After dividing  $Z_{max}$  into  $N$  equal parts and deciding the wanted point, if the volume of each cylinder is calculated, the wanted point can be obtained. For example, Assuming that  $x_1 = a$  is the point to want to know. it doesn't matter if  $x_2 = 0$ . Because it is the normal distribution. The probability at point  $a$  is  $f_a = f(a,0)$ . Then, the number of division between  $Z_{max}$  and  $f_a$  among  $N$  division is  $n = N * \frac{Z_{max} - f_a}{Z_{max}}$ . After obtaining  $n$ , the radius ( $r$ ) of each circle can be obtained the inverse equation of

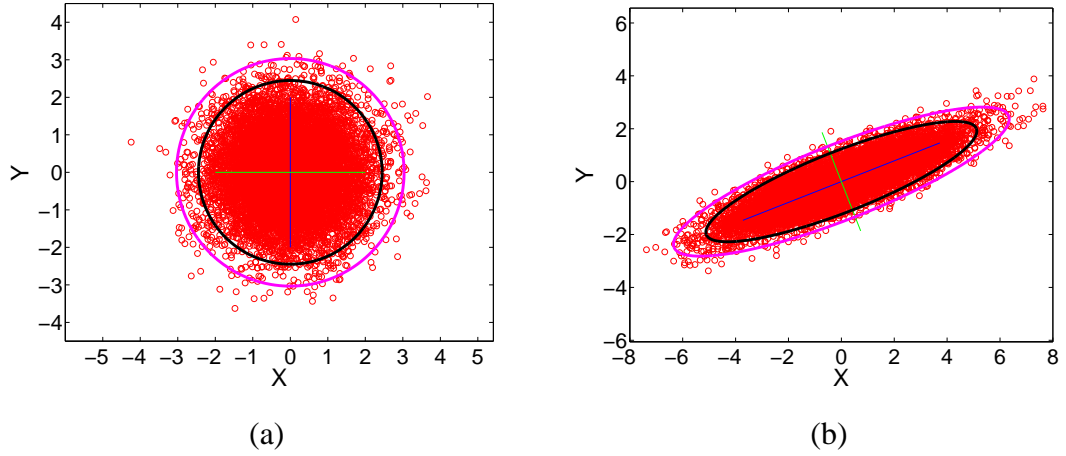


Figure 4.14: Contour line with 95% and 99% probability density. black line : 95%, pink line : 99%

bivariate density function as like following equation.

$$r = \sqrt{-2 * (1 - \rho^2) * \sigma_x^2 * \ln 2\pi \sigma_x \sigma_y \sqrt{1 - \rho^2} * Z_{max} * \frac{1-i}{N}} \quad (4.39)$$

where  $1 \leq i \leq n$ . After obtaining  $r$ , the volume of cylinders between beginning to  $n$  have to be computed as like  $r^2 \pi Z_{max} / N$ . After summing all of cylinder volume and another cylinder volume using  $a^2 \pi (N - n) Z_{max} / N$ , then the volume can be obtained at wanted point. Through this process, the probability between centroid and some point can be obtained. The point that the volume becomes 0.95 is  $a = 2.448$ , and point of 0.99 is  $a = 3.033$ . The point that the volume becomes 0.995 is  $a = 3.255$ , and point of 0.999 is  $a = 3.71$ . Using these points, contour line is drawn as like Figure 4.14.

By using the inverse process of whitening process, contour line can be drawn on the original data as like Figure 4.14 (b). This distance value is used  $d_{threshold}$ . Both volumes of (a) and (b) in  $d_{threshold}$  are same and it is proved in appendix.

Through this process,  $d_{threshold}$  can be determined. If the distance between each data point and the centroid of state  $i$  is smaller than  $d_{threshold}$ , it means that the data point is inside the  $i$ th ellipsoid. On the other hand, if the distance is greater than  $d_{threshold}$ , the data point does not belong to the state  $i$ .

$A(i, i+1)$  indicates the probability of transit from  $i$  state to  $i+1$  state. When state transition occurs from left to right, the probabilities of transition matrix,  $A$ , are not used directly. It is used just to confirm whether  $A(i, i+1)$  is zero or not. For example, in Figure 4.11, if the transition condition is satisfied, so the state transition occurs to number 1 from number 2, then its transition probability is 0.0556. However, in this

case, the state transition occurs from left to right, so the transition probability is just used to check whether  $A(i, i+1)$  is zero or not. This property can block the wrong input data to be recognized, because even if the condition of transition is satisfied, the state transition does not occur when the transition probability is zero. On the other hand, when state transition occurs from right to left, the probabilities of transition matrix,  $A$ , are used directly. In this case, because it is going to the opposite direction to the original and it means the input data has noise, an additional condition has to be added.

$$\sum_t A(i, i-1) \leq p_{threshold} \quad (4.40)$$

where  $p_{threshold}$  is a fixed value. This equation is needed when the sequences of state are changed frequently, because of noise. It will improve the performances when there are noisy data for the input. If the accumulation of probabilities become larger than  $p_{threshold}$ , then the FSMs are reset. This means that the input data is not recognized for this gesture. This property results in better performances than the existing methods. If deterministic FSMs without transition probabilities are used for the recognizer, then the values for each state-to-state transition in the FSMs will be zero or one. In this case, if the gesture data that has similar trajectories with the trained is used for the input, it also works well. However, if the input data with noise is used, then there are many chances of misrecognition.

If more than two recognizers are reached to the acceptor states, then a recognizer is chosen that has the shortest average accumulated distance between data point and assigned states. This means that the input data trajectory closely follows the trajectory of the recognizer that has the shortest average accumulated distance.

### 4.3 Performance experiments

In this section, the experiments are performed to recognize various mouse and hand gestures using only visual input. Letters and numbers (0 ~ 9) are used as the input for the recognition experiments. These inputs are suitable for confirming the performance of recognition, as it has the various similar shapes, such as "g" and "q".

Matlab2011a was used to perform experiments. The gesture videos were filmed at twenty frames per second. In the proposed method, the processes used to obtain the center of the hand in every frame from the video take a long time. It takes 0.1476

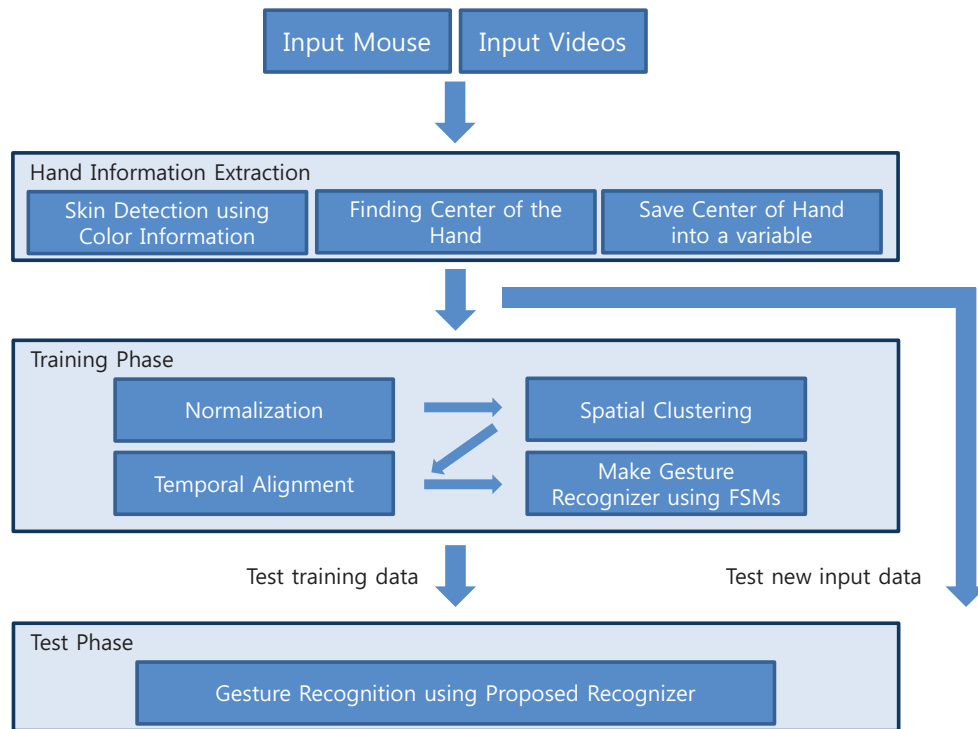


Figure 4.15: Overview of whole process

seconds that the center of the hand is obtained from one frame by using image processing. If OpenCV or some other program is used to obtain the center of the hand in every frame of video, then the time taken for these image processing takes will be reduced significantly. For training phase, it takes 1.092192 seconds to make a FSMs approach recognizer using 10 training data, which are already changed into the set of hand's center from the video. In the test phase, to recognize one input video from the 10 gestures classes takes 0.0402 seconds.

### 4.3.1 Parameters for HMM

In this section, the efficiency of parameters obtained by the training phase of the method will be confirmed. The parameters can be obtained such as transition matrix, mean, and covariance of cluster from the training phase of the method. These parameters can be used for the initial parameters of HMM training. Parameters obtained by the method and basic HMM are compared using log likelihood of data. The information used for the input for HMM are also the gesture input, 2D data. Therefore, the vector of observed sequences such as  $o_t$  is  $2 \times 1$  matrix. These data, which are easily obtained from the mouse input in Matlab, are saved in the array while the mouse

Table 4.1: The number of training iteration for each number data

	0	1	2	3	4	5	6	7	8	9
HMMs method	12.4	22.6	19.1	16.1	15.9	24.1	21.9	35.2	16.8	20.0
Proposed method	7	26	11	12	7	96	37	6	6	10

is moving. Continuous HMMs are used for representing the output symbols with one Gaussian distribution on each state.

The Baum-Welch algorithm is used to train the observed data with parameters obtained by the method and basic HMM. Mouse input from zero to nine is used to the input gestures. Five data sets of each number are used for the training data. The number of state is fixed to six states. Each state is modeled as a multivariate Gaussian, since this is sufficient to treat the 2D data. First, the initial parameters are obtained by using spatial clustering phase of training phase in the method. For instance, Figure 4.10 shows the result of spatial clustering of training phase.

In this case, the parameters are obtained as follows:

$$\mu = \begin{bmatrix} 8.4595 & 6.7192 & 4.1371 & 0.5282 & 8.9299 & 4.0666 \\ 9.0429 & 1.6314 & 5.5213 & 7.3468 & 5.7536 & 9.8247 \end{bmatrix}$$

$$A = \begin{bmatrix} 0.9606 & 0 & 0 & 0 & 0.0394 & 0 \\ 0 & 0.9716 & 0 & 0 & 0.0284 & 0 \\ 0 & 0 & 0.9505 & 0.0495 & 0 & 0 \\ 0 & 0 & 0 & 0.9721 & 0 & 0 \\ 0 & 0.0215 & 0.0215 & 0 & 0.9571 & 0 \\ 0.0459 & 0 & 0 & 0 & 0 & 0.9541 \end{bmatrix} \quad (4.41)$$

It is required to check whether these parameters can represent the training data well. For doing this, these parameters are used for the initial parameters for the training of HMM. In order to compare with basic HMM training, other initial parameters obtained by GMM are used.  $\pi$  among the parameters is set to the the probability of each state =  $\frac{1}{N}$ .  $N$  is number of states. The experiments are repeated 10 times for each number gesture. Table 4.1 shows the number of iterations for training in each method. Almost every iteration number in the method is smaller than with the initial parameters obtained from basic HMMs except for three cases.

In addition the number of iterations in the method is small, and the log-likelihood of

Table 4.2: log-likelihood for each number data after training

	0	1	2	3	4	5	6	7	8	9
HMMs method	-3.1447	-0.8997	-3.0761	-3.4299	-3.3957	-3.5884	-2.4401	-1.8662	-3.9648	-2.4107
Proposed method	-3.1456	-0.8979	-3.0817	-3.3943	-3.3770	-3.5403	-2.4552	-1.8046	-3.9619	-2.3611

each method also shows similar results. After training of the Baum-Welch algorithm, the log-likelihood of each method is represented in Table 4.2.

Its unit is  $10^3$ . Seven of the ten cases shows better log-likelihood in the method. In the method, the parameters are already set before training by using the Baum-Welch algorithm, so the log-likelihood is always converged to same values. However, the experiments with initial parameters obtained by basic HMMs shows various results, as it has more free parameters than with the method. For this reason, sometimes it shows good results, when the initial parameters are properly set. However, bad results can be also obtained. This is because the training using the Baum-Welch algorithms is a method to find the local optimization, not global optimization. In the proposed method, the log-likelihood for each number data before training using the Baum-Welch algorithm has similar values with after training. This means that the parameters obtained by the method are properly set to represent the spatial structures of the input data.

### 4.3.2 According to Number of States

When the FSM approach or HMM is used to recognize the gesture input, this influences the performance, as the number of states depending on the structure of data are very large. Therefore, in this subsection, it is checked that the performance of the proposed recognizer depending on the number of states. In other words, it is discussed how performance would change while varying the number of states.

In this section, mouse gestures and visual gestures are used as the input for the experiments.

At first, fixed number of states are used to the represent the input number data(0-9). The number of states is changed from four to seven. Table 4.3 shows the results with various numbers of states.

