Probability-Based Multipoint Gesture Recognition

Hyeokhyeon Kwon

The Graduate School
School of Electrical and Electronic Engineering
Yonsei University

Probability-Based Multipoint Gesture Recognition

A Master's Thesis
Submitted to the Department of
School 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

Hyeokhyeon Kwon

June 2018

This certifies that the master's thesis
of Hyeokhyeon Kwon is approved.

Thesis Supervisor: DaeEun Kim

Hong-Goo Kang

Euntai Kim

The Graduate School
School of Electrical and Electronic Engineering
Yonsei University

June 2018

Abstract

Due to recent advances in computer science, a variety of technologies have been developed, such as Internet of Things (IoT), which connects the Internet to various objects, and ubiquitous computers, which place computers everywhere. Human-computer interaction (HCI) is being studied for communication with such various computers. One of the most interesting areas of HCI is gesture recognition. Gesture recognition is the core technology of the future science. It can effectively communicate the user's intention to the computer and has various technology application possibility. In this paper, we study gesture recognition technology and develop algorithms with high recognition rate with low computational complexity. First of all, we will study the algorithm that can compensate the shortcomings of FSM. FSM can speed up recognition of simple path, but it has a disadvantage that recognition rate is not good when one gesture has various paths. Chapter 3 of this paper describes a multipath FSM that can overcome these drawbacks. In particular, we experimented with various algorithms of HMM in order to recognize various PATH, and proceeded with research on FSM enemy based on probabilities. As a second method for gesture recognition, we conducted a study on dynamic time warping in chapter4. Dynamic Time Warping is an algorithm that matches two signals in a time series and outputs the similarity. The output similarity finds a pattern that minimizes the two signals in the time series and represents the cumulative value of the distance. Most gestures, however, contain noise components and display different shapes each time they are drawn, so that small spots on both signals accumulate and can be similar, but the recognition rate can be lowered. In order to compensate for this, we made Probability Dynamic Time Warping applying probability distance to DTW. We added the weight using variance to the difference in the time series of the two signals, adding fewer errors in the twisted portion, and adding a large error in the bigger deviation. For a stochastic match, we need to find a representative gesture represented by each gesture. In this paper, we propose a time mean representation, a length mean representation, and a repeated warping representation. Finally,

in chapter 5, we applied and experimented with various methods to solve the problem of starting point and dimension, which is a persistent problem of gesture recognition. UTD-MHAD was used for experiments, and all algorithms showed a higher recognition rate than HMMs proposed by existing data producers. PDTW showed the highest recognition rate among the proposed algorithms, and the probability based FSM could overcome the shortcomings of the existing FSM.

Acknowledgements

새로운 것을 찾고 분석하는데 재미를 느껴 대학원에 지원한것이 한달 전인것 같은데 벌써 졸업을 해야할 시간이 되었습니다. 2016년 7월 처음으로 연구실에 출근하여 많은 사람으로부터 많은 것을 배울수 있는 시간이었습니다. 저에게는 매우 짧게만 느껴졌던 2년이라는 시간동안 항상 옆에 있어주고 도움을 사람들 덕분에 무사히 졸업 논문을 완성할 수 있었습니다. 비록 짧은 말로는 모든 감사를 표현할 수 없지만, 이곳에 짧게라도 남겨 제 책장속에서 항상 되새기고 감사함을 기억할 수 있도록 하고자 합니다.

우선 제가 논문을 쓸 수 있도록 지도해주신 김대은 교수님께 감사인사를 드립니다. 또한 바쁘신 와중에 논문 심사를 맡아주셔서 부족한 점, 고쳐야 할 점 등을 친절하게 알려주신 강홍구 교수님과 김은태 교수님, 그 외에도 2년이란 시간동안 수업을 통해 최신 기술과 과거의 이론을 알려주신 모든 교수님들께 감사 인사 드립니다.

또 절대 빼먹어서는 안되는 바이오사이버네틱스 연구실 선배, 후배들에게 감사를 전하고 싶습니다. 짧다면 짧고 길다면 긴 시간동안 너무나도 많은 동료를 만나 다양한 경험을 하고 많은것을 배울수 있었습니다. 그 누구보다 저에게 많은것을 알려줬고 너무나 많은 것을 배울 수 있었던 창민이형, 저에게 연구실 생활의 자세에 대해 알려준 재현이형, 연구실의 큰형으로 어른스러움을 보여주며 일과 병행하면서 연구를 완성하는 모습을 보여준 원기형, 항상 후배들에게 관심을 가지며 도움을 준 정원이형, 입학때부터 졸업하는 그날까지 함께했고 파리를 지나 스페인 끝에서 끝까지 연구실에서 그 누구보다 나와 많은것을 공유하고 함께 즐거워하고 함께 고생한 슬기형과 병문이, 연구실의 분위기를 밝게 지켜준 만동이, 자기주장 강한 동생때문에 고생한 의현이형, 막내지만 막내아닌 아는거 많은 재우, 공부도 연구도 항상 열심히하는 민철이, 기꺼히 연구실의 큰형이 되어준 종하형, 귀염둥이 진수까지 덕분에 포기하지 않고 끝까지 할 수 있었으며, 석사과정 2년을 누구보다 값진 시간으로 만들수 있었습니다.

마지막으로 대학교 졸업하고 석사를 준비할 때 격려해주며 편이 되어준 형과 계속 공부하고 싶다는 아들의 부탁에 말없이 믿어주고 지켜봐준 부모님께 너무

나도 큰 감사를 드립니다.

이제 저는 졸업하여 연세대학교 전기전자공학과 바이오사이버네틱스 연구실을 떠나지만, 이 2년동안 저에게 너무나 많은 도움을 준 사람들의 은혜를 끝까지 잊지 않겠습니다.

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.

(*Hyeokhyeon Kwon*)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

As computer technology has evolved recently, various additional technologies have been developed. The core technology of future science is the Internet of Things (IOT), which is ubiquitous IOT is a technology in which each object is connected to an Internet server to grasp user patterns and provides a recommendation or a desired service. Ubiquitous indicates providing a necessary service from anywhere in the vicinity of a user. In this technique, judging human behavior and providing service according to the determined information is called Human-Computer Interaction (HCI). One representative HCI is gesture recognition. Human behavior has many meanings. A person bending the waist to pick up objects indicates that necessary things are on the floor, and you can read feelings through facial expressions. Gestures can also be used to help communicate, and can be a means of communication when language is not available. Therefore, a computer recognizing a gesture means that it can grasp the intent of minor actions or changes and provide optimal services. For example, it is possible to develop robots that support human beings and deliver necessary tools during work, recognize a mood from a facial expression and provide a song accordingly. Sign language recognition allows users to recognize and translate words, even if they cannot perform sign language. This recognition of gestures is applicable to various ranges and is one of the efficient methods to control a computer. Therefore, to enhance various advantages of

gesture recognition and its application range, more accurate and quick gesture recognition methods are being studied.

1.1 Gesture Recognition

The difficulty in recognizing gestures is that gestures are not always shown in the same form. Suppose, for example, that two people draw a circle. If the first person draws a big circle and the second person draws a small circle, are these two different gestures? Both gestures are clearly the same gesture, but their input forms are different. Now assume that the same person repeats the same action twice. The first and second actions are similar, but both actions will not be able to follow the exact same path. Thus, it will result in slightly different circles and various errors will be added. To solve this problem, there are methods to divide a gesture into states and recognize it by using the transition between states. Representative methods for this are the Hidden Markov Model (HMM) and the Finite State Machine (FSM). Hidden Markov models are algorithms trained on transitions between states and recognize them roughly using hidden states. Gesture recognition using HMM has been researched for a long time and it has been used in various gesture recognition studies. An FSM memorizes the entire movement path between states and recognizes when the input gesture follows the movement of each state. In this manner, recognition using the state can detect the change by recognizing it as a rough motion by matching it to the representative state of the gesture, rather than matching the entire path. As another recognition method, there is a method that compares the similarity of the input path and trained learned path, such as Dynamic Time Warping (DTW), and recognizes the similarity at one or more levels. Furthermore, because of the development of computer science, various studies on gesture recognition methods using deep learning have been carried out, and high recognition rates have been obtained. In recent years, research on a method of recognizing a gesture by fusing a plurality of sensors instead of gesture recognition

using simpler.

1.2 Motivation

This thesis is inspired by various research studies. A number of studies have been conducted on various methods of gesture recognition, including the Finite State Machine (FSM), Hidden Markov Model (HMM), and Dynamic Time Warping (DTW). Recent research on gesture recognition is aimed at improved awareness through extensive learning. There is a perception using Neural Network as the most popular recognition method these days. As computer science develops, it is possible to analyze signals correctly through simple but repeated trainings. In the past, the computational speed of a computer could not satisfy the algorithm's required speed because of the large computational complexity of network computing. However, recent developments in high-performance computer computations have allowed for faster calculation and analysis of large amounts of data. Various methods were developed and used accordingly. Recently, it develops algorithms with high recognition rate by fusing past various recognition algorithms and neural networks. Therefore, it is necessary to study various past algorithms for fusion with neural networks.

1.3 Objectives

Therefore, this thesis seeks to reduce the shortcomings of the various gesture recognition algorithms. FSM and DTW recognize gestures through mathematical modeling, making them less durable than the trendy CNN. . For FSM, however, the overall state path pattern is observed, and exhibits a decrease in perception when the form appears to be multiple types of gestures. DTW, on the other hand, has a high recognition rate, but applies multiple training models in a simple sequential comparison. A gesture is recognized by its simple Euclidian distance estimates, so it does not represent the form of

the gesture itself. Accordingly, this thesis attempts to compensate for these shortcomings. We have studied the various algorithms developed in the past because they have the possibility of further development and are able to show high recognition rates using simple calculations. The research for this thesis was carried out with the aim of developing an algorithm applicable to a machine which does not have a high-level operating system by allowing a higher recognition rate while maintaining the same amount of information. We want to develop algorithms that can be used in a simpler manner to apply to small robots or products that do not have high computation systems, rather than to systems that can be operated quickly and accurately, such as real-life mobile phones and computers.

1.4 Organization of the dissertation

The composition of this paper is as follows. In Chapter 2, we conducted a background investigation of various features and algorithms used for gesture recognition. Chapter 3 introduces the probability-based FSM that was developed based on standard FSM. Chapter 4 introduces algorithms for stochastic application of DTW and finding the center path of each signal. Chapter 5 explains various experiments conducted to track multi-points based on the skeleton in gesture recognition. Finally, Chapter 6 explores each method, draws conclusions from the results, and discusses future developments.

Chapter 2

Background

Gesture recognition has been constantly evolving over the years, and is still developing. The most important aspect in gesture recognition is which data and which algorithm to use. In this chapter, we will review various data features and algorithms used for gesture recognition.

2.1 Data for Gesture Recognition

First, we will review various data used for gesture recognition. In this section, we describe a multimodal method using a combination of skeleton, vision, and various features to recognize gestures.

2.1.1 Skeleton

The skeleton is the most basic data used for gesture recognition (Zhao et al., 2014). The skeleton provides a means of analyzing joint movements and body movements by representing the human body in the form of a line. A gesture represents the movement of the body, but it requires too much computation to analyze all the corresponding movements of the body. Therefore, we use the skeleton to extract important points to

distinguish the desired gesture. One of the most basic features of real gesture recognition is that many types of algorithms use skeletons (Lee et al., 2015; Tamura et al., 2014). An example of using a skeleton is a dynamic algorithm. In the case of algorithms that represent body movements rather than instantaneous forms, they utilize consecutive body positions in the time series. In addition, Kinect, developed by Microsoft in 2009, has become popular in the gestures recognition using the skeleton (Stone and Skubic, 2015; Ying and LIU, 2016; Liu et al., 2016). Kinect consists of a red-green-blue (RGB) camera and an infrared sensor that can scan the surrounding environment and extract 47 parts of the body. In fact, Kinect can be used to recognize gestures while playing games (Soltani et al., 2012; Gerling et al., 2012). Another device for obtaining skeleton data is Leap Motion (Motion, 2015). Leap Motion is a device that measures and outputs the position of the joints of the hand. It uses an infrared sensor to extract the skeleton of the hand. The field where Leap Motion is mainly used is sign recognition (Potter et al., 2013; Elons et al., 2014; Funasaka et al., 2015). A variety of studies have been carried out to recognize sign language accurately by locating the hands and extracting the hand joints using Leap Motion (Chuan et al., 2014; Mohandes et al., 2014).

2.1.2 RGB Camera

Recognition using RGB cameras is a widely used method for gesture recognition (Kollorz et al., 2008; Wu and Huang, 1999). Vision contains much information, so it can be used in various ways for gesture recognition. As gestures represent visual motion and movement, cameras are suitable for recognition purposes. The histogram of oriented gradients (HOG) is a representative method of camera gesture recognition (Tsai, 2010; Freeman and Roth, 1995). HOG is a method of dividing an image into a certain range to obtain a histogram, and recognizing the gesture based on that form. It can be used to distinguish gestures when the shape of the hand represents the meaning of the gesture,

and to locate the hand when recognizing the gesture by the movement of the hand. Gesture recognition using Haar-like features is also frequently used. Gesture recognition using Haar-like features is also frequently used. The Haar-like feature is a method of putting black and white masks on images for gesture recognition and analyzing the meanings according to the mask scores (Chen et al., 2007; Barczak and Dadgostar, 2005). Data using RGB camera can be used as it is, but it can also be used with depth cameras to extract the skeleton (Schwarz et al., 2012; Zhang and Tian, 2012).

2.1.3 Multimodal analysis

Recently, there have been gesture recognition methods that combine multiple pieces of information rather than a simple gesture recognition using one piece of information (Escalera et al., 2017; Neverova et al., 2016). With the development of computer technology, the amount of computation has increased significantly, and it is possible to process information quickly, so multimodal gesture recognition is applied by supplementing information that cannot be obtained by one data source with other data. The most basic multimodal gesture recognition method is to extract the exact skeleton by fusing depth and RGB camera data. In addition, several Kinects may be used simultaneously to complement each other to create overlapped dat of the body. There are five ways to evaluate each gesture when using multiple gestures at the same time.

1. Select the most probable predominant gesture by multiplying the likelihood value of the gesture model obtained from each piece of data (Chang, 2014).
2. Multiply the likelihood value of the gesture model obtained by each data by assigning weights in order of significance to each gesture (Pitsikalis et al., 2017).
3. Select n models that have good classification properties for each gesture (Monnier et al., 2014).
4. Apply special roles to each data for recognition (Ohn-Bar and Trivedi, 2014; Peng

et al., 2014).

The first method assumes that each model has meaning. Assuming that all models are meaningful, multiply all likelihood values obtained by assigning them the same reliability. The second method assumes that the recognition rate of each model is different. When the recognition rate differs from one model to another, there is a difference in reliability, and gesture recognition is more dependent on data with high reliability. In the third method, only data with a certain level of reliability are used. Because the processing of meaningless data is not performed, the operation is faster, and because all data are meaningful data, the recognition rate increases. In the final method, a specific role may be assigned to each data. For example, a two-dimensional shape can be detected with an RGB camera, but three-dimensional information cannot be obtained. Therefore, a depth camera is also used to obtain three-dimensional information. The information used to recognize the gesture is varied and richer, and a better recognition rate by fusing the information.

2.1.4 UTD-MHAD

As multiple gesture recognition has been studied recently, many researchers are making a Dataset that acquired the gesture as a different sensor. The University of Texas at Dallas Multimodal Human Action Dataset (UTD-MHAD) (Chen et al., 2015) contains a variety of actions taking a daily actions. The dataset includes data obtained from Microsoft Kinect camera and electronic sensor. The composition of the gestures is shown in table 2.1.

The detailed behavior of each gesture was shown in Appendix A. Gestures were performed four times by eight subjects (four men and four women) and consist of 27 classes. The dataset composed of Hand Gesture (Swipe left, Swipe right, Wave, Clap and so on), Sports motion (Bowling, Boxing, Baseball swing and so on), daily activitiy

gesture index	gesture name	gesture index	gesture name
1	Swipe left	15	Tennis swing
2	Swipe right	16	Arm curl
3	Wave	17	Tennis serve
4	Clap	18	Push
5	Throw	19	Knock
6	Arm cross	20	Catch
7	Basketball shoot	21	Pickup and throw
8	Draw X	22	Jog
9	Draw circle (clockwise)	23	Walk
10	Draw circle (counter clockwise)	24	Sit to stand
11	Draw triangle	25	Stand to sit
12	Bowling	26	Lunge
13	Boxing	27	Squat
14	Baseball swing		

Table 2.1: Index and names of gestures.

(Knock on door, sit to stand, stand to sit, and so on), and training exercises (arm curl, lunge, and squat and so on) Skeleton data include 20 points obtained from the Camera using the Desert Motion Map.

Inertial data include the three axis gyro sensor, the three axis acceleration sensor, and the nine axis data obtained from the three axis gerometer sensor.

2.2 Classification Algorithm

2.2.1 Hidden Markov Model

In the field of pattern recognition, a conventional and popular method is the Hidden Markov Model (HMM) (Lissier, 1995; Rabiner, 1989; Lee and Hon, 1989; Rabiner

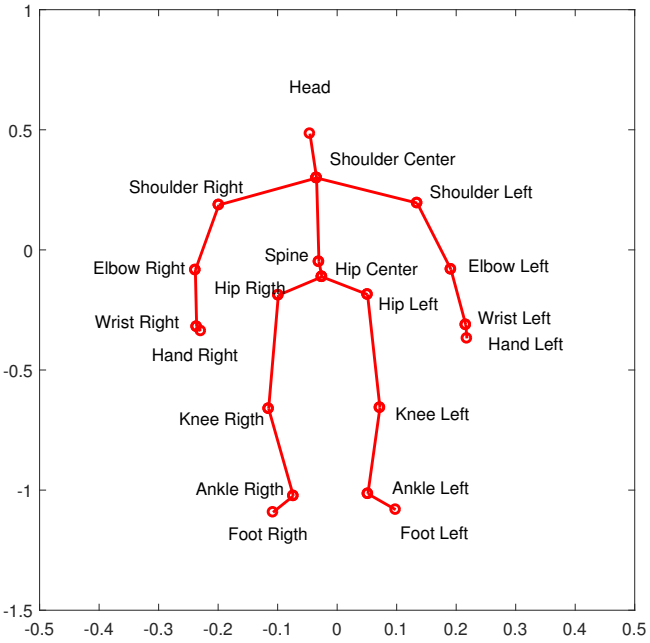


Figure 2.1: The skeleton of UTD-MHAD. The skeleton include totally 20 points.

and Juang, 1986). The HMM is a method of extending the correlation of successive sequences in the Markov chain (Gilks et al., 1995). This algorithm is used when the data pattern has stochastic tendency but cannot be calculated accurately. HMM uses hidden states to be robust against errors. Hidden states are used to prevent inputs from adding errors that cause undesired state movements. For example, if there is a noisy burst in a signal when there is no hidden state, there may be a state shift that would later be learned by the model, and the recognition attempt could fail. However, with a hidden state, we can re-estimate the state by comparing it with the current input and the previous state. Thus, the HMM extends the Markov chain by using a forward algorithm (Federgruen and Tzur, 1991), backward algorithm (Yu and Kobayashi, 2003), Viterbi algorithm (Forney, 1973), and the Baum-Welch algorithm.

The variables used in the HMM are shown below.

O : Observation. This is the set of observation of each time t. The element$o_t$is obser-

vation at time t.

s_i : i-th state. State is calculated by probablity and choosed best path.

q_t : The state of time t.

v_k : The k-th observation. $o_t = v_k$ means that the observation of time t is v_k

A : State transition matrix. It is N x N matrix and N mean the number of state. The element a_{ij} mean the probability of transition state S_i to S_j.

B : It is N x N matrix. The element $b_i(v_k)$ means the probability that we show v_k on i-th state.

π : The initial probability. π_i means the probability that initial state is S_i.

λ : HMM probability parameters. This consist of A, B, π.

The first step of HMM is finding $P(O|\lambda)$ This is the process of determining the probability of observations occurring in the trained model. Forward algorithm and backward algorithm are used to obtain this. In the HMM, there is no probability of transition of Observation in the trained model. However, the probability of the input sequence is obtained through the relationship between the hidden state and observation. The Forward Algorithm calculates the probability of observations occurring when observations are input from the time 1 to time T as the product of the probability parameters. If the o_1 is observation of time 1, the probability that model is start from hidden state i and o_1 is occur from state i. This is shown on eqation 2.1

$$\alpha_i(1) = P(o_1, q_1 = s_i|\lambda) = \pi_i * b_i(o_1) \tag{2.1}$$

We then obtain the probability of occurrence of o_2 in hidden state i at time 1 and state j at time 2. This can be expressed as follows.

$$\alpha_i(2) = P(o_1, o_2, q_2 = s_j|\lambda) = \sum_{i=1}^{N} \alpha_i(1) * a_{ij} * b_j(o_2) \tag{2.2}$$

In the same way we can calculate the probability of state t at time t and Observation is $\{o_1, o_2, ..., o_t\}$ by accumulate probability. After that we can calculate $P(O|\lambda)$ by summation the $\alpha_i(T)$ with every i. This process is shown on equation 2.3.

$$\begin{gathered} \alpha_i(t) = P(o_1, o_2, ..., o_t, q_t = s_i|\lambda) = \sum_{j=1}^{N} \alpha_j(t-1) * a_{ji} * b_i(o_t) \\ \alpha_i(T) = P(O, q_T = s_i|\lambda) \\ P(O|\lambda) = \sum_{i=1}^{N} \alpha_i(T) \end{gathered} \tag{2.3}$$

As can be seen from the equation, the forward algorithm computes the $P(O|\lambda)$ from time 1 to T, accumulating in time order. On the other hand, the backward algorithm does not perform the same computation sequentially from time 1 to T, but computes in reverse order of time from time T to 1. The equation for this is shown in equation 2.4.

$$\begin{gathered} \beta_i(T) = P(q_T = s_i|\lambda) = 1 \\ \beta_i(T-1) = P(o_T, q_{t-1} = s_i|\lambda) = \sum_{j=1}^{N} \beta_j(T) * a_{ij} * b_j(o_T) \\ \beta_i(t) = P(o_{t+1}, ..., o_T, q_{t+1} = s_i|\lambda) = \sum_{j=1}^{N} \beta_j(t+1) * a_{ij} * b_j(o_{t+1}) \\ P(O|\lambda) = \sum_{i=1}^{N} \beta_i(1) * b_i(i_1) \end{gathered} \tag{2.4}$$

To obtain the probability relation between newly input path and model using pre-trained λ, HMM use viterbi algorithm which is one of dynamic programming methods. The Viterbi algorithm is an algorithm that looks at the entered observation column when a new gesture is entered and finds an optimal path of the hidden state that the observation column can represent in the model. Forward algorithm showed the cumulative probability of moving in each state in order to obtain the probability of occurrence

of observations, not the path of hidden state when the entire observation column is input. The viterbi algorithm selects the state with the smallest accumulated error, not the sum of the probabilities at the previous time, to find the optimal of each path. The probability of each state at time 1 is obtained as follows.