It shows that recognition rates have better performances when the higher numbers of states are used to represent the input data. This is because when the input data is

Table 4.3: Mouse input results with various number of states

Number of states	train / test	0	1	2	3	4	5	6	7	8	9
four states	training data	100	80	100	100	80	100	100	100	60	100
four states	test data	100	80	100	100	100	100	100	100	40	100
five states	training data	100	100	100	100	100	100	100	100	40	100
five states	test data	100	80	100	100	100	100	100	100	80	100
six states	training data	100	100	100	100	100	100	100	100	100	100
six states	test data	100	80	100	100	60	100	100	100	100	100
seven states	training data	100	100	100	100	80	100	100	100	100	100
seven states	test data	100	80	100	100	100	100	100	100	100	100

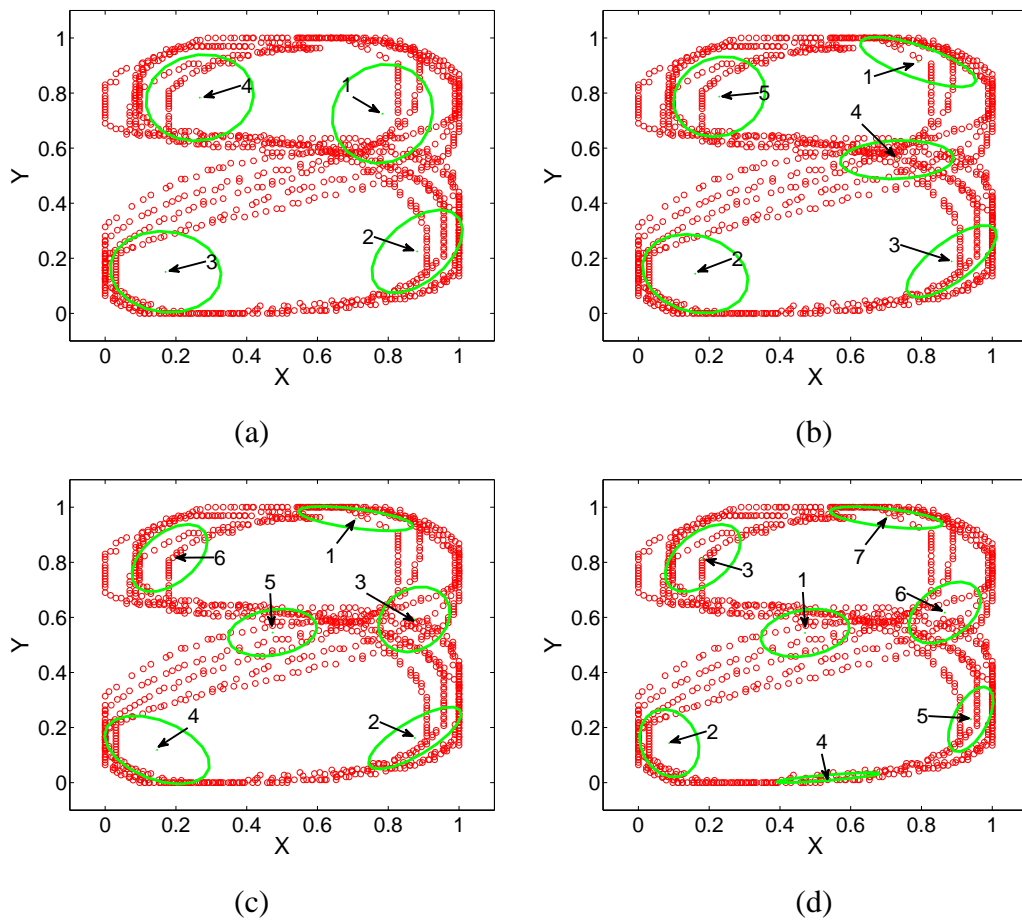


Figure 4.16: The state compositions of the input data according to number of states (a) Four states (b) Five states (c) Six states (d) Seven states

dispersed in the spatial space, the greater the number of states is positioned in space, and the better the input will be covered. Figure 4.16 shows this property well. When four or five states are used to represent the input data as in Figure 4.16(a) and (b),

Table 4.4: Total recognition rates for training set and test set of mouse input

Number of states	training set	test set
four states	92	92
five states	94	96
six states	100	94
seven states	98	98

Table 4.5: Total recognition rates for training set and test set of visual input

Number of states	training set	test set
four states	94	86
five states	95	90
six states	92	94
seven states	87	92

there are ambiguous moments. In case of four states, gesture "8" is redefined as  $1 \rightarrow 4 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 1$ . Ambiguous moments occur at the data point when the state is changed to 4 from 3. If data points are drawn a little down from the position at the interval, which has to change number 4 state from number 3 state, and the drawn data point is closer to the number 2 state than the number 4 state, the sequences of gesture are changed; thus, it is not recognized as the "8" gesture. The same problem happens in the case of five states. In case of five states, gesture "8" is redefined as  $1 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 5 \rightarrow 4 \rightarrow 1$ . At this time, if data points are drawn a little down from the position at the interval, which has to change number 5 state from number 2 state, and the drawn data point is closer to the number 4 state than the number 5 state, the sequences of gesture are different to the ordered sequence of gesture trained by the training phase, so it is not recognized as the '8' gesture. This is because the gesture drawn by each person has different trajectories, even if each person draws the same gesture.

The total recognition rates for training set and test set are as shown in Table 4.4 and 4.5.

If there are many states, the states are placed more densely in the space where the data points are scattered, so it is possible to avoid these error factors. However, if there are too many states, it will be overfitting to the training data, and thus the performance for test data set will worsen. Therefore, it can be presume that the many, but proper number of states can represent the input data well, and the better recognition rate will



Table 4.6: The performances comparison with HMM method, Basic FSM method and the proposed method

Method	Number of States	Total(%)
HMMs	3	79.0
	4	86.0
	5	94.5
	6	98.0
	7	92.5
Basic FSMs	3	75.5
	4	68.0
	5	83.0
	6	75.0
	7	77.5
Proposed method	3	83.0
	4	90.0
	5	92.5
	6	93.0
	7	89.5
	Incremental method	96.0
	MDL	96.0
	BIC	95.5

be thereby obtained.

Table 4.6 shows the performance comparison between the HMM method, the basic FSM method, and the proposed method for gesture recognition with visual input. When the HMM method is used with a fixed number of states, the results show that has good performance (higher than 79%), and the best performance is 98%. In contrast, the basic FSM method with a fixed number of states can only achieve 68 ~ 83% recognition rates. The proposed method can make the performances higher than 83% using a fixed number of states, and the best performance is 93%. In order to improve the recognition performances, the proposed method can use various numbers of states for representing each gesture. For doing this task, three different methods are used. First, Incremental method is used. In this method, the number of states is increased, when the average sum of distance between data points and states is larger than the fixed threshold. The fixed threshold is set by using heuristic method. This method shows the 96% success rates for recognizing the input gestures. Second, the estimation for the number of states is based on the Rissanen order identification criteria, which is also

known as minimum description length(MDL) (McGregor and Pallai, 1997). When the number of clusters is fixed, this criteria is equivalent to maximum likelihood estimation. However, it allows the number of states to be accurately estimated. In this case, 96% success rates for recognizing the input gestures are obtained. Third, the Bayesian Information Criterion is used to choose an optimal number of states (Schwarz, 1978; Calinon and Billard, 2005). It is also based on the likelihood function and the formula for the BIC is expressed as

$$BIC = -2\log(L) + k\log(n) \quad (4.42)$$

where  $L$  is the maximized value of the likelihood function of the model and  $k$  is the number of parameters to be estimated.  $n$  is the number of data points in the observed data. In this case, 95.5% success rates for recognizing the input gestures are obtained. Using these three different methods, therefore, each gesture can be redefined as the ordered sequences of states much better than with the fixed number of states. From these results, the proposed method is shown to exhibit similar performance with the HMMs method.

## 4.4 Summary of Chapter 4

In this chapter, it is tried to make a recognizing system that can handle the various mouse and camera hand gestures such as letters and numbers, using only visual input. For training or recognizing the gesture of hands, making a data set is needed. Therefore, these data sets were made using two different methods: mouse device and camera. When the mouse is used, the positions of the mouse cursor became gestures. When a single camera is used, the center positions of hand obtained by using several image processing methods are used for the input gesture. Skin-color detection using RGB color space is used to detect the hand in the video. After making the data set, these data sets have to be trained, because the data set should be trained to make the proper forms. Training is divided into two steps: spatial clustering and temporal alignments. From this process, some parameters to be used to make the recognizer model can be derived. Through the training phase, each gesture is also redefined as the sequences of states in spatial and temporal space. These redefined data sets are used to recognize the gestures with FSMs with probabilities.

# **Chapter 5**

## **Robustness analysis**

In this chapter, more specific analysis for the proposed method will be conducted through various experiments. Through this analysis experiments, the abilities of the proposed method and limits can be confirmed.

The proposed method is analyzed as to how it can recognize the various gestures through the comparison of the overlap in time and overlap in space. The reason for doing this experiment is to confirm the recognition performances of proposed method for the various overlap gestures. In addition, through the experiment, the gesture sets can be found that are most appropriate to be applied in the proposed method. When input data is entered, if the gestures have different time periods, even if they are the same gestures, whether recognition can be done or not is analyzed. In other words, the experiments will be performed with the same gesture data with various time periods. From this experiment, it can be confirmed that the proposed method can recognize the various gestures, which have different speed with different users. The proposed method doesn't need any extra training set or process to recognize the different speed gesture input. The advantages and disadvantages of the proposed methods and HMM are also explained.

The methods that are in Chapters 3 and 4 are combined to perform the experiments for the continuous gesture input, which is also called live video. This content is prepared to be published in a journal (Yim and Kim, 2013d).

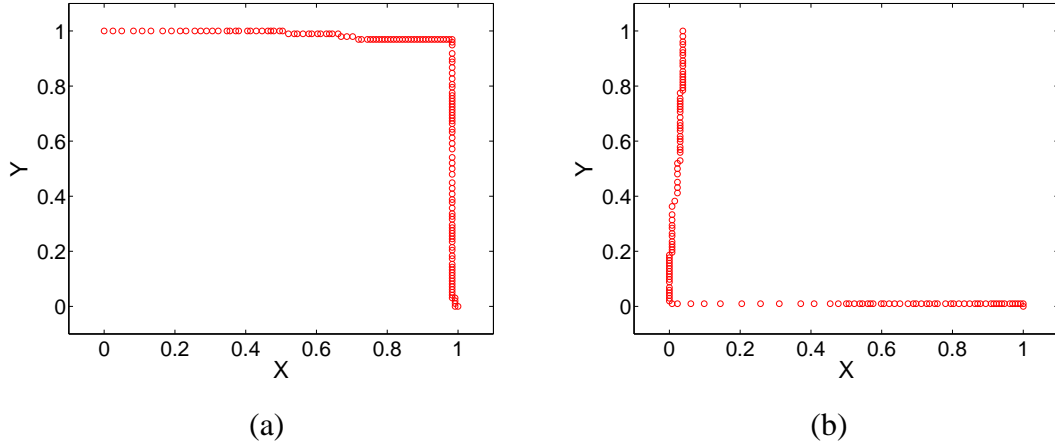


Figure 5.1: The gestures which have completely different distribution of spatial and temporal information

## 5.1 Overlap problem

Experiments are performed to confirm the recognition performances according to the overlap in space and time of gestures. Ten gesture data for each gesture is used for training and making recognizer, then the test is performed by using this recognizer. At first, it is confirmed that the proposed method can recognize the gestures that have completely different distributions of spatial and temporal information. These gestures show the completely different structures and forms as in Figure 5.1.

The experimental results show that these gestures are perfectly recognized by the recognizer. Next, it is tried to analyze the gestures that have exactly identical distributions of spatial information, but totally different temporal distribution. When the gestures are drawn as in Figure 5.2, it is confirmed that the gestures are properly recognized.

The experimental results show the 100% recognition success rates for training data sets and test sets. Through these results, if the gestures have different temporal distributions, even if the distributions of spatial information are matched perfectly, it was confirmed that the recognition is possible.

In case of Figure 5.3, when the recognizer is made, which is trained by using (a), the gestures that have same spatial distribution but different starting points are used for the input data of the recognizer. The experiments are performed with various starting points at intervals of 45 degrees. In other words, these gestures have the same distribution in space, while temporal sequences are slightly different. This experiment differs slightly from the above experiments, because the gestures that are used in this experi-

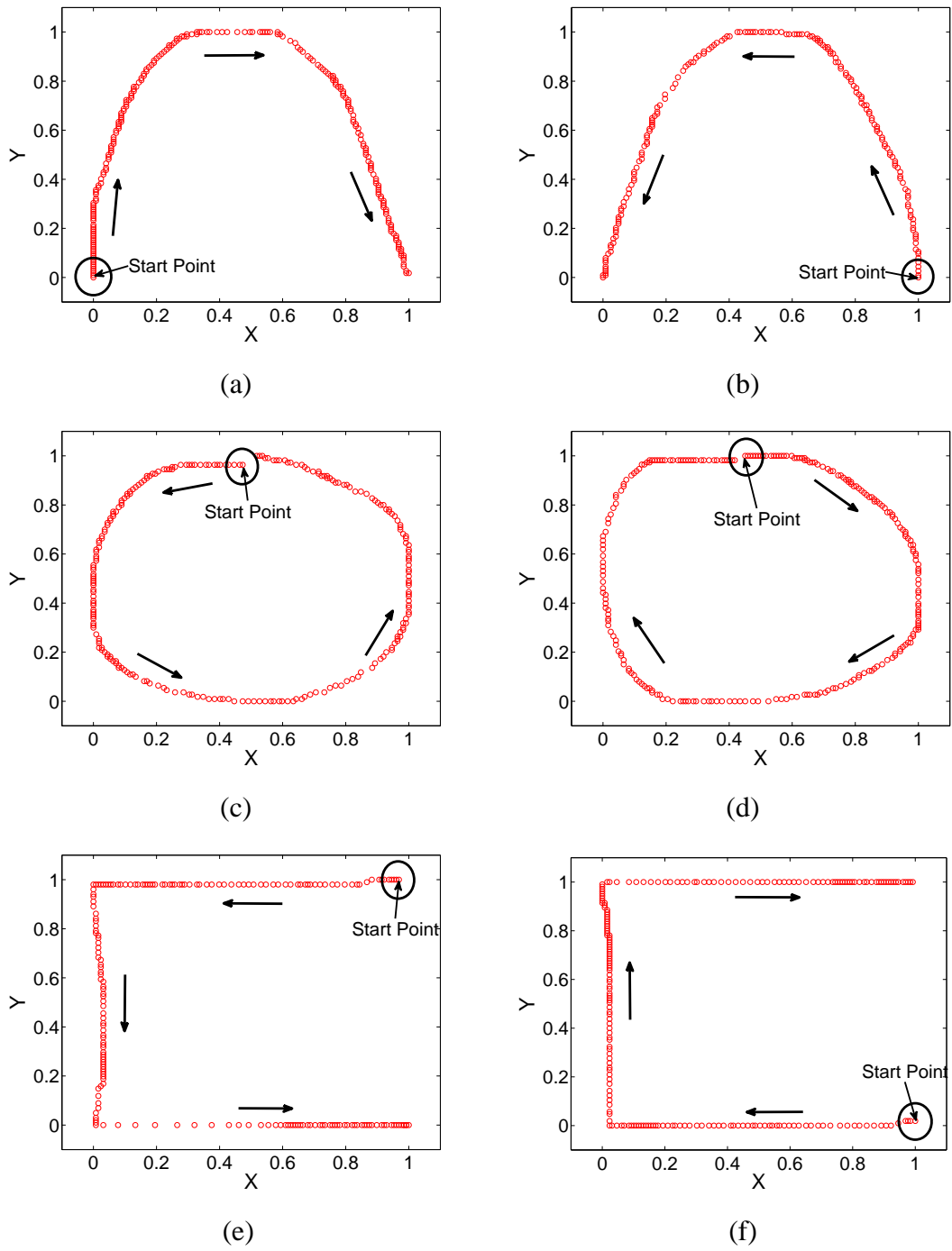


Figure 5.2: The gestures which have completely same spatial distribution, but are drawn in the other direction.