$$P(o_1, q_1 = s_1|\lambda) = \delta_i(1) = \pi_i * b_i(o_1) \tag{2.5}$$

Considering the probability at time 1, the probability of state i at time 2 is given by the probability that a transition occurs from state j to i at time 1 and state j, and observation o_2 at state i The probability is the product of the probability of state j at time 1. Therefore, the formula for obtaining the maximum probability is a value that maximizes the product of the probability of transition from state j to i, the probability of observation o_2 in state i, and the probability of state j in time 1 . This process is shown in equation 2.6.

$$\begin{aligned} P(o_1, o_2, q_1 = s_j, q_2 = s_i|\lambda) &= \delta_j(1) * a_{ji} * b_i(o_2) \\ \delta_i(2) &= max(P(o_1, o_2, q_2 = s_i)|\lambda) \end{aligned} \tag{2.6}$$

If we generalize it to the equation at time t, it is equal to equation 2.7.

$$P(o_1, o_2, ..., o_t, q_t = s_i|\lambda) = \delta_i(t) = max(P(o_1, o_2, q_2 = s_i)) \tag{2.7}$$

If this process is repeated up to time T, the values stored in time T are stored at the highest probability value of path ending in each hidden state at time T. Thus, at time T, the maximum value of $\delta_i(T)$ is the probability value of the optimal hidden state sequence. This is summarized by equation 2.8.

$$P(Q, O|\lambda) = max(\delta_i(T)) \tag{2.8}$$

Finally, we need training on probability model to classify each gesture. HMM use the Baum-Welch algorithm to train probability models. The Baum-Welch algorithm is a kind of EM-algorithm. In order to train the probability model, we calculate the probability of the path and re-estimate the model's stochastic parameters based on the path. There are initial probability(π_i), state transition probability(a_{ij}) and obervation symbol probability distribution($b_j(k)$) in HMM. To re-estimate the probability parameters, we first define the probability that a transition occurs state i to state j at time t to time t + 1 with given the observation. When the obervation sequence given, the probability that state at time t is state j is $\alpha_j(t)$ which is defined before. Also, the probability that state j at time t+1 is $\beta_j(t+1)$. Thus, given the observations, the probability of trainsition at state i to state j at time t to time t + 1 is obtained by multiply the $\alpha_i(t)$, $\beta_{j(t+1)}$, a_{ij} and o_{t+1}. The equation for this is shown in equation 2.9.

$$\begin{aligned}\zeta_t(i,j) &= P(q_t = s_i, q_{t+1} = s_j, O|\lambda) \\ &= \frac{P(q_t = s_i, q_{t+1} = s_j|O,\lambda)}{P(O|\lambda)} \\ &= \frac{\alpha_i(t) * a_{ij} * b_j(o_{t+1}) * \beta_i(t+1)}{P(O|\lambda)} \\ &= \frac{\alpha_i(t) * a_{ij} * b_j(o_{t+1}) * \beta_i(t+1)}{\sum_i^N \sum_i^N \{\alpha_i(t) * a_{ij} * \beta_i(t+1) * b_j(o_{t+1})\}}\end{aligned} \tag{2.9}$$

In this equation, $\zeta_t(i,j)$ is the probability of state i at time t and state j at time t + 1 with given observations, so we can calculate the probability of state i at time t when given an observation through integration with j. This equation is shown on 2.10.

$$\gamma_t(i) = P(O, q_t = s_i|\lambda) = \sum_{j=1}^{N} \zeta_t(i,j) \tag{2.10}$$

Using $\zeta_t(i,j)$ and $\gamma_t(i)$ which is obtained on equation 2.9 and equation 2.10, HMM re-estimate a_{ij} and $b_j(k)$. a_{ij} means probability that the state transition is occure state i to state j, so by dividing summation of $\gamma_t(i)$ which means the state i at time t when

obervation is given with summation of $\zeta_t(i,j)$ which means the state transition occure state i to state j at time t We use the obtained value, $\hat{a}_{ij}$, as re-estimated value of a_{ij}. This process is shown in equation 2.11.

$$\hat{a}_{ij} = \sum_{t=1}^{T} \frac{\zeta_t(i,j)}{\gamma_t(i)} \tag{2.11}$$

Also, with a given observation, we divide the probability of state i at time t by the probability of state i at time t and the probability that $o_t = v_k$ to obtain $b_j\hat{(}k)$ and update $b_j(k)$. This is equivalent to equation 2.12.

$$\hat{b}_j(k) = \sum_{t=1}^{T} \frac{\gamma_t(i)|_{o_t=v_k}}{\gamma_t(i)} \tag{2.12}$$

In the HMM, when the data is input, the state is estimated based on the input data, and the probability model is trained by repeating the process of re-estimating the probability parameter based on the state.

Hidden Markov models are used in various fields such as gesture recognition (Schlömer et al., 2008) and speech recognition (Rabiner, 1989), where the vector Taylor series is applied to HMM to recognize speech(Acero et al., 2000; Rose and Paul, 1990; Yamato et al., 1992; Chen et al., 2003). A threshold HMM (Lee and Kim, 1999) uses a threshold model to calculate the likelihood threshold of the input pattern and confirms the matching gesture pattern to process the un-staged pattern using the hidden Markov model based technique (Lee and Kim, 1999). In the HMM method, the threshold method is used to recognize the most similar gesture(s). To prevent recognition of a gesture having a higher probability value than other models (even though the input gesture is completely different data), use a method that is not recognized as a gesture. The authors of this paper used this method to maintain a high recognition rate and to develop a recognizer that can recognize mistakes even if incorrect gestures are input. Another method is to parametrize the existing output probability with the parametric

Hidden Markov Model (Wilson and Bobick, 1999). This method can be used to add a three-dimensional angle to a moving gesture in two dimensions to provide smooth motion using parameters (Gales, 1998).

2.2.2 Finite State Machine

A finite state machine (FSM) which is mainly used for gesture recognition or signal recognition classification is a method of inputting various training data to obtain feature points, dividing states, and confirming inputs based on state movement. Most of the signals used for pattern recognition have a similar shape, but they have a shape that gradually changes. Therefore, FSM algorithm divide the state based on a certain range and divide the form into several clusters and use the characteristic that the same gestures pass through the same state in the same order. A FSM is a method frequently used in the field of logic circuits. When there is a combination of certain states, the state is changed according to a specific input, and if all features of the state are moved equally, FSM can recognize the gesture. Because it only computes the motion of the state, this method has fast operation speed. Because it only computes the motion of the state, this method is fast.

However, it does not have a process for error handling such as HMM's, so it is vulnerable to errors. . A FSM used for gesture recognition uses vector quantization applied to input gestures to distinguish states and use them for recognition. . Especially, the model of the finite state machine used for gesture recognition is a Mealy model. When the sequence for the current position of the gesture is input, the state is moved according to the input and the recognition is made when the gesture follows every state. The millimetric model is mathematically defined by six parameters $(S, s_0, \Sigma, \Lambda, T, G)$. A description of each parameter is given below.

S : State Group

s_0 : initial state

Σ : Set $\{o_1, o_2, ...o_{T-1}, o_T\}$,$o_i$ means input at time i

Λ : output

T : Function that output the next state according to the current state and inputs.

G : Output function generating the output according to current state and inputs.

When a state sequence of a gesture is input, it is assumed that the state moves according to the input and is recognized when passing through all states. Most of the gestures follow similar paths, so when representing the same gesture, a sequence of the same state is produced. Therefore, it is applied as a method for gesture recognition (Hong et al., 2000a; Davis and Shah, 1994; Hong et al., 2000b). Another type of gesture recognition method using a finite state machine is a threshold finite state machine (TFSM) (Bhuyan et al., 2005). In the TFSM, researchers emphasize the need to remain in a state for more than a certain period of time to maintain the shape of the gesture. Therefore, they used a method of matching a simple sequence where a gesture remains in each state for a minimum threshold of time. Another FSM gesture recognition algorithm, Yim (2013) used forward algorithm to make FSM rubust to errors.

2.2.3 Dynamic Time Warping

Dynamic Time Warping (DTW) is a method that involves two time-varying input signals and compares the differences between the two signals (Berndt and Clifford, 1994; Keogh, 2002; Keogh and Pazzani, 2001). DTW finds the points of greatest similarity between two signals through dynamic programming to find matching points. In gesture recognition, when there are various patterns of signals, the pattern recognized is the pattern that has the smallest difference or if the difference is less than the predetermined distance value (Salvador and Chan, 2007). Algorithm 1 presents the pseudo code for DTW.

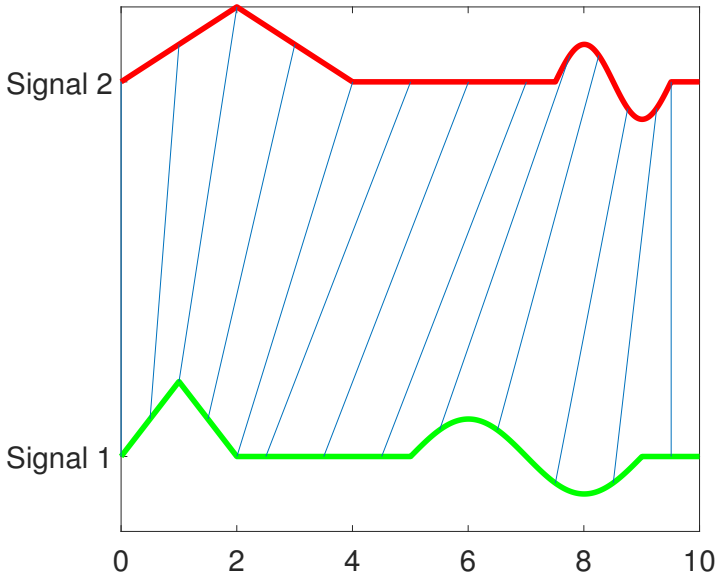


Figure 2.2: Two signal and matching point of DTW

Algorithm 1 Optimal Warping Path

Input: : Input signal SIG1 and SIG2 which length is N, M.

Output: : Optimal warping path similarity d^k

for i *in* 1..N **do**
  for $jin1..M$ **do**
    $W(i,j) = |s1_i - s2_j|$
  end for
end for

for $iin1..N$ **do**
  for $jin1..M$ **do**
    $D(i,j)$ = D(i,1), D(i,2) $\sim$ D(i,j-1), D(i,j) if i = 1
    $D(i,j)$ = Sum of D(1,j), D(2,j) $\sim$ D(i-1,j), D(i,j) if j = 1
    $D(i,j)$ = D(i,j) + argmin{D(i-1,j),D(i-1,j-1),D(i,j-1)}
  end for
end for

For example, suppose that the two signals in 2.2 are input. Signal 1 and Signal 2 are similar in overall shape, but vary in time. Each sequence of the two signals is matched and their similarity is compared. . If recognition is performed in this manner, a path can be more accurately recognized. The reason is that, unlike the methods described previously, differences are obtained not by extracting feature points but because the entire path is considered. When comparing the state and signal, even if a large difference occurs outside the state, the distance from the expected state is too far away to analyze it. Dynamic time warping, on the other hand, has the advantage that no blind spots are created by analyzing distances from all parts without using specific information. On the other hand, it has a disadvantage in that it consumes significant calculation time because it evaluates a very large amount of data. When used for gesture recognition, one reference signal is created for each gesture and a gesture with the smallest distance value is obtained. To overcome the disadvantages of DTW, lower bounding is being studied. One lower bounding method is to use the observation that all points greater than the maximum of other sequences in one sequence should contribute at least the squared difference of the maximum value of the other sequences to the final DTW distance (Yi et al., 1998). Another method is to use a window on the signal, store the minimum value inside a certain interval, and declare U and L to store the maximum value. The equation of U and L is as follows. In equation 2.13 q_i is signal at time i and r is window size.

$$\begin{aligned} U_i &= max(q_{i-r} : q_{i+r}) \\ L_i &= min(q_{i-r} : q_{i+r}) \\ P(q_{t+1} = S_i|\lambda) &= argmax(P(q_{t-1} = S_i)(i > 1) \end{aligned} \tag{2.13}$$

Then reduce the interval beyond U_i and L_i.

2.2.4 Neural Network

Neural networks (NNs) are one of the recognition methods that imitate the brain (Demuth et al., 2014). A neural network consists of several hidden layers, with nodes (neurons) inside each layer, weighted lines connecting the nodes between layers, and an input and an output layer. One of the reasons why neural network algorithms are popular is their ability to learn different types of relationships among data (LeCun et al., 1990; Tollenaere, 1990). To adjust weight values, weight values are not directly modified by the user, but are modified based on error feedback calculated at the output units after a signal has propagated through the network. For example, if a signal1 is input and the expected output to signal1 is 001, the responses calculated at the output layer provide feedback to each weight by calculating the error between the current response and the expected response of 001. When this process is repeated, weights should eventually converge to constant values so that 001 is output when signal1 is input. If the weights converge correctly, then signal1 can be recognized. If the weight is converged, then signal1 is input and output becomes 001 and can be recognized. The feedback of the output to the input requires extensive training data and computation time. The development of current hardware can compensate for these shortcomings, and recently NNs have regained popularity owing to their high recognition rates and selection capabilities. In recent years, various methods for improving the recognition rate have been studied, and solutions for problems such as local minima, calculation speed, and overfitting have been proposed. The advantage of NNs is that it is possible to construct a recognizer with high recognition rate through repeated training and to construct different types of recognizers by adjusting the learning method and/or the composition of layers and nodes. As mentioned previously, NNs do require extensive computation time for training and require a large amount of training data; it is difficult to use NNs when training data is insufficient.

2.3 Application

Recently, gesture recognition using various methods has been carried out for various applications. An older example using HMMs is a gesture-recognition paper by Rung-Huei Liang and Ming Ouhyoung (Liang and Ouhyoung, 1998). They used HMM to recognize sign language (Liang and Ouhyoung, 1995; Starner, 1995). For this purpose, they analyzed statistics according to the four parameters of a gesture (posture, orientation, and motion) and implemented a prototype with 250 vocabulary in Taiwanese Sign Language. The system used a Hidden Markov model for 51 basic positions, 6 directions and 8 movements (Charniak, 1996; Vaananen and Bohm, 1993). More recently, Palma et al. (2016) combined HMM and DTW as a method to evaluate the accuracy of motion during rehabilitation treatment. An example using a combination of an NN and an HMM was recognizing a gesture and recognizing an action taken by a dancer to determine the gesture's action path (McCormick et al., 2014). A re-estimate of a gesture drawn on a touchpad through long short-term memory (LSTM), a type of neural network, was evaluated by Alsharif et al. (2015). In addition, gesture recognition has been applied in sign language recognition (as mentioned previously), robot control (Lim et al., 2017; Chen et al., 2017), and in various other fields.

2.4 Summary of Chapter 2

In this chapter, we reviewed the data and algorithms used for gesture recognition. As gestures are based on body movements, there are many methods involving vision. Recently, methods of simultaneously processing multiple data instead of single data owing to increases in hardware operation speed have been studied. Algorithms for gesture recognition have been developed and evaluated from the 1980s to the present and have been applied in various ways. The use of gesture recognition is applied to modern science such as in robot control, computer control, sign language recognition, and motion

recognition, and there will be further developments in the future. Therefore, in this paper, we have studied an algorithm to enhance existing gesture recognition methods.

Chapter 3

Probability-based Finite State Machine

In this chapter, we propose a multipath FSM to compensate for the fact that there is only one path, which is a disadvantage of FSM, and that it is vulnerable to errors. For recognize more then one path, We use state transition matrix which train transition probabilty between state and assumed that there is a path between state when state trainsition probability is nonzero. Also to choose one path when more than two paths are recognized, we compare the probability that multiply every state transition and mahalanobis distance between state center and each sample. Remember the probabilities for the starting and ending points as a way to maintain the overall shape of the gesture. This section is has published as a paper(Kwon and Kim, d) and additinal experiment being prepared for submission as a paper in journal 'Sensors and Actuator'(Kwon and Kim, a).

3.1 State and State Sequence

An FSM is a state transition based recognizer, therefore before using it for gesture recognition, a state sequence should be generated by classifying the states. In order to classify the states in the space, a vector quantization is performed, which divides the whole path into k groups using the K-means algorithm. The K-means algorithm is

one of the vector quantization methods that classifies samples with N dimensions into k clusters. For clustering, k random samples are selected and set to the initial center position. The entire sample is then included in the closest cluster of k centers. The centers of mass of the obtained k clusters are then found, and set again as the center values. This method is repeated until the center of each cluster converges. When the center position of each state is determined, the center of each gesture is found at each time, and a sequence of states is generated to obtain the gesture path. The formula for creating the state sequence at this time is as follows.

$$v_t{}^b = arg_i min(d_i)\ (d_i\ is\ distance\ between\ x_t\ and\ center\ of\ state\ i) \tag{3.1}$$

Let us assume that the generated path is a basic path. At this time, the Mahalanobis distance is used to select the closest state. The Mahalanobis distance is a probability-based distance that is affected by the differences in simple coordinates, and by covariances. The gesture does not match all the paths, but most of them have similar shapes, and therefore, they do not fall more than a certain distance from the average. Using this feature, we can apply covariance to the Mahalanobis distance estimation. When a covariance is applied, it is possible to apply not only the distance from the center of the state, but also the dense section of the state configuration. The Mahalanobis distance equation is equivalent to equation 3.2.

$$\begin{aligned} \mu &= E(X) \\ \Sigma_i &= E((X-\mu_i)(X-\mu_i)^T) \\ Distance(x_t,\mu_i) &= \sqrt{(x_t-\mu_i)^T {\Sigma_i}^{-1}(x_t-\mu_i)} \end{aligned} \tag{3.2}$$

3.1.0.1 Forward Path

In order to obtain the simplest State Sequence, the Mahalanobis distance is used to obtain the closest state of the hand at each time, and it is expressed as a sequence.

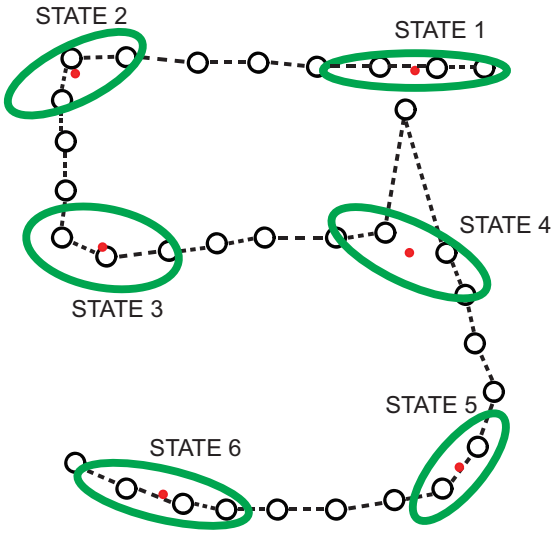


Figure 3.1: A gesture with an impulse error added. An added error in the gesture can cause unwanted state movement.

However, when the actual gesture is used, an error may be added to the actual user's desired path due to either the situation at the time, a user's mistake, or a recognition error. The basic state sequence adds the sequence of the unwanted state, by including the gesture in the nearest state. For example, assuming that a gesture of figure 3.1 has been input, the actual gesture is quite similar to 5; however, an impulse error is added, resulting in a sequence of undesired states. When such an error occurs, we make up a basic path that can be fatal to create a forward path that can locate a path problematically by looking at the previously input sequence. This is similar to the forward algorithm used in an HMM. We can modify the position of the current state by comparing it with the previous state through probability. Stochastic modification is possible using matched training parameters. The HMM forward algorithm is applied by using the fact that the path can be modified through the hidden state. Using the HMM forward algorithm, we can obtain the most probable state based on previously input data. This process is shown in equation 3.3.

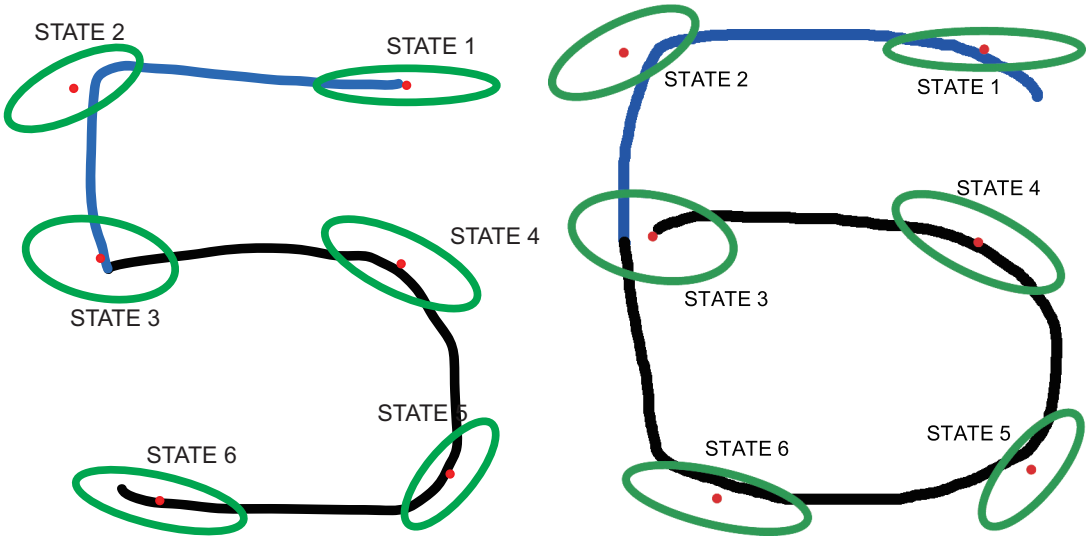


Figure 3.2: (a) Gesture ”5” and its state. (b) Gesture ”6” and its state. The path in the blue part is almost the same, but the path in the black part is very different.

$$\begin{aligned} \alpha_1(i) &= \pi_i * b_i(1) \\ \alpha_{t+1}(j) &= \sum_{i=1}^{N} \alpha_t(i) * a_{ij} * b_j(k) \\ v_t{}^f &= arg_i max(\alpha_t) \end{aligned} \tag{3.3}$$

The probability that π_i in equation 3.3 starts at state i, $b_i(k)$ is the probability of observation k occurring in hidden state i, and a_{ij} is the state transition probability. Thus, $\alpha_t(j)$ is the probability of state i at time t, assuming that time t is affected by time t-1.