ment have some overlap in temporal distribution. If the number of states is small, then the gestures of Figure 5.3 (a) and (b) is well recognized as the same gesture. However, if there are many states to represent the spatial distribution of (a), then gesture (b) is also considered as a different gesture to (a). Gestures (c) ~ (h) were not recognized

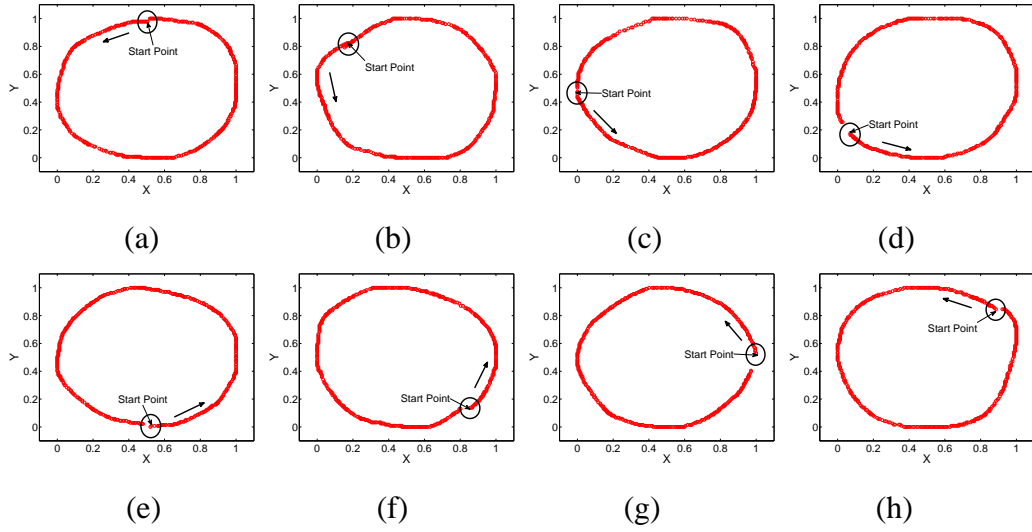


Figure 5.3: The gestures which have completely same distribution, but different start position.

gesture (a).

Next, the gestures that have similar distributions of spatial and temporal information are used to confirm the performance of the recognizer for the next experiments. First, the experiments are performed by using the gestures, which have 30 70% similarities for spatial and temporal information. The gestures in Figure 5.4 are the inputs that a few similar spatial and temporal distribution, but not as a whole. In other words, they are gestures that have the part drawn exactly the same phase.

In the case of (a) and (b) in Figure 5.4, considerable parts of the gestures are matched for spatial and temporal distribution. However, when the gestures are entered to the proposed method, the normalization process is performed for gesture setting to make a proper input to the recognizer. In the normalization process, the maximum and minimum values are used in x and y axes. Gesture (b) has the spoing part, so it affects the maximum value of the x-axis. For this reason, the spatial distributions of (b) are changed. Because of this normalization process, the gestures of (a) and (b) have considerably similar shape and distribution; however, the spatial distributions are changed differently after normalization. Therefore, these two gestures are perfectly recognized by using this property.

Figure 5.4 (c) and (d) are drawn perfectly identically in the first part of the gesture. However, the spatial and temporal distributions are changed, after drawing the first part of the gesture. These gestures can be considered as the gestures which have 33%

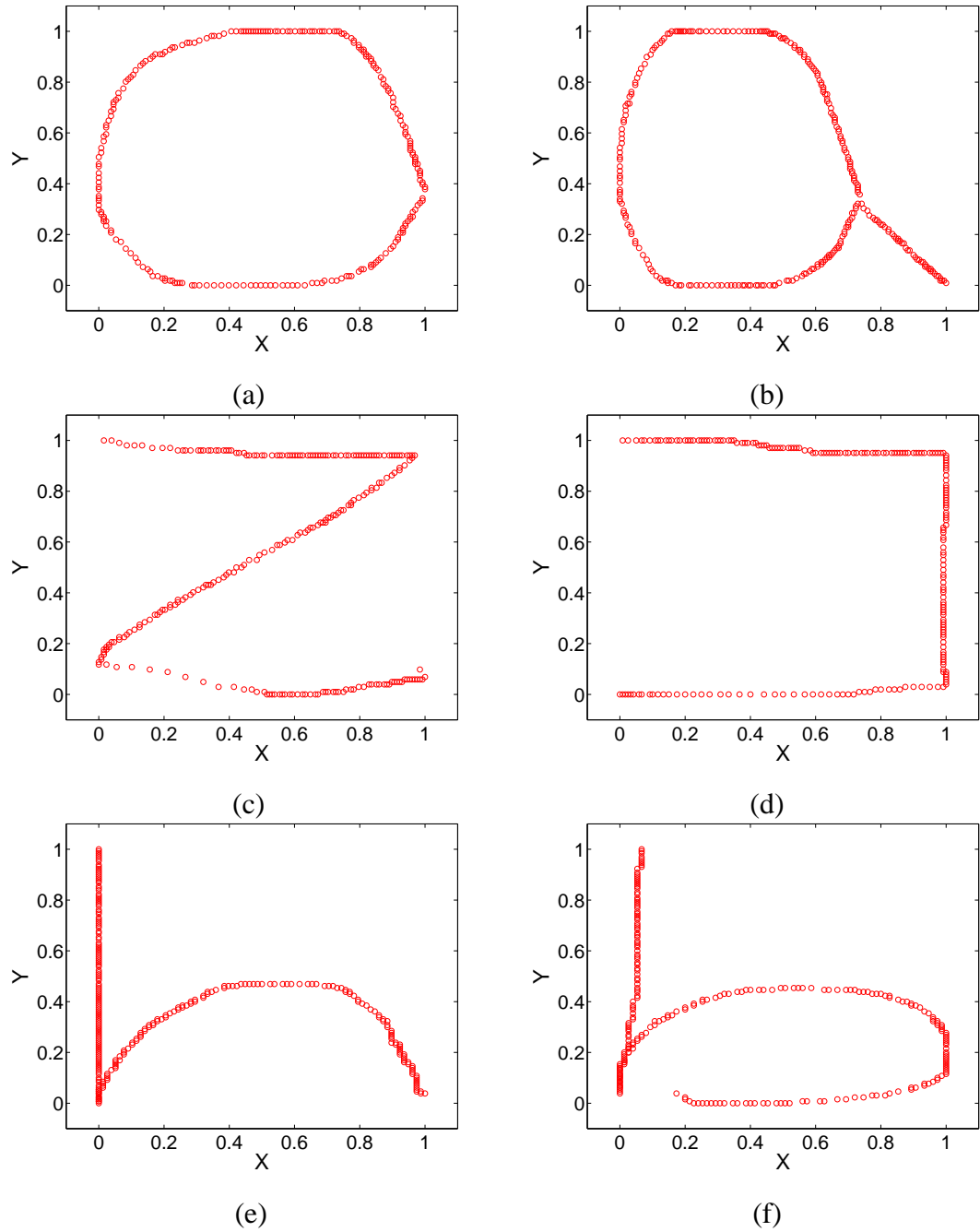


Figure 5.4: The gestures which have some few similarities for spatial and temporal distribution.

similarities. In the case of (e) and (f) in Figure 5.4, they have 60% similarities. They are drawn in exactly the same shapes in the two-third part of the gesture. Gesture (f) has the most spatial distributions at the last part of gesture. Except for this last part, gestures (e) and (f) are completely identical. When the trained recognizer is used, the training data sets and test data sets were perfectly recognized.

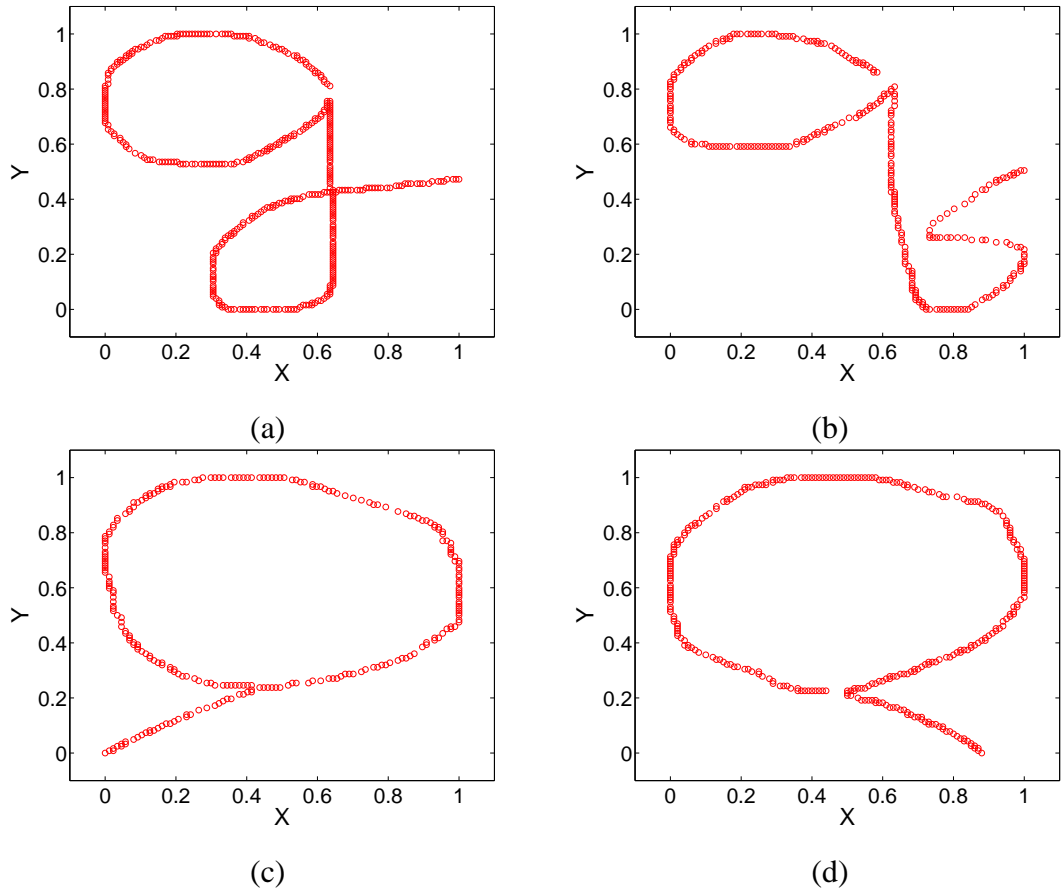


Figure 5.5: The gestures which have considerable similarities for spatial and temporal distribution.

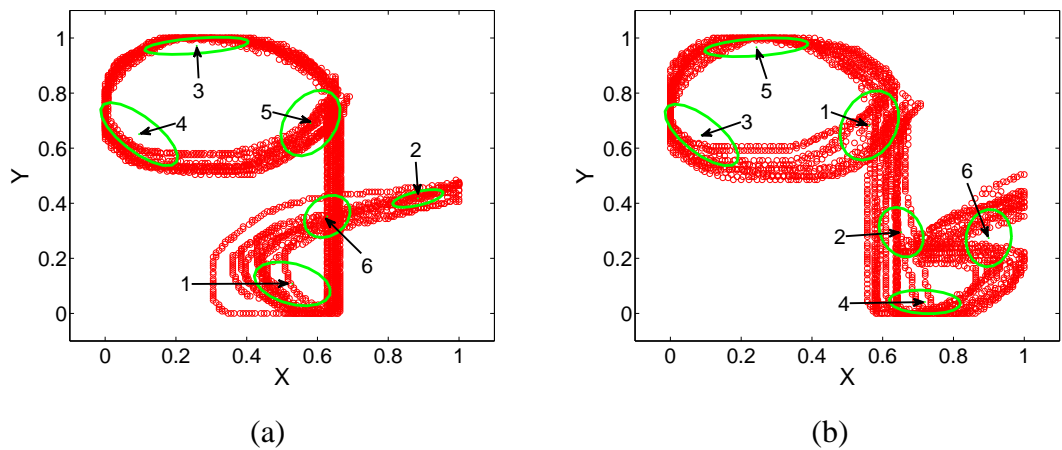


Figure 5.6: The state structures of (a) and (b) in figure 5.5

Figure 5.5 (a) and (b) show the gestures that have similarities more than 80%. Even if they have considerably similar distributions, the states generating at the last part of gestures are made at different location, as in Figure 5.6, so they are well recognized.