3.1.0.2 Forward + Backward Path

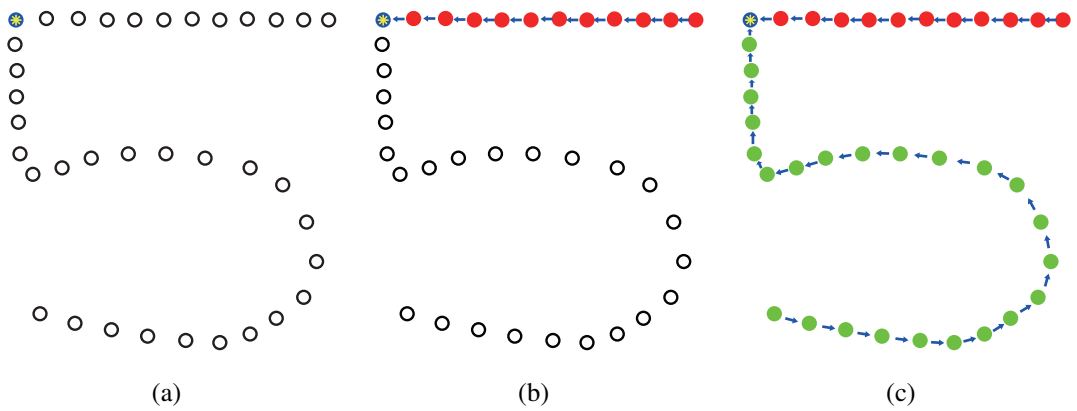
The forward path determines the state at time t using the state of the sequence at time t - 1, and the position of the gesture at time t when the gesture is input. However, the gesture is meaningful when one input is completed, and it is meaningful not only for the state at time t - 1, but also for the state at time t + 1. Take the figure 3.2 as an example. Figure 3.2 (a) gesture ”5”, figure 3.2 (b) is the gesture of gesture ”6”. Gesture ”5” and gesture ”6” draw a roughly matched path in the red part, but draw a

totally different path in the blue part. As can be seen from this, the gesture is important not only from the information of the previously inputted sequence, but also from the information of the sequence that was inputted later. Therefore, we used the forward and backward algorithm for both the previous state and the next state. In order to estimate the position of the state, the probability of state i at the end of the entire time sequence will be $b_i(T)$. The probability of state i (at time t) is then multiplied by the probability of a state transition from state 1 to 6 (at time t + 1), the state transition from state 1 to 6 (at state i), and b_i probability. Then, the path (using the information of the entire path) can be generated by using the information of time 1 t of the forward path, and the probability of using the information of time t to T. This is equivalent to the expression 3.4.

$$\begin{aligned} \beta_T(i) &= 1, t = T \\ \beta_t(i) &= \sum_{j=1}^{N} \beta_{t+1}(j) * a_{ij} * b_j(o_{t+1}) \\ v_t &= arg_i max \alpha_t(i) * \beta_t(i) \end{aligned} \tag{3.4}$$

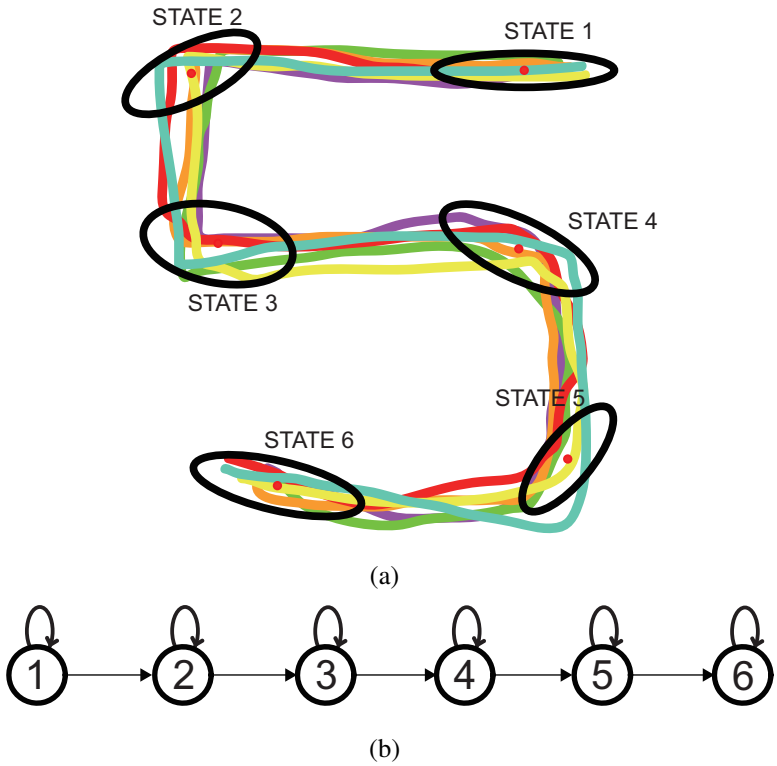
In this equation, $b_i(k)$ is the probability that the observation is k when hidden state i, pi_i is the prior probability and a_{ij} is the state transition probability. Thus, $\beta_t(j)$ is the probability of state i at time t, assuming that time t is affected by time t + 1. Assuming that time t is affected by time t - 1, and affects time t + 1, $\alpha_t(i) * \beta$ is the probability of state i at time t. We summarize the information used for each path state in figure 3.3.

Figure 3.3 shows that the forward path uses more information than the basic path, and the forward + backward path uses more information than the forward path. Each path applies to all FSMs.



(a) (b) (c)

Figure 3.3: (a) Information used by the basic path. (b) Information used by the forward path. (c) Information used by the forward + backward path. The basic path uses only the position information at the time, but the forward path and the forward + backward path obtain a probabilistic position through the forward algorithm and the backward algorithm.



(a)

(b)

Figure 3.4: (a) Gesture ”5” and trained state (b) An FSM generated from gesture ”5”. Gesture ”5” state change order is represented by the FSM. If the input state sequence reaches the end without going out of the FSM, it is judged as a gesture.

3.1.1 MultipathFSM

The FSM recognizes gestures with state sequences created in the previous section. To make an FSM used for recognition, we train the sequence of the state to make the state machine the most frequent sequence of the state. Suppose, for example, that gesture ”5” is entered in the trained state as figure 3.4 (a). The input gesture ”5” represents the state sequence of most sequences 11...1122...2233...3344...4455...5566...66. At this time, the state sequence of the most input gesture is made into an FSM as figure 3.4 (b). If the input gesture does not follow the FSM, it means that the input gesture, and the gesture indicated by the FSM, are different gestures. Conversely, when passing through all FSMs, the input gesture is recognized as a gesture equivalent to the state indicated by the state. By recognizing the gesture as a single FSM, unlike the HMM it is possible to reduce the influence of the time that the gesture stays in the same state, and to distinguish the gesture by concentrating only on the path where the gesture travels. In the HMM, the probability of staying in one state changes according to both the moving speed of the gesture and the sampling frequency. Therefore, the transition matrix is changed, and the recognition may fail when the gesture is input at a different speed to the trained gesture. However, when an FSM is used, the information about the state movement is less influenced by the time, and the effect of the speed of the gesture is reduced. An FSM used for recognition is generated for each gesture. If the input gesture is recognized through two or more finite state machines, a gesture with greater similarity is determined by calculating the distance. This process will be described later.

3.1.1.1 Multiple FSM

In this section, we describe how to extend an FSM so that it recognizes multiple paths in a single gesture. Because every gesture does not always move to the same position, an FSM recognizes the gesture based on the state. However, the movement of the state

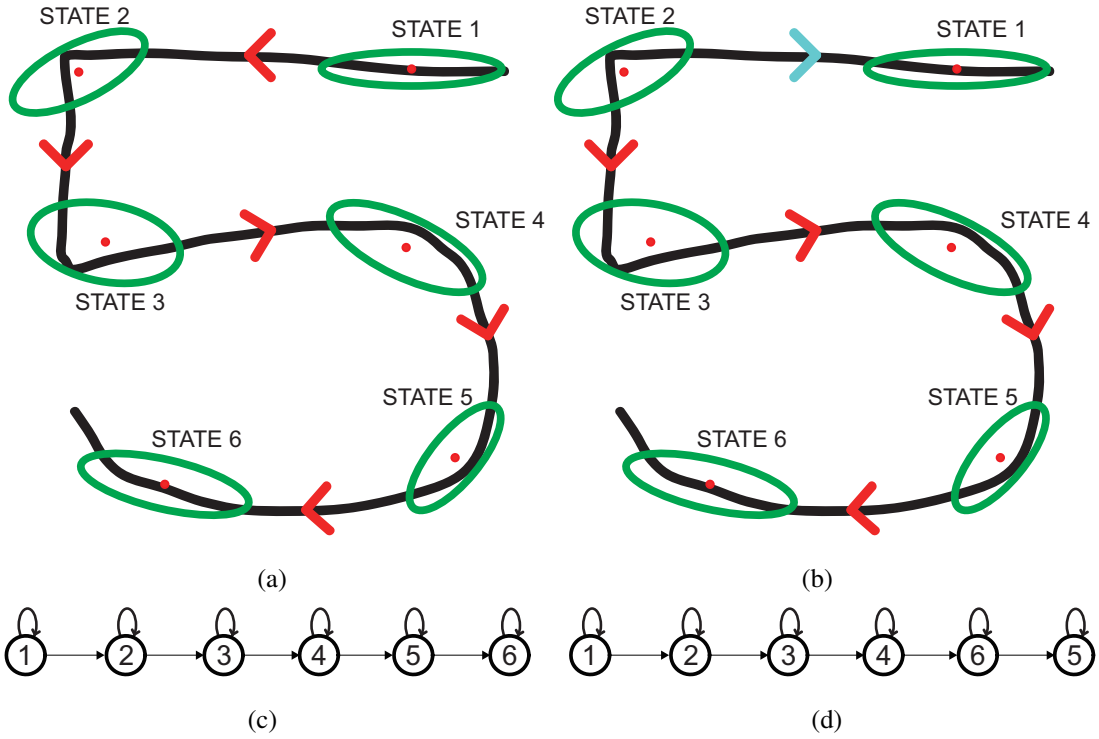


Figure 3.5: (a) path type of gesture ”5” 1. (b) path type of gesture ”5” 2. (c) Type 1 FSM. (d) Type 2 FSM. Both paths represent a gesture of ”5”, but there is a difference in the paths: one recognizes one, and the other does not. Therefore, we created an FSM for both paths, and modified it to recognize it.

may be different according to the range of the actual state, and the order of movement of the gesture itself may vary. However, the basic FSM recognizes gestures by creating one machine in one gesture. Therefore, only one path can be memorized, and when the order of drawing changes, it is recognized as another gesture and processed as a fail. This is shown in figure 3.5. The actual gesture ”5” has two paths, (a) and (b) of figure 3.5. With a basic FSM, only one path sequence out of two paths can be remembered, and only one gesture is recognized as ”5”. However, since both paths represent the path of gesture ”5”, both are required to be memorable. Therefore, we propose a multipath FSM that memorizes multiple paths. Figure 3.5 verifies that the path is not unique. Therefore, you can memorize multiple gestures by making all the sequences of paths that the gesture has using FSM. For example, if you create an FSM for gesture ”5,” we can recognize two gestures by creating two paths with FSM when there are two

Symbol	Meaning
$S^{(k,l)}$	The l th model state set of gesture "k"
$Q^{(k,l)}$	lth model input state set of gesture "k"
${q_1}^{(k,l)}$	l th model initial state of gesture "k"
Λ^k	Output of gesture "k"
$T^{(k,l)}$	A function that returns the next state based on the output state and input of gesture "k"
$G^{(k,l)}$	A function that returns the next output based on the output state and input of "k"

Table 3.1: Mathematical definition of a multiple FSM. l represents the index of the state machine. Each state machine has different functions and state configurations.

paths such as figure 3.5. In this case, the mathematical definition of the multiple FSM of gesture "k" is $(S^l, {S_0}^{(k,l)}, \Sigma^{(k,l)}, \Lambda, T^{(k,l)}, G^{(k,l)})$ and each definition is 3.1.

3.1.1.2 Probability-based-FSM

A multiple FSM is a way to create an FSM for all the paths gestures, and to recognize gestures drawn in various forms. However, because multiple FSMs need to remember all paths, they need a lot of storage space. Furthermore, they cannot train all paths because of a lack of training data, and the time required for recognition may increase, because all FSM results must be checked and the values must be obtained. In order to overcome these disadvantages, we have developed an FSM based on probability. The transition matrix used for gesture recognition stores the probability of moving from state i to state j. When a probability matrix is generated through an HMM, if there is at least one transition from state i to state j, it stores a non-zero probability at a_{ij}. Finally, when the training of the gesture ends, all the movements of the state shown in the whole training are saved. Therefore, if the probability matrix of the transition matrix is 0, assuming that there is no path and if the probability is nonzero, assume that there are paths. This can have an effect similar to an FSM that determines the

gesture based on the existence of the path. The starting and ending positions are also important, as the order is crucial for constructing the gesture. If the starting or ending position is different, it may be recognized as a gesture (without following the whole form of the gesture), which can generate an error. To prevent this, the gesture is made to conform to the whole shape by training the probability of the starting and ending positions of the gesture. Assuming that the two paths shown in figure 3.5 are trained, the path of figure 3.5 (a) is trained first, so that the path that can be transitioned between the states as figure 3.6 and states 1 and 6 end the training. Then, when figure 3.5 (b) is trained, the state transition probability of state 2 to state 1, state 6 to path 2, and the prior probability of state 2 are trained. Finally, when multiple paths between states are trained, such as in figure 3.6 (b), recognition fails when the gesture moves out of state, but all paths are recognized when trained. Therefore, both gestures can be input by the method proposed in this paper. The expression for this is shown in equation 3.5.

$$\begin{aligned} result = true,\ pi_1 * \sigma_T * \prod_{1}^{T-1} (a_{v_t v_{t+1}}) > 0 \\ result = false,\ pi_1 * \sigma_T * \prod_{1}^{T-1} (a_{v_t v_{t+1}}) = 0 \end{aligned} \tag{3.5}$$

3.1.1.3 Gesture decision

Finally, when two or more of the same gestures are input, the similarity between the model and the gesture is determined to distinguish the gesture. When the gestures of the training model move in a similar order, it can be concluded that the gestures input by two or more models are the same as the gestures represented by the model. At this time, a gesture closer to the gesture indicated by the input gesture is judged. The cumulative Gaussian probability is then applied between the model and the gesture, to compare their similarities. The probability-based method is used to train each gesture to obtain a probability parameter, and to determine the presence or absence of a path through

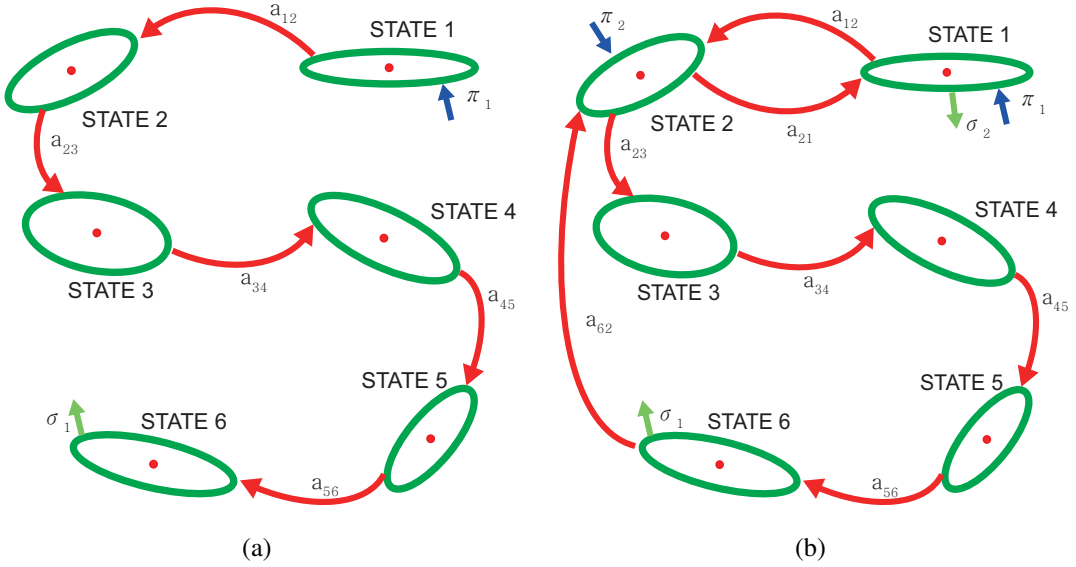


Figure 3.6: (a) The Probability-based FSM which training only type 1. (b) The Probability matrix based FSM which training both type 1 and type 2.

comparison. In order to compare the similarity using a probability based feature, the Gaussian probability density between the point and the state is obtained. The Gaussian probability density formula is as follows.

$$b_t = -0.5 * log(|\Sigma_{v_t}|) - 0.5 * (x-\mu)_{v_t}{}^T \Sigma_{v_k}{}^{-1} (x-\mu_{v_k}) \tag{3.6}$$

Depending on the characteristics of the logarithmic Gaussian probability density function, it is affected by the covariance and has a larger probability density with a smaller distance from the center, and a smaller probability density with a larger distance. Therefore, the probability of similarity between the input path and the center of each state is obtained. The probabilities for the prior, ending, and the state transition are also added, to obtain the probability for the path. The greater the similarity means the more similar are the paths. This is equivalent to equation 3.7.

$$s_k = log(\pi_1) + log(\sigma_T) + \sum_{t=1}^{T}(b_t) + \sum_{t=1}^{T-1}(a_{v_t v_{t+1}})|_{v_t \neq v_{t+1}} \tag{3.7}$$
$$gesture = arg_k max(s_k)$$

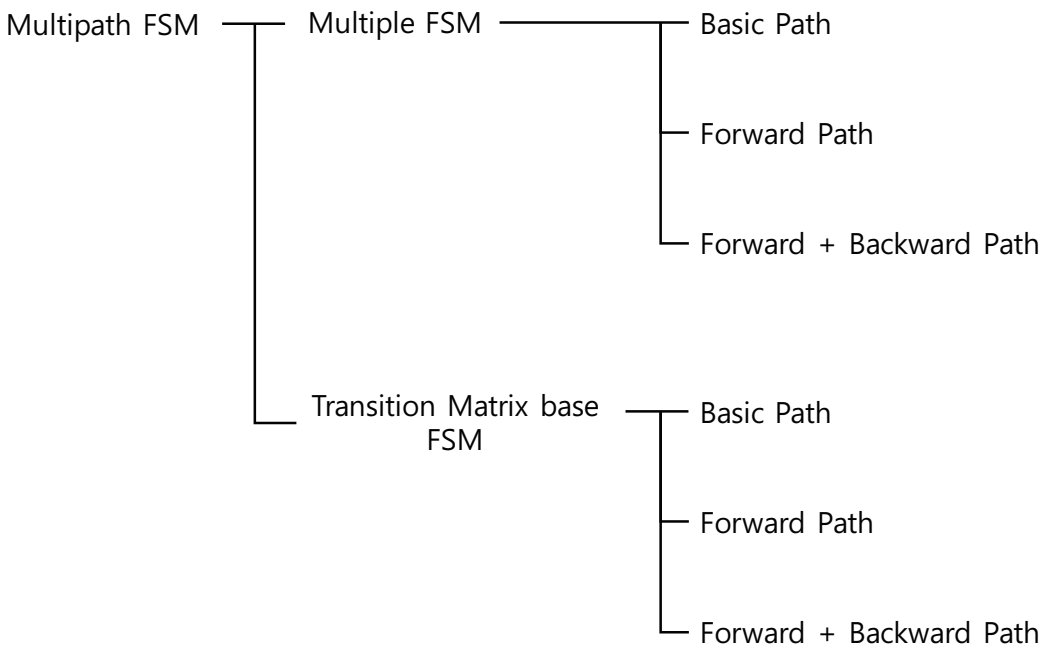


Figure 3.7: A schematic diagram of the entire gesture.

3.1.2 3-dimesional body gesture recognition

In the previous section, we proposed an FSM that recognizes gestures using a single point. It recognizes gestures by tracking a single point, but recognizes gestures by the whole body, not gestures by one hand. We used skeleton of gestures to extend three-dimensional body gesture recognition. We also selected the points needed for the entire skeleton, to take advantage of the many motion points for gesture recognition. The method used to select points and identify body gestures is discussed in Chapter 5.

The methods used in this chapter are summarized in figure 3.7.

3.2 Experiment

3.2.1 Experiment setting

In this paper, we propose FSM, which is more robust against errors and can produce good results with a small number of training data. The proposed algorithm is

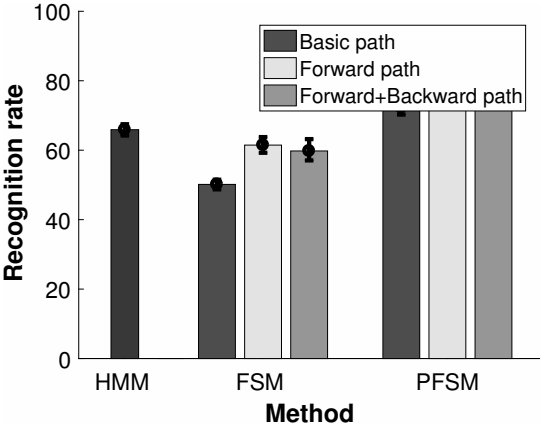


Figure 3.8: Recognition rate of each method with basic path, forward path and forward + backward path.

the probability-based method and it is possible to recognize the gesture with a small number of data. Experiments were conducted using UTD-MHAD dataset to confirm the performance of the developed algorithm. UTD-MHAD dataset extracts skeleton using RGB camera and Depth camera and attaches inertial sensor to obtain inertial data. The data includes 27 gestures that were repeated 4 times by 8 subjects. We conducted a five fold cross-validation to divide the training data and the test data by a ratio of 4: 1 to confirm the recognition rate. The experiment confirmed the recognition rate using the HMM for comparison and the proposed FSM. Also, we confirmed the recognition rate by changing the number of states applied to gesture recognition from 3 to 15.

3.2.2 Experiment Result of FSM

Each method was used to determine the rate of perception used. The recognition rate using the various FSMs is shown in figure 3.8.

First, we confirmed the gesture recognition results according to the three state sequences method. As a result of checking the recognition rate, it was confirmed that the recognition rate improved when there was considerable information used in the path. Unlike the basic path, which uses the state at the closest distance, forward path and forward + backward path are obtained by accumulating the probabilities at various times.

Thus, if different paths are accumulated, the recognition rate is improved through modification, even if the paths do not fully match. It was also confirmed that the recognition results of the probability-based FSM concurred with the existing FSM. The recognition rates revealed that the existing FSM generally achieved a higher recognition rate than the probability-based method. The reason for this is the kind of path that is recognizable. For existing FSM, a gesture is recognized when the entire path in the state matches. However, actual gestures appear in many different forms, and if a few paths are missed, the recognition rate decreases. Alternatively, probability-based FSMs only recognize the path between states, rather than the entire path, based on probability. Consequently, the probability-based FSM can recognize the path between the states where the transition occurred, even if it only occurred once; therefore, it is possible to recognize the paths of various entities. As a result, the overall rate of the perception of gestures is higher in the probability-based FSM, when compared with other methods. Next, recognition was accomplished by changing the state counts of the existing FSM, and the probability-based FSM that was tested earlier. We confirmed the recognition rate by changing the number of states from 3 to 15 (for recognition). The results are shown in figure 3.9.