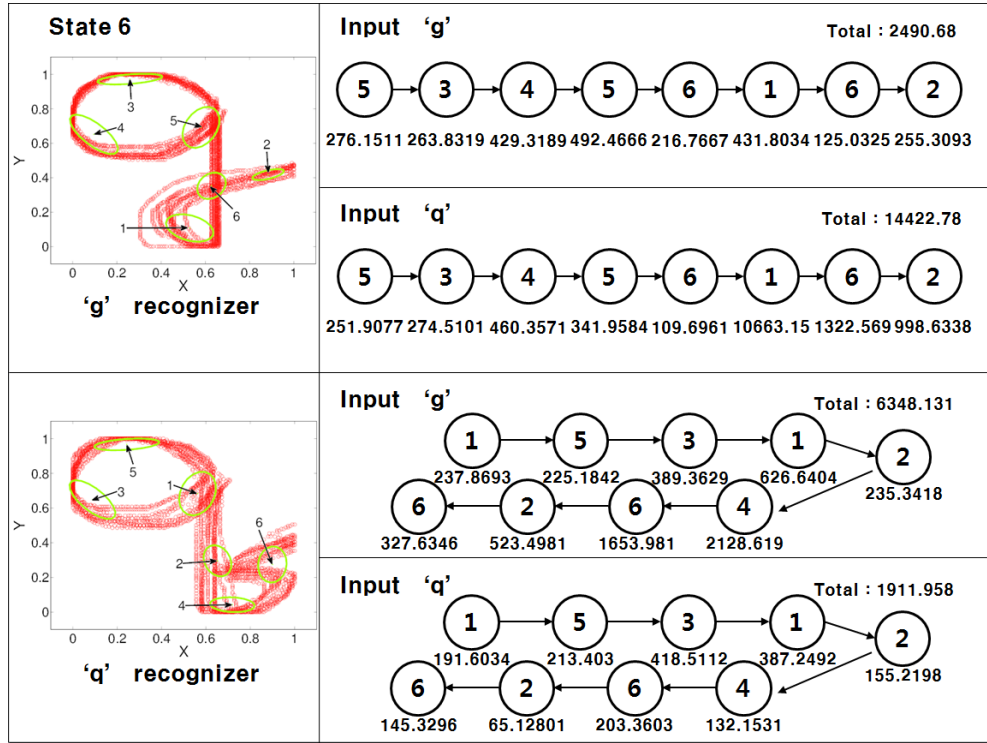


Figure 5.7: The mahalanobis distance for each state and total distance.

As shown in Figure 5.6, the first five state sequences are the same in gestures (a) and (b). However, the ordered state sequences at the last parts of the gestures are different. In the case of Figure 5.6 (a), it has  $6 \rightarrow 1 \rightarrow 6 \rightarrow 2$  state sequences at the last part of gestures; however, in case of (b), it has  $2 \rightarrow 4 \rightarrow 6 \rightarrow 2 \rightarrow 6$  state sequences. For this reason, these gestures are well recognized, even if they have similar distributions. Figure 5.7 shows this property well with various numbers of states.

Figure 5.7 represents the sum of the squares of Mahalanobis distance for each state. As shown in Figure 5.7, the distances between ordered state sequences and data points are similar before the last part of the ordered state sequences of gestures. In Figure 5.7, the last part of the ordered state sequences are  $1 \rightarrow 6 \rightarrow 2$ . In this part, the 'g' input still has a similar sum of distances with other order ordered state sequences. However, in 'q' input cases, the distances are increased tremendously. This means that if the gestures have slightly different shapes and are trained with the proper number of states, then it can be recognized well, even when they have similar distributions. The recognition success rate is 95% in training data sets and 90% in test data sets.

Figure 5.5 (c) and (d) show the gestures which have similarities more than 90% in contrast with (a) and (b). In figure 5.5, these two gestures have similar ordered state

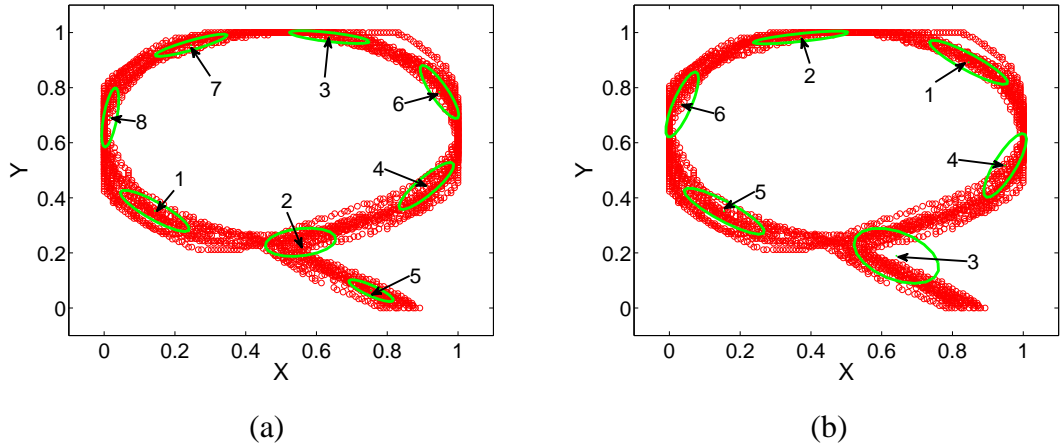


Figure 5.8: The state structures of (c) and (d) in figure 5.5

sequences, so the problem occurs more than in (a) and (b).

If the clusters are properly generated at the last parts of gestures, as in Figure 5.8 (a), then these two gestures are well recognized. However, if the clusters are ambiguously generated, then it is difficult to recognize. For instance, if the clusters are generated as in Figure 5.8 (b), then it is difficult to recognize. In this case, these two gestures have the same ordered state sequences, so the noisy gestures may be misrecognized. For this reason, it shows the 95% recognition success rates in training data sets and 85% recognition success rates in test data sets. In order to confirm the results of these experiments in more detail, the Mahalanobis distance was checked for various numbers of states.

Figure 5.9 shows the average Mahalanobis distance for one data point with various inputs in a recognizer. For example, the first one of Figure 5.9 is the case of using a 'g' recognizer with various inputs. It shows that the trivial difference inputs are almost rejected since they cannot pass through the whole ordered sequence of states or they don't stop at the accept state. These cases are represented as a star shapes in Figure 5.9. 'g' and 'q' have similar shapes; thus, when small number of states are used for representing the gestures, similar average distances are shown. Therefore, the distance differences between each input gesture are not large. However, when a larger number of states is used, then the distance differences are increased. This is because the larger the number of states, the better the training data will be represented. Hence, the average distances of 'g' input gestures in the 'g' recognizer scarcely changes, while the average distances of 'q' inputs in the 'g' recognizer are increased. However, if things come to

	2 state	3 state	4 state	5 state	6 state	7 state	8 state	10 state	20 state
g recognizer	2	3	4	5	6	7	8	10	20
¬ input	7.87	★ 10.68	★ 12.16	★ 13.62	★ 16.23	★ 24.05	★ 27.91	★ 38.38	★ 81.03
b input	★ 7.54	★ 7.03	★ 7.73	★ 11.83	★ 12.59	★ 22.75	★ 13.02	★ 26.62	★ 32.84
g input	2.12	2.17	2.14	2.19	2.26	2.31	2.39	2.58	3.24
q input	2.60	2.62	3.12	3.34	3.82	6.26	3.64	8.14	10.02

	2 state	3 state	4 state	5 state	6 state	7 state	8 state	10 state	20 state
q recognizer	2	3	4	5	6	7	8	10	20
¬ input	6.12	★ 9.88	★ 10.27	★ 13.40	★ 10.75	★ 13.18	★ 17.71	★ 18.84	★ 36.30
b input	★ 7.19	★ 6.68	★ 9.08	★ 13.77	★ 14.32	★ 17.03	★ 16.80	★ 17.85	★ 41.55
g input	2.69	2.90	2.95	3.06	3.15	★ 3.29	★ 3.36	3.52	★ 5.15
q input	1.74	1.77	1.75	1.79	1.75	1.79	1.79	1.86	1.95

	2 state	3 state	4 state	5 state	6 state	7 state	8 state	10 state	20 state
LQ recognizer	2	3	4	5	6	7	8	10	20
¬ input	★ 3.26	★ 3.99	★ 4.12	★ 10.00	★ 7.73	★ 13.66	★ 10.92	★ 14.66	★ 15.35
b input	★ 2.82	★ 4.30	★ 6.75	9.60	★ 10.52	★ 13.26	★ 15.23	★ 21.38	★ 25.52
LQ input	1.68	1.70	1.70	1.64	1.62	1.60	1.53	1.52	1.55
RQ input	2.60	2.53	★ 2.71	3.13	3.51	★ 3.73	★ 4.64	4.18	5.56

	2 state	3 state	4 state	5 state	6 state	7 state	8 state	10 state	20 state
RQ recognizer	2	3	4	5	6	7	8	10	20
¬ input	★ 3.79	★ 4.04	★ 5.06	★ 5.54	★ 8.53	★ 8.48	★ 11.49	★ 13.78	★ 22.44
b input	1.77	★ 3.29	★ 4.83	6.84	8.77	10.92	19.48	21.15	★ 36.96
LQ input	2.13	1.95	★ 2.21	2.59	3.16	3.60	3.07	3.58	6.11
RQ input	2.07	2.06	2.05	2.06	2.06	2.06	2.06	2.04	2.05

Figure 5.9: The average Mahalanobis distance for one data point with various inputs in a recognizer. ★ shape means that the gestures are already rejected. 1. 'g' gesture recognizer 2. 'q' gesture recognizer 3. 'LQ' gesture recognizer 4. 'RQ' gesture recognizer

this, the state distributions are too closely fit to the training data; thus, it is increasingly vulnerable on the noise input. Therefore, there is tradeoff between the number of states and performances. Other gesture recognizers show the same properties. Figure 5.10 represents these values in table as the graphs.

If the square of Mahalanobis distances is used, then these properties will be more reliable.

When the gesture data was trained, the mean and covariance of each state were continuously changed. Therefore, it can also give the effect to the recognition. The experiments were performed to confirm this effect, and it was presumed that the average distances of gesture input in the same gesture recognizer with input show no change, while other gestures are influenced by the clustering. Figure 5.11 shows the result of these experiments. Through the results, it was confirmed that the average distances of gesture input in the same gesture recognizer are almost the same, while the average distances of other gestures have large variances.

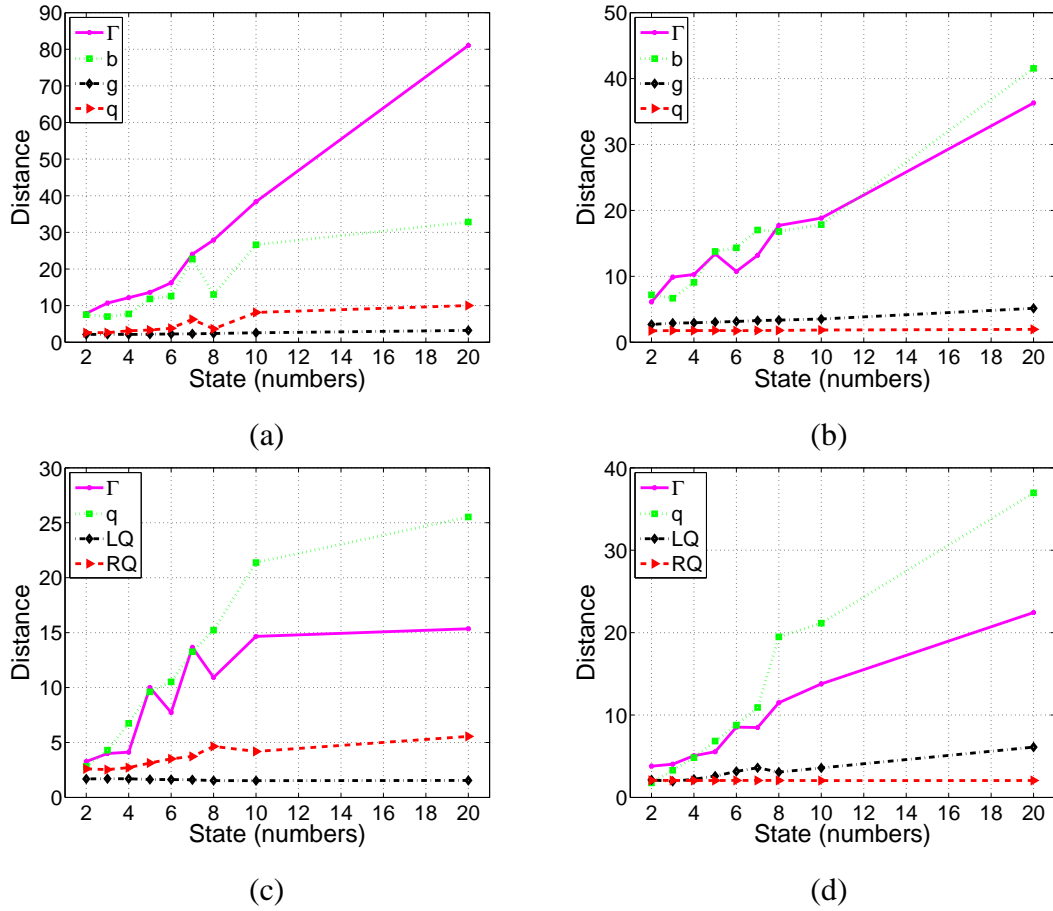


Figure 5.10: The average mahalanobis distance for one data point with various inputs in a recognizer. (a) 'g' recognizer (b) 'q' recognizer (c) 'LQ' recognizer (d) 'RQ' recognizer

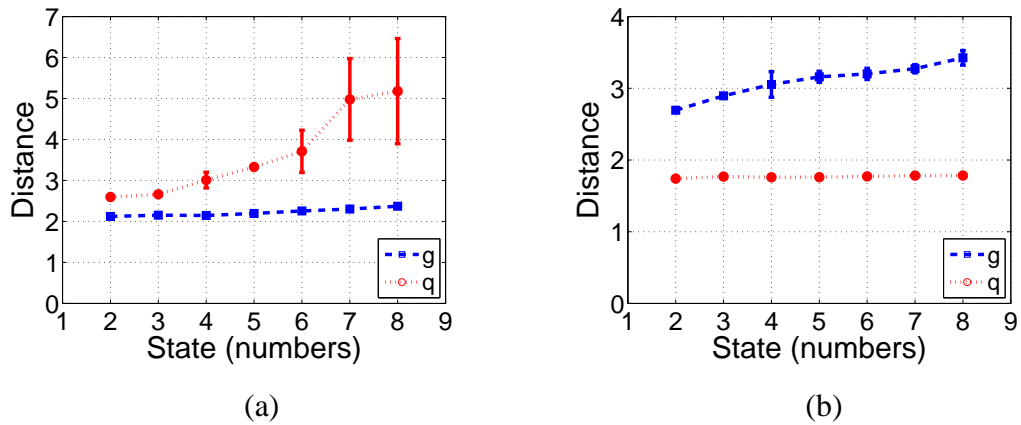


Figure 5.11: The average mahalanobis distance for one data point with several clustering steps. (a) 'g' recognizer (b) 'q' recognizer

Experiments for Figure 5.1, 5.2, 5.4 and 5.5 gestures were performed using HMMs recognizer. The same numbers of training sets with FSMs experiments are used. The

results of HMMs recognizer show the different properties to the FSMs recognizer. First, The gesture of Figure 5.1 (a) is perfectly recognized in HMMs recognizer. However, in case of (b), it is miss recognized as the gesture of Figure 5.2 (e). It shows the 90% recognition success rates in test data sets. All of gestures in Figure 5.2 (a), (d), (e), (f) are recognized well. However, in case of (b) in Figure 5.2, it is miss recognized as Figure 5.4 (a). it shows the 100% recognition success rates in training data sets and 90% recognition success rates in test data sets. Figure 5.2 (c) is confused with Figure 5.4 (a). Figure 5.4 (a) has the similar spatial distribution with Figure 5.2 (c), but they have different starting points. it shows the 95% recognition success rates in training data sets and 95% recognition success rates in test data sets. In the inverse case, it shows the 100% recognition success rates in training data sets and 70% recognition success rates in test data sets. Another miss recognition case is Figure 5.4 (e) and (f), and it shows 95% recognition success rates in test data sets. In contrast with FSMs recognizer, all of gestures in Figure 5.5 are perfectly recognized.