Figure 3.9 (a) shows the recognition rate according to the number of states of multiple FSMs. Looking at the recognition rates, we can see that the recognition rates of the two methods are different according to the number of states. The multiple FSM showed the highest recognition rate when the number of states was 3. In addition, the recognition rate was significantly reduced, even when the number of states was increased, based on the number of states. Furthermore, in the probability-based method, the recognition rate was the highest when the state number was 6, 15, 15, according to the number of paths. The probability-based method has a smaller decrease in recognition rate than the multiple FSM, and the interval in which the maximum recognition rate is maintained is also wider. Furthermore, the recognition rate of each path shows that the recognition rate is greatly affected by the number of states in the basic path. However, it can

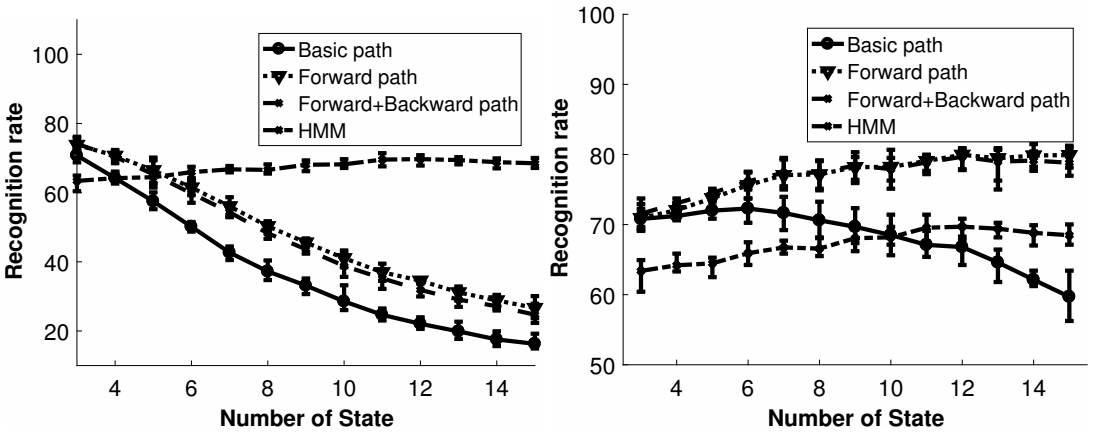
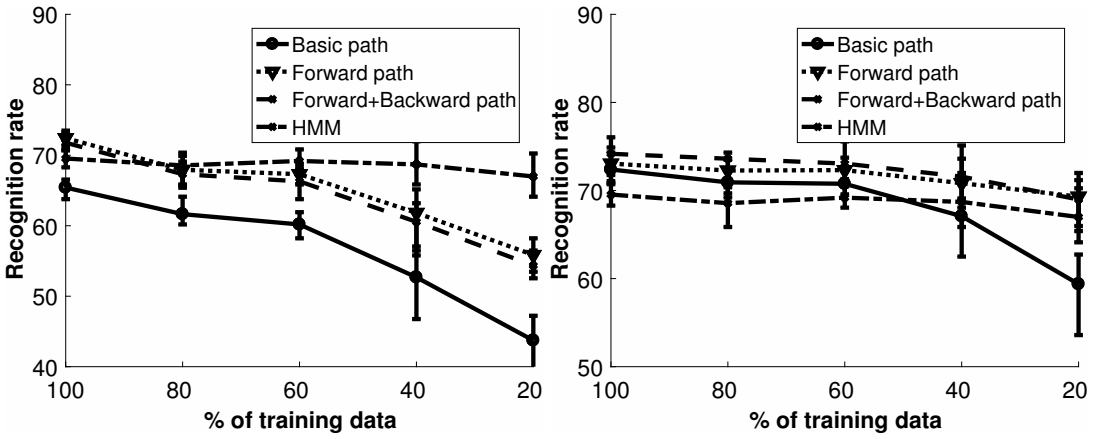


Figure 3.9: (a) The results of the probability-based FSM according to the number of states. (b) The results of the multiple FSM according to the number of states. The probability-based FSM results show that the greater the number of states possible, the better the results. However, the multiple FSM shows the best result with a small number of states.

be seen that the number of states is less influenced when modifying the path through the stochastic parameter, such as forward path and forward + backward path. Theoretically, when the number of states is small, the recognition rate is low due to the lack of information representing the gesture. However, in the case of multiple FSMs, as the number of states increases, the number of state sequence cases of gestures increases, and the recognition rate decreases. Conversely, in the probability-based FSM, the recognition rate of the gesture is improved. Therefore, the multiple FSM had the highest recognition rate when the number of states was 3, but the recognition rate of the probability-based FSM continuously increased.

We can understand the reason for this result by changing the number of training data. In order to change the number of training data, the experiment was verified by varying the ratio of training data to 100%, 80%, 60%, 40% and 20%. The results are shown in figure 3.10.

The results show that as the number of test data increases, and the number of training data decreases, the overall recognition rate decreases. At this time, the recognition re-



(a) (b)

Figure 3.10: (a) The results of the probability-based FSM according to the rate of training data. (b) The results of the multiple FSM according to the rate of training data. The probability-based FSM shows a relatively good result with a small amount of training data.

duction rate of the probability-based method was lower than other methods. Moreover, in the case of a multiple FSM that memorizes sequences for the entire path (rather than probability-based recognition), the recognition rate is greatly reduced as the number of training data decreases. Multiple FSMs can recognize when a path is entered, and when there is a training gesture that matches the path. Therefore, when the number of training data is small, the number of training gestures is decreased, and the recognition rate is drastically lowered. The probability-based FSM enabled gesture recognition when there was at least one instance of information about state movement, making it possible to recognize even when there was little training data. Although the recognition rate did not show a large difference when the number of actual training data was large, it can be confirmed that the recognition rate was reduced by a much smaller amount than the multiple FSM, when the number of training data was small. The rate was affected by these results: the recognition rate of the probability-based method increased when the number of states was increased, although the recognition rate of the existing FSM also decreased. As a result, compared to the traditional FSM (that compares the overall path of the gesture), the probability-based FSM required less likely training data,

which caused an increase in both the state counts and the complexity of the path.

In this chapter we have studied the different FSM methods and how to create different state requirements to apply them. The results displayed that the transition matrix-based FSM had the highest recognition rate. In particular, it had the advantage of being able to achieve high recognition rates with a small amount of training data. Accordingly, compared to FSMs that check the entire route, only the movement between each state was checked to recognize the gesture. As a result of changing the path creation method, to correct the error in the gesture, gestures with a large amount of information had a higher recognition rate. Nevertheless, the recognition rate of gestures was not high, only approaching 80%. The reason why these results are presented is because of the location of the gesture. Each person does not represent a form that is entirely consistent with either the size of the entire motion, or the starting position of each gesture. There are a number of possible ways to calibrate this, but there are limitations, in particular a complete calibration is not possible. As a result, if the position of the gesture deviates significantly (or differs from the general form), the gesture is not recognized. Moreover, the FSM methods presented may reduce the rate of perception of repetitive gestures. By checking the movement between simple states, if the entire route is repeated, it is not possible to reflect the repeated forms, resulting in decreased recognition. However, we observed a relatively high rate of recognition of gestures that do not have repetition.

It is also expected to gain a higher recognition rate by studying how to align the size and position of the gestures.

3.3 Summary

In this paper, we propose a gesture recognition method that recognizes several types of gestures in order to overcome the disadvantages of state - based gesture recognition method FSM. The proposed method is a probability-based FSM that recognizes

gestures by training multiple FSMs that operate and store two or more FSMs in one gesture, and a connection between states that are not entirely sequenced, in order to recognize the diversity of gestures. In order to compare the advantages and disadvantages of the two methods, we confirmed the recognition rate by varying the number of gesture training dates. In the multiple FSM, it was confirmed that the recognition rate of the gesture increases as the training of the path increases. On the other hand, when the ratio of training data is reduced, the recognition rate can not be guaranteed because it can not guarantee the diversity of recognizable gestures. probability-based FSM showed the highest recognition rate when the number of training data was large. On the other hand, when the number of training data decreases, the recognition rate decreases much less. This result influences the experiment of changing the number of states, and the probability-based FSM shows that the recognition rate increases continuously when the number of states increases and that the Multiple FSM decreases when the number of states is more or less than 6.

Chapter 4

Probability-based Dynamic Time Warping for Gesture Recognition and Signal Warping

In this chapter, we propose dynamic time warping based on probability. Conventional dynamic time warping can not be applied to a relative distribution by comparing simple distances between two signals. Therefore, in this paper, we have studied a method that can apply different distances according to distribution by stochastic application. To obtain a stochastic distribution, we need to find the path that is the center of the whole signal. Therefore, the path representing each training data was obtained by three methods. Then, the probability distributions centered on each path were obtained, and the gesture was recognized. This section is being prepared for submission as a paper in journal 'Sensors and Actuator'(Kwon and Kim, c).

4.1 Dynamic Time Warping

To propose the proposed Probability-based Dynamic Time Warping (PDTW) in this paper, it is necessary to develop a DTW which is the basis of the algorithm. Dynamic Time Warping (DTW) aligns two distorted signals in a time series and outputs the dis-

Condition	Meaning
Bonundary	p_1 = (1,1) and p_L =(N,M), The starting and ending points of matching are the starting and ending points of the two signals.
Step size	$p_{l+1} - p_l \in \{(1,1),(1,0),(0,1)\}$, A matching point can be moved by one index at a time.
Monotonicity	$n_1 \leq n_2 ... \leq n_{L-1} \leq n_L, m_1 \leq m_2 ... \leq m_{L-1} \leq m_L$, Matching points do not reverse in time series.

Table 4.1: Three conditions of Dynamic Time Warping. We fix the starting point and the end point by three conditions: suggest a step size that moves at once, make the matching point not to be backward, and to be able to match the whole signal.

tance at that time. To find the target value, you can move the sequence to compare and find the matching points that represent the smallest error at this time. To match the two signals, the DTW creates three conditions: a boundary condition, a step size condition, and a monotonicity condition. The variables to explain this are as follows.

L : Last matching point.

N : Last index of signal 1

M : Last index of signal 2

p_l : First l

(n,m) : The matching point that the index of signal 1 is n and signal 2 is m.

$s1_l$: Index of signal 1 in matching point l.

$s2_l$: Index of signal 2 in matching point l.

$$Rt_k(i) = \sum_{d=1}^{D} \frac{origin\ gesture_d(i)}{D} \tag{4.1}$$

A boundary condition means that the starting point of warping is (1, 1), and the ending point is (n, m), when the lengths of signal 1 and signal 2 are n and m, respectively. This is to limit the starting and ending points of the two signals, so that they match all parts of the two signals, not just some of the parts. In order to ensure the continuity of the two signals, the step size condition limits the previous matching points to (a-1, b), (a,b-1), (a - 1, b - 1). If the step size condition is not guaranteed, it is possible to jump over a part of the two signals. This would mean that only the starting and ending points could be matched, resulting in a small error of the actual dissimilar signal. The last condition (monotonicity), is a condition in which the matching point runs without reversing the time series. Monotonicity is based on the premise that the two signals are not inverted, because they are twisted in the time series and are the same signal. If the monotonicity condition is not guaranteed, the matching points are staggered, and the repetition frequency of the periodic signal is ignored. The three conditions are shown in table 4.1.

The DTW obtains the cumulative distance based on these three conditions. Dynamic programming is utilized to accumulate the distances at which the best conclusions are drawn, to obtain the optimal path. This is accumulated from the first value of each signal to the end, to find the optimum value between successive points. The process of obtaining this is shown in figure 4.1. Ultimately, the route that is found has two matching points, two signal-signal 1 and 2, stored and can be used as a similarity between the two signals.