From these results, it can be confirmed that spatial and temporal overlap in HMMs is also important factor, but not much in FSMs. HMMs recognizer determines the input gestures to the gesture which has similar pattern through the probability. The likelihood of miss recognizing case above the experiments shows slightly difference with true gesture. It means that there are many number of parameters which are needed to be set, so training data required is large to obtain the good recognition performances.

In addition, it is confirmed that the proposed method can distinguish the gestures that share similar shapes, such as 'g' and 'q', and 'b' and 'h' in addition to trivially different gestures. If the spatial clustering works properly to represent the gestures as the ordered sequence of states, the performance will be high.

## 5.2 Different time period problem

The input data that is used for the gesture recognition has some possibilities that have different time periods according to the speed of the user's gesture. In other words, the number of input data points can differ with different users. To solve these problems, HMM have to use the many training data sets that have the same gestures with various time periods for training. However, the proposed method does not need many training data sets to recognize the various time period gestures. The proposed method can solve

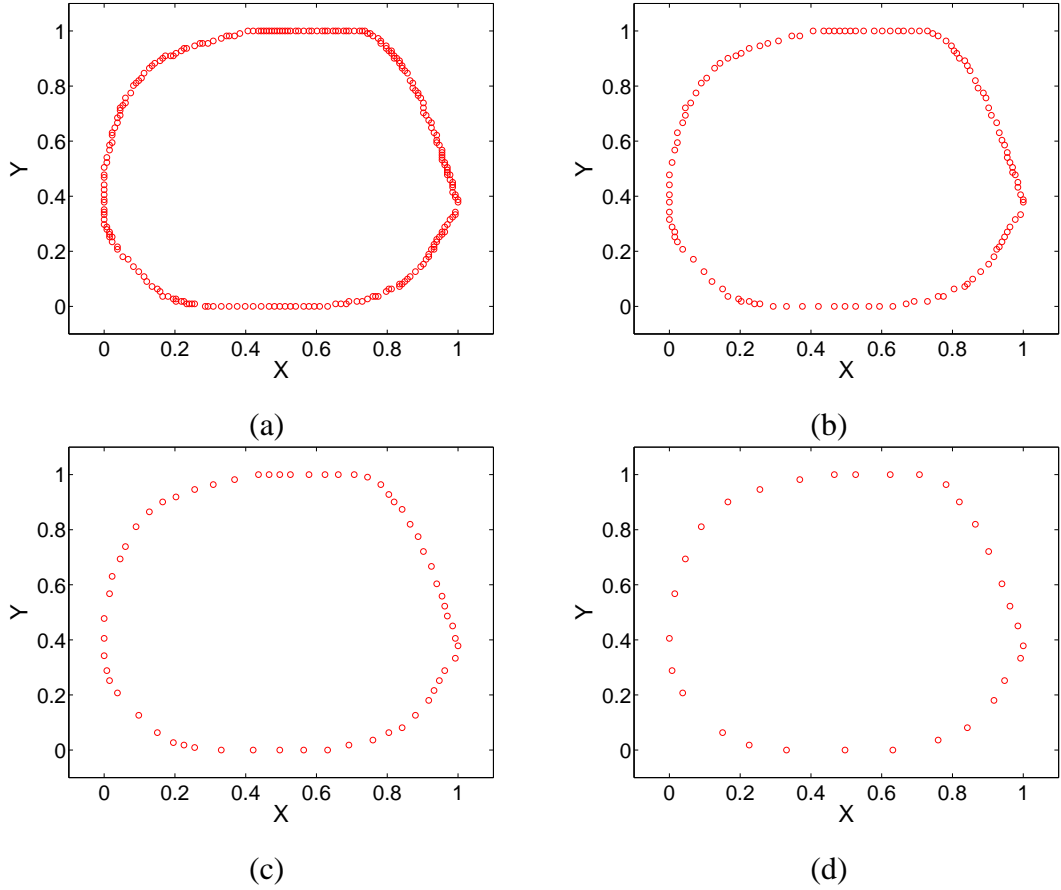


Figure 5.12: The same input gestures that have different time periods (Uniform case)

the problem, even if the gestures have different time periods. When the recognizer is generated by using training data in the proposed method, the spatial distributions of data are very important.

The states, also called clusters, are generated according to the spatial distributions of data using training phase, and using this information a recognizer is generated. Therefore, after generating a recognizer, if the input gesture has passed accurately on the states that make up the recognition, then the time period of the input data is irrelevant. For the experiments, test sets are made by uniform sampling in the input gestures. As shown in Figure 5.12, the same gestures that have different time periods are used. The experimental results show that the input gestures are recognized as the same gesture if they have passed in regular sequence on the ordered states even if the input gestures have different time periods. In other words, the gesture inputs of Figure 5.12 (a) (d) passed through the ordered state sequences that are already trained in regular sequence, and, thus, they are considered as the same gesture. However, if the sampling frequency becomes low, the input gesture data cannot pass through the

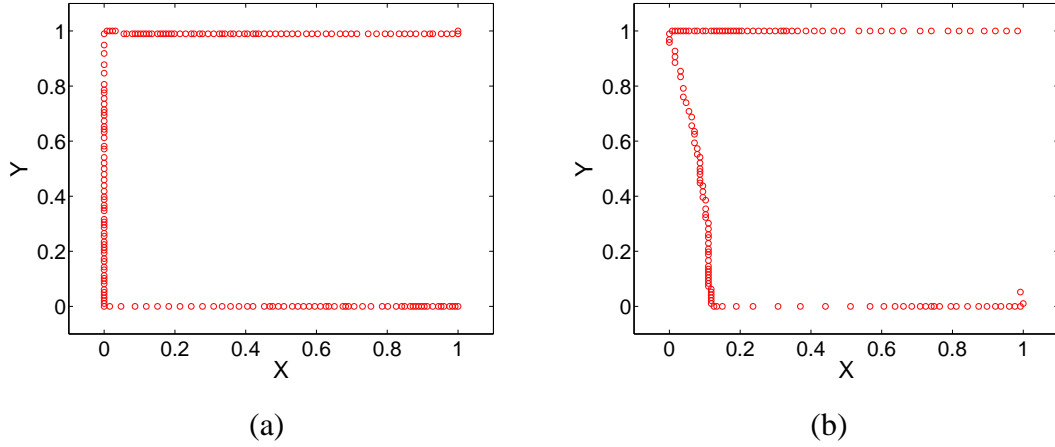


Figure 5.13: The same gestures that have the different input speed

ordered state sequences, so the gesture is not recognized. This property is applied equally to the input gestures, which have different input speeds, as in Figure 5.13, not sampling data. These gestures are also recognized well.

In order to confirm the effect of input speed professionally, the gestures were used, which have various input speed as in Figure 5.14.

There are input gestures that have various input speeds, as in Figure 5.14. In the case of (a), a gesture is entered to the normal input speed while only 25% is entered to the fast speed in the whole gesture in case of (b). Gestures (c), (d), (e), (f), and (g) have input speeds in which 50% of the whole gesture is entered at fast speed and other parts are entered at normal speed. In case of (h), 75% of the overall gesture has fast input speed. Gesture (i) has fast input speed in the whole gesture, unlike gesture (a). Although the gestures have various input speeds, they can pass through the ordered state sequences trained by gesture (a), so it is considered the same gesture. In the method, it doesn't need any extra training set or process to recognize the different speed gesture input.

In addition, the proposed method can distinguish the same gesture performed at different speeds. Figure 5.15 shows the same gesture performed with different speeds in specific parts.

These sorts of data was trained and it was checked that each recognizer for Figure 5.15 (a), (b), and (c) can recognize the data computing log-likelihood of data. It was called (a) data1, (b) data2, and (c) data3.

Figure 5.16 shows the log-likelihood values for each data in each recognizer with various numbers of states. As the figure shows, the proposed method can also distinguish

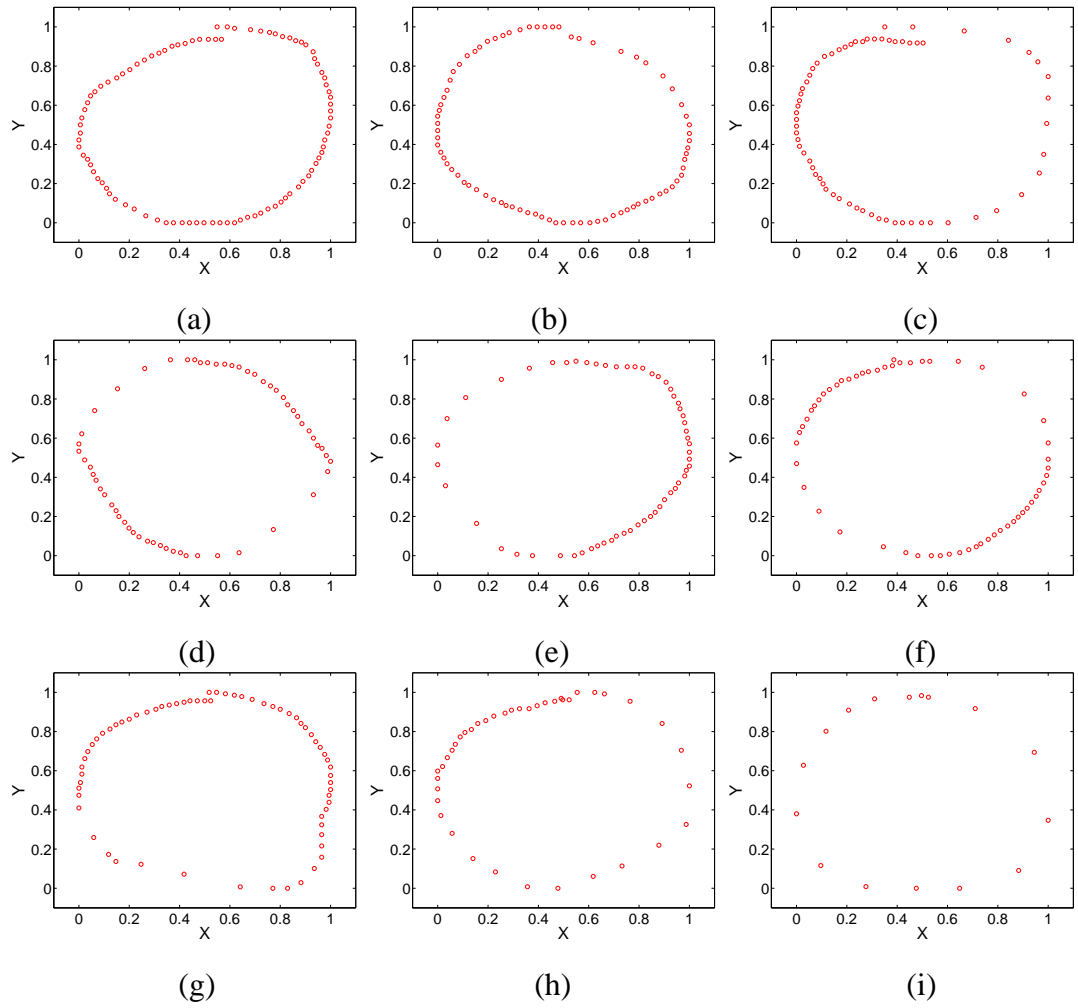


Figure 5.14: The same gestures which have the various input speed (Not uniform case)

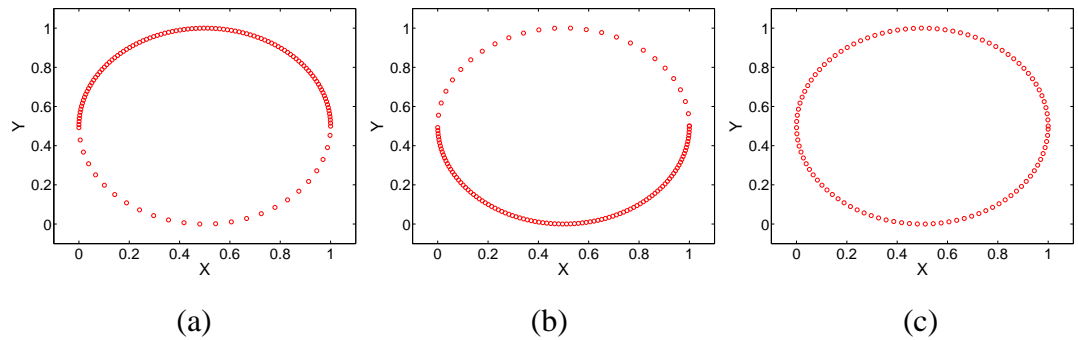


Figure 5.15: The same gestures which have the various input speed in specific parts

the same gesture with different input speed as different gestures. However, when the gesture recognition systems are used to applications in real life, there are no cases that distinguish the same gestures with different input speeds as the different gestures. In contrast, there are many cases that have to be determined the same gesture even if it



data1 recognizer	2 state	3 state	4 state	5 state	6 state	7 state	8 state
data1	-4.31508	-3.21292	-2.399	-1.80853	-1.28471	-0.80241	-0.39245
data2	-5.38712	-4.06672	-3.46078	-2.8428	-2.05788	-1.84501	-1.50344
data3	-4.85261	-3.64767	-2.92881	-2.33114	-1.69105	-1.33164	-0.95637

(a)

data2 recognizer	2 state	3 state	4 state	5 state	6 state	7 state	8 state
data1	-5.38638	-4.29358	-3.46175	-2.86529	-2.35468	-1.84622	-1.25601
data2	-4.31532	-3.32881	-2.3985	-1.80426	-1.26973	-0.80233	-0.45233
data3	-4.85171	-3.81463	-2.9291	-2.33172	-1.81493	-1.33221	-0.85903

(b)

data3 recognizer	2 state	3 state	4 state	5 state	6 state	7 state	8 state
data1	-4.52803	-3.61362	-2.77092	-2.14466	-1.59895	-1.18302	-0.80783
data2	-4.5316	-3.49094	-2.77607	-2.14333	-1.63464	-1.15189	-0.73775
data3	-4.52909	-3.54982	-2.76825	-2.13463	-1.60394	-1.14992	-0.76708

(c)

Figure 5.16: The log-likelihood for each data input (a) data1 recognizer. (b) data2 recognizer. (c) data3 recognizer.

has different input speed.

### 5.3 Combined method for continuous recognition

In the previous section, the performances of the proposed method are confirmed with individual gestures which are already saved into the data. This section shows a new method that combines the methods of Chapter 3 and 4 for recognizing the continuous gesture in real-time. The hand shape classifier, explained in Chapter 3 and the hand gesture recognizer based on FSMs in Chapter 4 are used together in this chapter. To recognize the continuous gestures in real-time, at first, the hand postures have to be distinguished to confirm whether the gesture is or not. Hand postures are classified by using the proposed method introduced in Chapter 3.

As explained above, classification of various hand postures is done by using only visual input obtained from a single camera. For doing classification, the proposed method is composed of three parts. The first one is hand information extraction through skin detection using RGB color values. From that, the edge of the hand can be obtained using a canny edge filter and the center of hand. Through several preprocessing steps, the HCV is calculated by using distance between center of hand and edge of hand. The image information is changed into this distribution based on distance. The final process is classification comparing this reference vector and current image obtained



Figure 5.17: combined method for continuous hand gesture recognition (a) Rock posture is considered as the gesture (b) Paper posture is considered as the meaningless movement

from the camera. The detail processes are introduced in Chapter 3.

It is desired to recognize the continuous input gestures using this classifier for the hand postures. Two hand postures, rock and paper, were used for the experiments. If the proposed classifier determines the hand of input gesture to rock, then the center of hand in the input gesture is saved into the array in Matlab. In other words, when the input data is decided as the rock posture, the system recognizes this input as gestures. While the center of hand is saved into the array, if the classifier determines the hand of input gesture to paper, then the saving process stops and the recognizing process begins using the data already saved. The inputs that are decided as paper posture from the classifier are not considered as meaningless gestures. Figure 5.17 shows this process. When the hand posture is rock, as in Figure 5.17 (a), the hand movement is considered as the gesture and saves into the array. However, in the case of Figure 5.17 (b), the hand movement is considered as the meaningless movement, so there are no processes to save or recognize during this case.

When a classifier operates, there are moments to change the hand posture from paper to rock or from rock to paper. The hand image is not obtained exactly at these changing moments, as in Figure 5.18. At this time, since the exact shape of the hand is not obtained, the hand posture is classified by matching the current image and reference images with the HCV values. In other words, the one current vector set is matched to the most similar one in the reference sets. In the case of Figure 5.18 (a), where the hand posture is classified as the paper shape, it is not considered as the gesture. It



Figure 5.18: The moment to change rock from paper (a) it is classified as paper (b) it is classified as rock

is just meaningless movement of the hand. In contrast, since Figure 5.18 (b) is more similar to the rock reference than paper reference, it is considered as the rock posture, so it is regarded as the gesture.

In addition to these cases, when the hand that actually has a paper posture makes some movements in the space, the hand image sometimes fades, so it is hard to extract exactly. This is because when the hand image fades, the color of finger becomes similar to the color of the background. For this reason, it is sometimes considered as rock posture, even if it is actually paper posture. However, this phenomenon occurs pretty rarely. In other words, when the hand makes an actually paper posture, the hand posture is occasionally misclassified as rock posture while it should be continuously classified as paper posture. This problem is called as paper-fading problem in this paper. A median filter is used to solve this problem. Median filter operates to pick up the middle value of neighboring values, except for itself, using a moving window, the size of which is five frames. Four frames are located before the current frame, and one frame is the current frame. When the value of the current frame is decided by median filter, only four frames, which are located before the current frame, are used to compute the middle values. In this case, the hand postures are represented; rock and paper; as 0 and 1. Therefore, when the value obtained by median filter is larger than 0.5, then the current frame is decided as 1, rock posture.

Figure 5.19 shows the method of median filter operation.

Figure 5.19(a) shows the paper-fading problem. A window five frames in length is used to modify the original input, and it only moves on the original input. The current frame

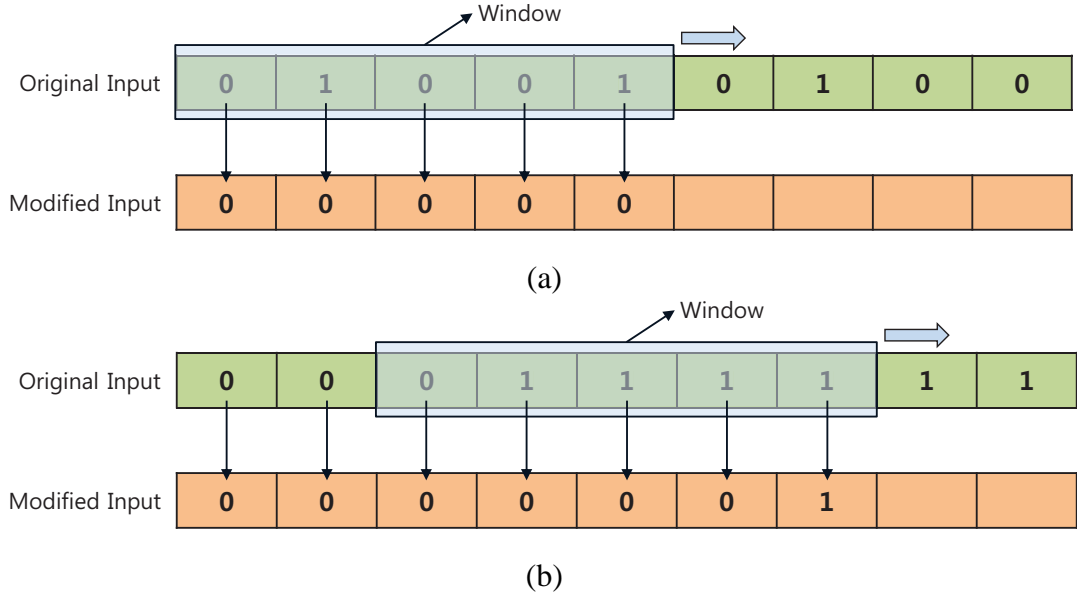


Figure 5.19: The operation method of median filter to solve the 'paper' hand posture fading problem

is determined to the image, which has rock posture, 1; however, this is a misclassification. The four frames that are already passed are used to modify this problem. The average value of four frames is 0.25; thus, the value of the current frame is decided as 0, paper posture. When the classifier works correctly, as in original input in the Figure 5.19 (b), the median filter makes the input move three frames back. In this case, the average value of four frames in the window is 0.75, so the value of the current frame is decided as 1, rock posture. To move three frames back is not a critical problem to save the gesture into the variable, so it is not necessary to do this.

Through these processes, the more stable classifier can be obtained, which can distinguish meaningful movement and meaningless movement. After that, a variance that is saved to the gesture is recognized by using the proposed recognizer introduced in Chapter four. Training is already done by using each gesture that has to be recognized. Test sets are continuously made one by one by the combined method, explained in the present chapter.

If test sets are exactly extracted from the video through the proposed method, the performances are identical with the method that was introduced earlier. This is because the same recognizer that is trained by the same training data sets is used to test the various input data obtained by the proposed combined method. It is confirm that the recognizer can also recognize well the gestures obtained by this method.

## 5.4 Comparing with proposed method and HMMs

In this section, the advantages of the proposed method and differences with HMMs which have been widely used for modeling the temporal sequences of data are introduced. The proposed method uses FSMs with probabilities to model the gesture recognition. There are many similarities between the proposed method and HMMs. First, the structure of FSMs is similar to the left-right model without jump transition among many structures of HMMs (Hong and Huang, 2000). The left-right model has good properties for dealing with time sequential data. Second, in this paper, each state is modeled as a multivariate Gaussian distribution since it is sufficient to treat the 2D data. HMMs can also use the multivariate Gaussian for each state, even if the number of Gaussian distribution in each state have to be set at first in HMMs. Each Gaussian distribution has parameters  $\mu$  and covariances to represent the spatial information. However, many free parameters of HMMs, which have to be adjusted in training, prevent a rapid learning (Hong et al., 2000a).

Recognition processes in HMMs have many complex steps, such as segment input data, computing joint probability (which is related to the input data), comparing the likelihood with fixed threshold, and decision. Therefore, it requires high computation complexity. On the other hand, FSMs have less calculation requirements and more certainty, as the spatial information training and temporal information training are divided, and it has only a few free parameters. In the recognition phase, current input data is handled immediately at each state using the information and threshold trained already in the training phase. This reduces the computation complexity.

In HMMs, the number of states and the structure (ergodic, left-right, etc.) have to be set at first for training, and it is also needed well-aligned data segments. In proposed FSMs case, the training data is segmented and aligned, and the gesture model is generated simultaneously. Another advantage is that, in order to train the gesture model, the proposed FSMs do not require large data sets; on the other hand, since HMMs have to find high probability statistically at a certain state, a large data set is required in order to increase the accuracy.

In the method, a different time period of the same input gesture is not relevant to recognize gestures. This is because if the data points of input gesture are positioned properly at each state obtained by training, it recognizes the relevant gesture, even if the hand gesture moves fast. Only one camera is used to make the hand gesture data

set, so it doesn't have any problem related to the matching problem of images.

The proposed method is successfully tested on the test set of hand gestures captured from video and mouse gestures. However, in the method, if the gesture data is ambiguously entered, they may be misrecognized. When the gesture data is ambiguously entered, each point of data is difficult to determine which states are the correct classes for each point in the method. In addition, if input data have passed in a different sequence on the state of recognizer, the recognizer is reset and it is not recognized. On the other hand, in HMM, the forward-backward algorithm or Viterbi algorithm is used to compute probabilities for ambiguous input data, so it can find the most similar model. Of course, the final computed probability is small; however, if it is larger than the fixed threshold, then ambiguous input data is also recognized. Therefore, it was confirmed that the proposed method requires more rigid conditions to input the gestures.

## 5.5 Summary of Chapter 5

In this chapter, experiments were performed with various sort of gesture input using a mouse and camera to confirm the performance of the proposed recognizer. The proposed recognizer shows good recognition success rates as in HMM models. Experiments were performed with overlap perspective about spatial and temporal information, and time period perspective. In the proposed method, the distribution overlaps of spatial and temporal information are allowed in great measure. It shows good performance, even if the gestures are considerably similar.

The results show that if temporal distributions are different, the overlapping degree of spatial distribution is no matter. The proposed method can also cover the significant overlapping level of temporal distribution.

In addition, the proposed method is confirmed to handle the gesture data with various time periods without many training data sets. From this experiment, it can be confirmed that the proposed method can recognize the various gestures, which have different speed with different users. The difference properties between proposed method and HMMs recognizer were also explained.

Lastly, the method for recognizing the continuous gesture in real-time is proposed. This method combined the methods presented in Chapters 3 and 4.

## **Chapter 6**

# **Conclusions**

In this dissertation, studies of a state based approach to gesture recognition have been presented. Since there are many potential applications, related to tablet PCs, televisions, smartphones, virtual reality, and so on, such research has been actively increased.

The work could be classified into two groups: hand posture recognition and hand gesture recognition. The hand posture recognition part concentrates on classification of different hand postures, such as ASL, while the hand gesture recognition concentrates on making state based recognizers that can recognize the various hand gestures from the video.

Chapter 3 describes the detailed methods of the hand posture recognition system. Chapters 4 and 5 provide the detailed methods of the hand gesture recognition systems and corresponding results of experiments. The experiments were conducted by using the input data obtained from a web camera.

The detailed conclusions are described in the following subsections.

### **6.1 Hand posture recognition**

In Chapter 3, the classification system for hand postures is shown. This research focuses on the recognition of American Sign Language (ASL) finger-spelling alphabet based on vision using a cheap webcam without using any expensive devices such as gloves or marking.

Three parts have been used for doing this task. First is the "Hand Extraction", which is designed to find the hand from the image for finding the center of the hand and distribution of distance between center and edge pixel. This distribution is called the Hand Contour Vector (HCV) set in this paper. Then the image is changed into one HCV set. In the second part, the process of data collection progresses using the first one for making a reference data set. The third part is classification, where one HCV of the current image is compared with the HCV data set. Then detail process of above three parts is checked. From these processes, the system can classify the hand shape of different letters in the ASL alphabet.

The contribution of this chapter is the proposal of a method that can classify the hand postures quickly and accurate in a simple way using only a cheap web camera (i.e., without expensive equipment such as a glove, marker, time-of-flight camera, etc.). The proposed method can recognize input hand posture in real-time at reasonable accuracy. An average of 81.12% was obtained when all 25 classes were used and average 94.45% success rate was obtained when grouping method was used. The proposed method shows 94.08% success rate without grouping, which changes to 98.88% success rate using grouping (in the best case). The results of individual participants are also shown, facilitating observation of the individual differences. It shows very good performance in classification of hand postures. Thus, the proposed system can be used for HCI contexts such as TVs and smartphones control, which uses some designated commands by matching some gestures with commands.

## 6.2 Hand gesture recognition

In Chapter 4, an effective recognizer based on states method is designed and manufactured to confirm if they can recognize the hand gestures. In other word, this dissertation tries to make a recognizing system which can handle the various mouse and camera hand gesture such as numbers using only visual input. The recognizer has been made by FSMs with some probabilities for transition. In order to make a data set, a mouse device and camera are used to obtain the gesture input. In the mouse case, the position of mouse cursor is used to determine gestures, while in the camera case, skin-color detection for finding centroid of hand using RGB color space is used to make gestures in the same way as in Chapter 3.



A training phase is needed to make the recognizer. For training the gesture data, the training phase is divided into two steps. The first step is the spatial clustering to represent the gestures as the set of states using GMM. The second step is the temporal alignments to represent the gestures as the path of state sequences. From these processes, the redefined data sets were obtained, and they are used to train and recognize the gestures.

The recognizer turned out to be fairly accurate to recognize the gestures with various number of states. Although the proposed method can make a fairly accurate recognition rate, experiments were performed with various situations such as same gestures with various time periods and overlap problems in Chapter five.

The recognition performances are influenced according to the spatial and temporal overlap of gestures. The results show the good performance even if the gestures are considerable similar. The results also show that if temporal distributions are different, the overlapping degree of spatial distribution is not important. The proposed method can also cover the significant overlapping level of temporal distribution. The main problem for obtaining good recognition success rates is how to make and decide the proper position and number of states.

Experiments are also performed with the input data that have different time periods according to the speed of the user's gesture. The proposed method does not require many training data sets to recognize the various time period gestures. If the input gestures pass through accurately on the ordered state sequences, then the time period of input data is not significant, as the recognizer is just composed of the state sequence list.

A method was proposed, which is combined the methods of Chapter 3 and 4 to recognize the continuous gestures in real-time. There are two steps to do this work. First, the hand postures obtained from the input gestures have to be distinguished to confirm whether the input is a gesture or not. The hand posture is classified by using the method already introduced in Chapter 3. From this process, it can be confirmed whether the input gesture can be considered as a gesture or not. If the input is determined to be a gesture, then the input is saved into a variable. After obtaining the variable that has the gesture data, the recognizers introduced in Chapter 4 are used to interpret the gesture. This process is almost the same as the method in Chapter 4. It was confirmed that these processes work well, and the proper gesture data can be obtained.

The significance of this chapter is that it established that the proposed methods have low processor overhead and are easy to implement. As current input data is handled immediately at each state using the information and threshold trained already in the training phase, this reduces the computation complexity. The proposed method does not require large data sets. In the proposed method, a different time period of the same input gesture is not relevant to recognizing gestures. This is because, if the data points of the input gesture are positioned properly at each state obtained by training, it recognizes the relevant gesture, even if the hand gesture moves fast.

### 6.3 Future studies

This dissertation proposed the hand posture and gesture recognition systems, and confirmed the performances of these systems with various input data.

The proposed hand posture and gesture recognition systems are used RGB color spaces to extract the hand from the images. For doing this, it have to be used with the clean background, therefore, experiments were performed in front of white panel. However, it has many weaknesses for illumination and cluttered background. To overcome this weakness, it is essential to use more robust color space for detecting the hand. Ultimately, the hand models have to be made, which can be used in the cluttered background.

The 2D position data of the hand obtained from camera is used for the input data of the hand postures and gestures. If Kinect is used instead of a single web camera, the depth information of the hand can be obtained, thus, the more precise information about the features can be made. By using these information instead of 2D data, the robust hand posture and gesture recognition systems can be developed. It can be also the solution to the misclassified problems when the hand rotates in the 3D.

The centroid of hand is relatively extracted well in the proposed method. However, the centroid position of the hand slightly shakes when the hand moves. In order to stabilize the centroid position of the hand extracted by using image processing methods, the method fused with kalman filter, median filter, and so on can be used.

In the proposed method, only one hand is used in the video or images to input and recognize the gestures. However, when the gesture recognition is used in everyday life, the faces and hands of user are necessary to appear in the video or images. Hence,

it has to be developed the systems that can extract the hand and obtain the information even in this situation.

In addition, the research has been needed to find the method how to segment the hand on the cases that there is background that has the similar color with the hand or face, and overlaps the hands. This part is an important side needed to operate the recognition system in the real life. Many algorithms and models have been developed to overcome these problems, however, there are no perfect antidotes to solve these problems. For instance, precise hand contour line have to be obtained in shape based model, but there are also many environmental limited primaries. In motion based model, there are assumptions that movements of hand are the only motions in the image, so it also has environmental limited primary.

Researches are also needed to the tracking methods after detecting the hands. Some approaches show the good performances, but still not robust. For example, template matching method is influenced by the condition of illumination, so it is not always effective. Contour based tracking approaches also have problems to obtain the precise contour line because of the difficulty to extract it.

The performances of the hand gesture recognition systems heavily depend on the image processing techniques such as detection and tracking method which is already explained above. Therefore, it is important to use the optimal detection and tracking methods to fit the application.

The proposed method has the relatively low computation complexity and it is easy to implement. However, there are rigid conditions to input the gestures. For example, the large differences between distribution of training gestures and input gestures that intend the same meaning are not allowed. If it is desired to recognize the same gestures that have the different distributions, then another recognizer has to be made using training with the gestures that have different distributions. However, if the number of the recognizer increases in this manner, then it takes a long time to recognize the input gestures. For operating the gesture recognition system in real time, as previously stated, to find the optimal image processing techniques is important. Therefore, the optimal image processing techniques have to be found or developed, which are suitable to the proposed method in order to operate the systems in real time, and design the robust recognizer.

# Appendix

## A. Contour Distance ( $d_{threshold}$ ) Setting Method

In this section, the contour distance ( $d_{threshold}$ ) setting method for representing the positions that have certain volumes is derived. The equation for the process in Figure 4.12 was derived in Hansen (2005). The equation was introduced in chapter 4. If there are data that are Gaussian distributions with means as  $\mu$  and covariances as  $\Sigma$ , they can be represented and changed as in the following equations.

$$N(\mu, \Sigma) \sim \mu + N(0, \Sigma) \quad (1)$$

$$N(0, \Sigma) \sim \Phi N(0, \Lambda) \quad (2)$$

$$\Phi^T N(0, \Sigma) \sim N(0, \Lambda) \quad (3)$$

$$N(0, \Lambda) \sim \Lambda^{1/2} N(0, I) \quad (4)$$

$$\Lambda^{-1/2} N(0, \Lambda) \sim N(0, I) \quad (5)$$

In this section, it will be confirmed that the volumes of Figure 4.12 (b) and (c) are the same. It is assumed that the mean of data distribution is zero for convenience.

First, the volume of Figure 4.12 (c) is derived. The fixed Mahalanobis distance,  $d$ , for representing the certain volumes, such as 0.95 or 0.99 of (c), is calculated as

$$d = \sqrt{(X - \mu)^T \Sigma^{-1} (X - \mu)} \quad (6)$$

Covariance matrix  $\Sigma$  of (c) is Identity matrix. Therefore, bivariate density function for (c) can be expressed as

$$f(x, y)_c = \frac{1}{2\pi} \exp\left(-\frac{1}{2}(x^2 + y^2)\right) \quad (7)$$

The Area of (c) with certain distance,  $d$ , is the circle, so it follows the equation of circle,  $x^2 + y^2 = d^2$ . Therefore, the volume of (c) in  $d$  can be expressed as

$$\int_{-d}^d \int_{-\sqrt{d^2-x^2}}^{\sqrt{d^2-x^2}} f(x, y)_c dy dx + \pi d^2 f(x, y)_c \quad (8)$$

Second, the volume of Figure 4.12 (b) is derived. In this case, covariance matrix  $\Sigma$  and  $\Sigma^{-1}$  is expressed as

$$\Sigma_b = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \quad (9)$$

$$\Sigma_b^{-1} = \frac{1}{\sigma_x^2 \sigma_y^2} \begin{bmatrix} \sigma_y^2 & 0 \\ 0 & \sigma_x^2 \end{bmatrix} \quad (10)$$

The fixed Mahalanobis distance,  $d$ , is already set, so it can be represented as the following equation using  $\Sigma_b$ .

$$d = \frac{1}{\sqrt{\sigma_x^2 \sigma_y^2}} \sqrt{\sigma_y^2 x^2 + \sigma_x^2 y^2} \quad (11)$$

Using this equation, the elliptical equation that has same Mahalanobis distance can be obtained.

$$\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} = d^2 \quad (12)$$

Bivariate density function for (b) can be expressed as

$$f(x, y)_b = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left(-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right)\right) \quad (13)$$

Using equation 12 and 13, the volumes of (b) in  $d$  can be expressed as

$$\int_{-\sigma_x d}^{\sigma_x d} \int_{-\sqrt{\sigma_y^2 d^2 - \frac{\sigma_y^2}{\sigma_x^2} x^2}}^{\sqrt{\sigma_y^2 d^2 - \frac{\sigma_y^2}{\sigma_x^2} x^2}} f(x, y)_b dy dx + \pi\sigma_x\sigma_y d^2 f(x, y)_b \quad (14)$$

Replace  $x$  with  $\sigma_x u$  and  $y$  with  $\sigma_y v$ . Then,  $dx = \sigma_x du$  and  $dy = \sigma_y dv$ . The ranges of integral are also changed as in the following equations.

$$\begin{aligned} -\sigma_x d &\leq x \leq \sigma_x d \\ -d &\leq u \leq d \end{aligned} \quad (15)$$

$$\begin{aligned} -\sqrt{\sigma_y^2 d^2 - \frac{\sigma_y^2}{\sigma_x^2} x^2} &\leq y \leq \sqrt{\sigma_y^2 d^2 - \frac{\sigma_y^2}{\sigma_x^2} x^2} \\ -\sqrt{\sigma_y^2 d^2 - \frac{\sigma_y^2}{\sigma_x^2} \sigma_x^2 u^2} &\leq \sigma_y v \leq \sqrt{\sigma_y^2 d^2 - \frac{\sigma_y^2}{\sigma_x^2} \sigma_x^2 u^2} \\ -\sqrt{d^2 - u^2} &\leq v \leq \sqrt{d^2 - u^2} \end{aligned} \quad (16)$$

Therefore, equation 14 can be changed to

$$\int_{-d}^d \int_{-\sqrt{d^2-u^2}}^{\sqrt{d^2-u^2}} f(u, v)_c dv du + \pi d^2 f(u, v)_c \quad (17)$$

Equation 17 is same equation 8. It can be confirmed that the volumes of Figure 4.12 (b) and (c) are the same.

# Bibliography

- Ahmad, S. (1994). A usable real-time 3d hand tracker. In *1994 Conference Record of the Twenty-Eighth Asilomar Conference on Signals, Systems and Computers, 1994.*, volume 2, pages 1257–1261. IEEE.
- Alldrin, N., Smith, A., and Turnbull, D. (2003). Clustering with em and k-means. *University of San Diego, California, Tech Report.*
- Bahl, L. and Jelinek, F. (1975). Decoding for channels with insertions, deletions, and substitutions with applications to speech recognition. *IEEE Transactions on Information Theory*, 21(4):404–411.
- Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):179–190.
- Baker, J. (1975). The dragon system—an overview. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 23(1):24–29.
- Baudel, T. and Beaudouin-Lafon, M. (1993). Charade: remote control of objects using free-hand gestures. *Communications of the ACM*, 36(7):28–35.
- Becker, M., Kefalea, E., Maël, E., Von Der Malsburg, C., Pagel, M., Triesch, J., Vorbrüggen, J. C., Würtz, R. P., and Zadel, S. (1999). Gripsee: A gesture-controlled robot for object perception and manipulation. *Autonomous Robots*, 6(2):203–221.
- Black, M. J. and Jepson, A. D. (1998). Recognizing temporal trajectories using the condensation algorithm. In *Proceedings. Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998.*, pages 16–21. IEEE.
- Blake, A., North, B., and Isard, M. (1999). Learning multi-class dynamics. *Advances in Neural Information Processing Systems*, pages 389–395.

- Bobick, A. F. and Wilson, A. D. (1997). A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337.
- Bolt, R. A. (1980). *Put-that-there: Voice and gesture at the graphics interface*. ACM.
- Boulic, R., Rezzonico, S., and Thalmann, D. (1996). Multi-finger manipulation of virtual objects. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST)*, pages 67–74. Citeseer.
- Boulos, M. N. K., Hetherington, L., and Wheeler, S. (2007). Second life: an overview of the potential of 3-d virtual worlds in medical and health education. *Health Information & Libraries Journal*, 24(4):233–245.
- Bourke, A., O'Brien, J., and Lyons, G. (2007). Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm. *Gait & posture*, 26(2):194–199.
- Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'reilly.
- Bradski, G. R. (1998). Real time face and object tracking as a component of a perceptual user interface. In *Proceedings., Fourth IEEE Workshop on Applications of Computer Vision, 1998. WACV'98.*, pages 214–219. IEEE.
- Bretzner, L., Laptev, I., and Lindeberg, T. (2002). Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. In *Proceedings. Fifth IEEE International Conference on Automatic Face and Gesture Recognition, 2002.*, pages 423–428. IEEE.
- Bryson, S. (1996). Virtual reality in scientific visualization. *Communications of the ACM*, 39(5):62–71.
- Bulatov, Y., Jambawalikar, S., Kumar, P., and Sethia, S. (2004). Hand recognition using geometric classifiers. In *Biometric Authentication*, pages 753–759. Springer.
- Calinon, S. and Billard, A. (2005). Recognition and reproduction of gestures using a probabilistic framework combining pca, ica and hmm. In *Proceedings of the 22nd international conference on Machine learning*, pages 105–112. ACM.
- Cipolla, R. and Hollinghurst, N. J. (1996). Human-robot interface by pointing with uncalibrated stereo vision. *Image and Vision Computing*, 14(3):171–178.



- Cipolla, R., Okamoto, Y., and Kuno, Y. (1993). Robust structure from motion using motion parallax. In *Proceedings., Fourth International Conference on Computer Vision, 1993.*, pages 374–382. IEEE.
- Cootes, T. F., Taylor, C. J., Cooper, D. H., and Graham, J. (1995). Active shape models-their training and application. *Computer vision and image understanding*, 61(1):38–59.
- Cui, Y. and Weng, J. J. (1996). Hand sign recognition from intensity image sequences with complex backgrounds. In *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, 1996.*, pages 259–264. IEEE.
- Cutler, R. and Turk, M. (1998). View-based interpretation of real-time optical flow for gesture recognition. In *Proceedings. Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998.*, pages 416–421. IEEE.
- Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005.*, volume 1, pages 886–893. IEEE.
- Darrell, T. and Pentland, A. (1993). Space-time gestures. In *Proceedings. 1993 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1993. CVPR'93.*, pages 335–340. IEEE.
- Delac, K. and Grgic, M. (2004). A survey of biometric recognition methods. In *Proceedings Electronics in Marine, 2004. 46th International Symposium*, pages 184–193. IEEE.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38.
- Derpanis, K. G. (2004). A review of vision-based hand gestures. *Internal Report, Department of Computer Science. York University.*
- Doucet, A., De Freitas, N., Gordon, N., et al. (2001). *Sequential Monte Carlo methods in practice*, volume 1. Springer New York.
- Freeman, W. T. and Weissman, C. (1995). Television control by hand gestures. In *Proc. of Intl. Workshop on Automatic Face and Gesture Recognition*, pages 179–183.
- Goza, S., Ambrose, R. O., Diftler, M. A., and Spain, I. M. (2004). Telepresence control

- of the nasa/darpa robonaut on a mobility platform. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 623–629. ACM.
- Hall, E. T. (1959). *The silent language*, volume 3. Doubleday New York.
- Hansen, N. (2005). The cma evolution strategy: A tutorial. *Vu le*, 29.
- Hernandez-Rebollar, J. L., Lindeman, R. W., and Kyriakopoulos, N. (2002). A multi-class pattern recognition system for practical finger spelling translation. In *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*, page 185. IEEE Computer Society.
- Hong, P. and Huang, T. S. (2000). Learning to extract temporal signal patterns from temporal signal sequence. In *Proceedings. 15th International Conference on Pattern Recognition, 2000.*, volume 2, pages 648–651. IEEE.
- Hong, P., Turk, M., and Huang, T. S. (2000a). Constructing finite state machines for fast gesture recognition. In *Proceedings. 15th International Conference on Pattern Recognition, 2000.*, volume 3, pages 691–694. IEEE.
- Hong, P., Turk, M., and Huang, T. S. (2000b). Gesture modeling and recognition using finite state machines. In *Proceedings. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000.*, pages 410–415. IEEE.
- Hu, C., Meng, M. Q., Liu, P. X., and Wang, X. (2003). Visual gesture recognition for human-machine interface of robot teleoperation. In *Proceedings. 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003.(IROS 2003).*, volume 2, pages 1560–1565. IEEE.
- Hu, J., Brown, M. K., and Turin, W. (1996). Hmm based online handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):1039–1045.
- Hu, J., Gek Lim, S., and Brown, M. K. (2000). Writer independent on-line handwriting recognition using an hmm approach. *Pattern Recognition*, 33(1):133–147.
- Huang, T. S. and Pavlovic, V. I. (1995). Hand gesture modeling, analysis, and synthesis. In *In Proc. of IEEE International Workshop on Automatic Face and Gesture Recognition*. Citeseer.
- Imagawa, K., Lu, S., and Igi, S. (1998). Color-based hands tracking system for sign

- language recognition. In *Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998. Proceedings.*, pages 462–467. IEEE.
- Isard, M. and Blake, A. (1998a). Condensationconditional density propagation for visual tracking. *International journal of computer vision*, 29(1):5–28.
- Isard, M. and Blake, A. (1998b). A smoothing filter for condensation. In *Computer VisionECCV'98*, pages 767–781. Springer.
- Jaimes, A. and Sebe, N. (2007). Multimodal human–computer interaction: A survey. *Computer vision and image understanding*, 108(1):116–134.
- Jain, A. K., Ross, A., and Prabhakar, S. (2004). An introduction to biometric recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):4–20.
- Jelinek, F., Bahl, L., and Mercer, R. (1975). Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21(3):250–256.
- Juang, B.-H. and Rabiner, L. R. (1991). Hidden markov models for speech recognition. *Technometrics*, 33(3):251–272.
- Kadous, M. W. (1995). *Recognition of Australian Sign Language using instrumented gloves*.
- Karam, M. (2006). *PhD Thesis: A framework for research and design of gesture-based human-computer interactions*. PhD thesis, University of Southampton.
- Kendon, A. (1986). Current issues in the study of gesture. *The biological foundations of gestures: Motor and semiotic aspects*, 1:23–47.
- Kim, D. (2006). Memory analysis and significance test for agent behaviours. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 151–158. ACM.
- Kiyokawa, K., Takemura, H., Katayama, Y., Iwasa, H., and Yokoya, N. (1996). Vlego: A simple two-handed modeling environment based on toy blocks. In *Proc. of ACM Simpo. on Virtual Reality Software and Technology (VRST96)*, pages 27–34. Cite-seer.
- Kofman, J., Wu, X., Luu, T. J., and Verma, S. (2005). Teleoperation of a robot manip-

- ulator using a vision-based human-robot interface. *IEEE Transactions on Industrial Electronics*, 52(5):1206–1219.
- Kohavi, Z. (1978). *Switching and finite automata theory*. McGraw-Hill Incorporated.
- Krueger, M. W., Gionfriddo, T., and Hinrichsen, K. (1985). Videoplace an artificial reality. In *ACM SIGCHI Bulletin*, volume 16, pages 35–40. ACM.
- Kurata, T., Okuma, T., Kourogi, M., and Sakaue, K. (2001). The hand mouse: Gmm hand-color classification and mean shift tracking. In *Proceedings. IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001.*, pages 119–124. IEEE.
- Lee, H.-K. and Kim, J.-H. (1999). An hmm-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973.
- Lee, J. and Kunii, T. L. (1995). Model-based analysis of hand posture. *Computer Graphics and Applications, IEEE*, 15(5):77–86.
- Lepetit, V. and Fua, P. (2005). *Monocular model-based 3d tracking of rigid objects: A survey*. Now Publishers Inc.
- Linde, Y., Buzo, A., and Gray, R. (1980). An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95.
- Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.
- Mase, M. and Suenaga, K. (1992). Real-time detection of pointing actions for a glove-free interface. In *In IAPR Workshop on Machine Vision Applications*. Citeseer.
- McGregor, M. J. and Pallai, P. V. (1997). Clustering of large databases of compounds: Using the mdl keys as structural descriptors. *Journal of chemical information and computer sciences*, 37(3):443–448.
- Mitra, S. and Acharya, T. (2007). Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(3):311–324.
- Munib, Q., Habeeb, M., Takruri, B., and Al-Malik, H. A. (2007). American sign

- language (ASL) recognition based on hough transform and neural networks. *Expert Systems with Applications*, 32(1):24–37.
- Murakami, K. and Taguchi, H. (1991). Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 237–242. ACM.
- Nishikawa, A., Hosoi, T., Koara, K., Negoro, D., Hikita, A., Asano, S., Kakutani, H., Miyazaki, F., Sekimoto, M., Yasui, M., et al. (2003). Face mouse: A novel human-machine interface for controlling the position of a laparoscope. *IEEE Transactions on Robotics and Automation*, 19(5):825–841.
- Ottenheimer, H. (2008). *The anthropology of language: an introduction to linguistic anthropology*. Cengage Learning.
- Pavlovic, V. I., Sharma, R., and Huang, T. S. (1997). Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695.
- Phadtare, L. K., Kushalnagar, R. S., and Cahill, N. D. (2012). Detecting hand-palm orientation and hand shapes for sign language gesture recognition using 3d images. In *Image Processing Workshop (WNYIPW), 2012 Western New York*, pages 29–32. IEEE.
- Plamondon, R. and Srihari, S. N. (2000). Online and off-line handwriting recognition: a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):63–84.
- Pugeault, N. and Bowden, R. (2011). Spelling it out: Real-time ASL fingerspelling recognition. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops), 2011.*, pages 1114–1119. IEEE.
- Rabiner, L. and Juang, B.-H. (1993). Fundamentals of speech recognition.
- Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.
- Rautaray, S. S. and Agrawal, A. (2012). Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, pages 1–54.
- Rehg, J. M. and Kanade, T. (1994a). Digiteyes: Vision-based hand tracking for human-

- computer interaction. In *Proceedings of the 1994 IEEE Workshop on Motion of Non-Rigid and Articulated Objects, 1994.*, pages 16–22. IEEE.
- Rehg, J. M. and Kanade, T. (1994b). Visual tracking of high dof articulated structures: an application to human hand tracking. In *Computer Vision ECCV'94*, pages 35–46. Springer.
- Rittscher, J. and Blake, A. (1999). Classification of human body motion. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.*, volume 1, pages 634–639. IEEE.
- Schultz, M., Gill, J., Zubairi, S., Huber, R., and Gordin, F. (2003). Bacterial contamination of computer keyboards in a teaching hospital. *Infection control and hospital epidemiology*, 24(4):302–303.
- Schwarz, G. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464.
- Segen, J. and Kumar, S. (1998). Gesture vr: Vision-based 3d hand interace for spatial interaction. In *Proceedings of the sixth ACM international conference on Multimedia*, pages 455–464. ACM.
- Sharma, R., Huang, T. S., PavloviC, V. I., Zhao, Y., Lo, Z., Chu, S., and Schul, K. (1996). Speech/gesture interface to a visual computing environment for molecular biologists. In *Proceedings of the 13th International Conference on Pattern Recognition, 1996.*, volume 3, pages 964–968. IEEE.
- Shimada, N., Shirai, Y., Kuno, Y., and Miura, J. (1998). Hand gesture estimation and model refinement using monocular camera-ambiguity limitation by inequality constraints. In *Proceedings. Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998.*, pages 268–273. IEEE.
- Song, C. G., Kwak, N. J., and Jeong, D. H. (2000). Developing an efficient technique of selection and manipulation in immersive ve. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 142–146. ACM.
- Song, L. and Takatsuka, M. (2005). Real-time 3d finger pointing for an augmented desk. In *Proceedings of the Sixth Australasian conference on User interface-Volume 40*, pages 99–108. Australian Computer Society, Inc.
- Starner, T. and Pentland, A. (1997). Real-time american sign language recognition

- from video using hidden markov models. In *Motion-Based Recognition*, pages 227–243. Springer.
- Starner, T. E. (1995). Visual recognition of american sign language using hidden markov models. Technical report, Department of Brain and Cognitive Sciences, MIT.
- Stokoe, W. C. (2005). Sign language structure: An outline of the visual communication systems of the american deaf. *Journal of deaf studies and deaf education*, 10(1):3–37.
- Sturman, D. J. and Zeltzer, D. (1994). A survey of glove-based input. *Computer Graphics and Applications, IEEE*, 14(1):30–39.
- Sutherland, I. E. (1964). Sketch pad a man-machine graphical communication system. In *Proceedings of the SHARE design automation workshop*, pages 6–329. ACM.
- Terrillon, J.-C., Shirazi, M. N., Fukamachi, H., and Akamatsu, S. (2000). Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In *Proceedings. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000.*, pages 54–61. IEEE.
- Thieffry, S. (1981). Hand gestures. *The Hand*.
- Van den Bergh, M. and Van Gool, L. (2011). Combining rgb and tof cameras for real-time 3d hand gesture interaction. In *IEEE Workshop on Applications of Computer Vision (WACV), 2011.*, pages 66–72. IEEE.
- Verevka, O. and Buchanan, J. W. (1995). The local k-means algorithm for colour image quantization. In *Graphics Interface*, pages 128–128. CANADIAN INFORMATION PROCESSING SOCIETY.
- Verma, R. and Dev, A. (2009). Vision based hand gesture recognition using finite state machines and fuzzy logic. In *International Conference on Ultra Modern Telecommunications & Workshops, 2009. ICUMT'09.*, pages 1–6. IEEE.
- Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.
- Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically opti-

- num decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.
- Wagstaff, K., Cardie, C., Rogers, S., and Schrödl, S. (2001). Constrained k-means clustering with background knowledge. In *ICML*, volume 1, pages 577–584.
- Wexelblat, A. (1995). An approach to natural gesture in virtual environments. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 2(3):179–200.
- Wu, Y., Lin, J. Y., and Huang, T. S. (2001). Capturing natural hand articulation. In *Proceedings. Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001.*, volume 2, pages 426–432. IEEE.
- Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden markov model. In *Proceedings CVPR'92., 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1992.*, pages 379–385. IEEE.
- Yang, H.-D., Sclaroff, S., and Lee, S.-W. (2009). Sign language spotting with a threshold model based on conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(7):1264–1277.
- Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *Acm Computing Surveys (CSUR)*, 38(4):13.
- Yim, H. Lee, C. and Kim, D. (2013a). A robust static hand gesture recognition system based on vector model (being prepared). *Pattern Recognition*.
- Yim, H. and Kim, D. (2012). Evolving internal memory strategies for the woods problems. In *12th International Conference on Control, Automation and Systems (ICCAS), 2012*, pages 366–369. IEEE.
- Yim, H. and Kim, D. (2013b). Control system based on probabilistic finite state machine for solving t-maze problem (being prepared). *IEEE Transactions on Robotics and Automation*.
- Yim, H. and Kim, D. (2013c). Multi objective particle swarm optimization approach to quantify internal states needed for the woods problem (being prepared). *Connection Science*.
- Yim, H. and Kim, D. (2013d). A state-based approach to the gesture recognition (being prepared). *IEEE Transactions on Pattern Analysis and Machine Intelligence*.



- Yoruk, E., Konukoglu, E., Sankur, B., and Darbon, J. (2006). Shape-based hand recognition. *IEEE Transactions on Image Processing*, 15(7):1803–1815.
- Zabulis, X., Baltzakis, H., and Argyros, A. (2009). Vision-based hand gesture recognition for human-computer interaction. *The Universal Access Handbook*. LEA.
- Zelevnik, R. C., Herndon, K. P., and Hughes, J. F. (2007). Sketch: an interface for sketching 3d scenes. In *ACM SIGGRAPH 2007 courses*, page 19. ACM.