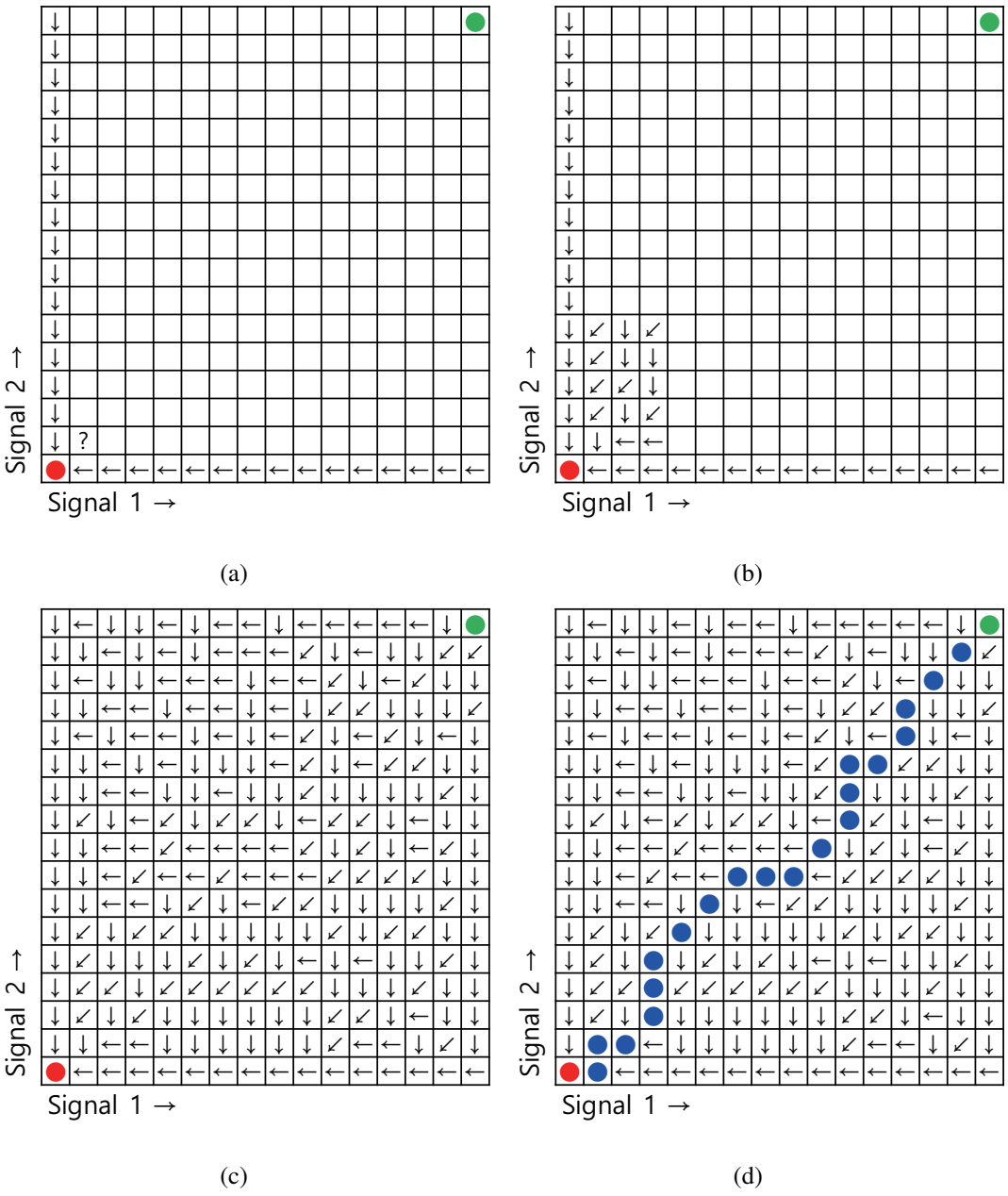


Figure 4.1: The process of DTW.(a) Save the distance of (n, 1), (1, m) (b) To make the distance of (i, j), use the smallest distance of (I - 1, j), (i, j - 1), (I - 1, j - 1) (c) repeat until it reaches (N, M) (d) Backtracking for finding the best matching path.

4.2 Representative Path Generation

The probabilistic DTW uses the distribution probabilities based on one path, so one path generation is required to represent each gesture. For setting gestures to represent the training data, we propose three methods: time mean, length mean, and repeated

warping.

4.2.1 Time Mean Representation

The average of the positions at each time of the initially acquired data was obtained by the method used first. This process is shown in equation 5.3.

$$Rt_k(i) = \sum_{d=1}^{D} \frac{origin\ gesture_d(i)}{D} \tag{4.2}$$

Rt_k of equation 5.3 represents the path representing gesture k, $origin\ gesture_d$ represents the i-th index of the normalized signal, and D represents the length of the entire data. The gesture obtained in this way has an average position of the gesture at each time. This is the most basic gesture path training method using averaging, and is a relatively simple method.

4.2.2 Length Mean Representation

However, a representative gesture that averages data by index might not produce a gesture at a representative point, where the matching points of the two signals are different when the speed varies with time. Therefore, the second method uses gestures with the same interval, according to length. In the case of length normalization with equal spacing according to length, assuming that each gesture has the same shape, it is located at a comparable position when the index of the gesture is equal to the normalize along the length. The equation for finding the path representing each gesture is shown in equation 4.3.

$$RL_k(i) = \sum_{d=1}^{D} \frac{length\ gesture_d(i)}{D} \tag{4.3}$$

Here, RL_k represents the path representing the gesture "k", and $length\ gesture_D(i)0$ represents the gesture evenly divided by the length. When the representative path is obtained by using evenly divided gestures according to the length, the center of the entire gesture can be found by arranging the index equally (according to the position), as compared with the time mean representation.

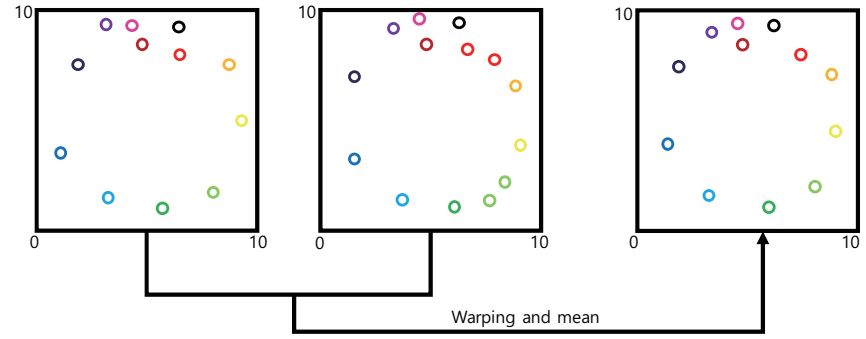
4.2.3 Repeated Warping Representation

As a final method, we used DTW instead of simple averaging, to re-estimate and relocate new locations through path matching. Since DTW represents a 1:1 match of two signals, training was performed by repeating the process of comparing two gestures and repositioning them. Figure 4.2 shows this process. Figure 4.2 (a) shows the process of finding the matching points of two gestures and repositioning them. When two signals are input, we obtain the points of signal 2 that match the i-th index of signal 1. After finding the matching point, i-th index of signal 1, and the average position of signal 2 which matches with i-th position of signal 1 are found. By repeating this process with a ratio of g:1 when train g-th training gesture and trained gesture, we can obtain a final representation path with a warping gesture. This process is shown in figure 4.2 (b).

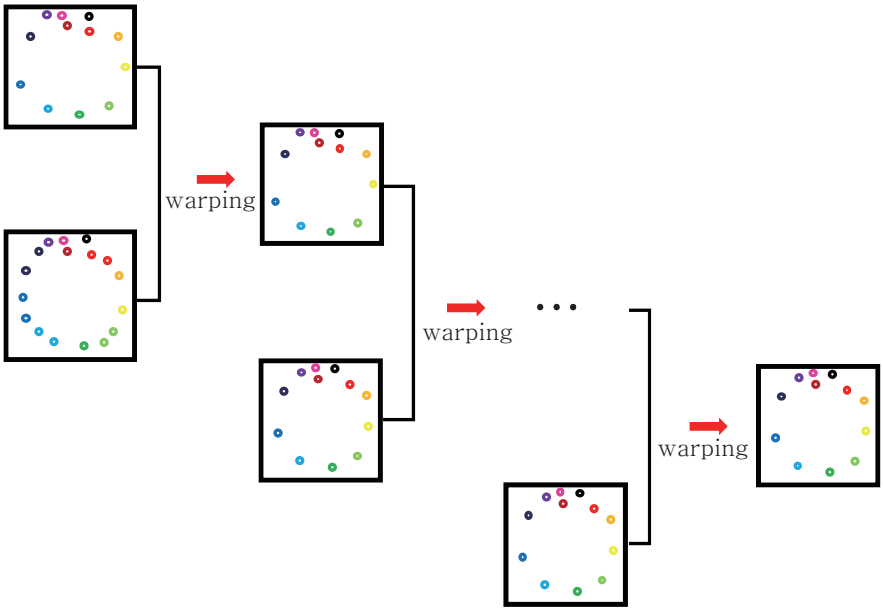
In this process, if a gesture is represented by a path that is representative of each gesture, training can be performed while warping the training data, by matching with a similar point through warping, even if the shape of the gesture is slightly different.

4.3 Probability based Dynamic Time Warping

Finally, to construct the probability model, DTW between the representative path and each training data is processed, and a probability model is obtained. The trained representative path passes through the center of each training data. At this time, assuming



(a)



(b)

Figure 4.2: The process of making repeated warping representation path.

that the points matching on the representative path follow the Gaussian model, the mean and variance of the points that match each index of the representative path (with DTW) are obtained to make a Gaussian model. The equation of the Gaussian model is as follows.

$$f(x) = \frac{1}{(2\pi)^{D/2}|\Sigma|^2} exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\} \tag{4.4}$$

At this time, the Mahalanobis distance was used to simplify the gaussian distribute function. The equation of the Mahalanobis distance is as follows.

$$D = \sqrt{(x-\mu)\Sigma_{-1}(x-\mu)^T} \tag{4.5}$$

Unlike the conventional Euclidian distance, the Mahalanobis distance allows you to distinguish between shapes in terms of covariance. Using these advantages, we can obtain a model of the point where each point is matched, and apply DTW to the probability distance to recognize the signal whose distribution changes according to the shape.

4.3.1 Multipoint gesture recognition

We used a skeleton to recognize 3D gestures. Assuming that each point of the skeleton is probabilistic independent, we trained each point separately and added Mahalanobis distance obtained from each point. In addition, when using all the points, we apply the gesture recognition to the selected points in order of many movements in order to solve the problems caused by the increase of dimension and fitting. The method for selecting points will be described in detail in chapter 5. The similarity used in gesture recpgmotion is expressed in equation 4.6.

$$S_g = \sum^{M} \{D_p^g\} \tag{4.6}$$

In this equation, S_g represents the similarity with the gesture g, M is the set of points selected in gesture g, p is the element of M, and D_p^g is the similarity with the p-th skeleton of the trained gesture g.

4.4 Expreiment Setting and Dataset

To verify the representation of the proposed representation path, we compared the distance difference between each training data and the representative path. We then

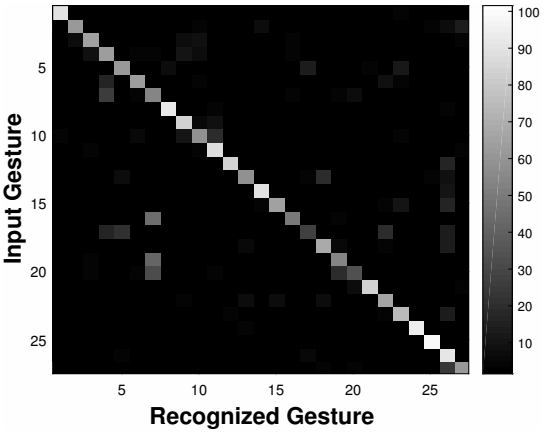
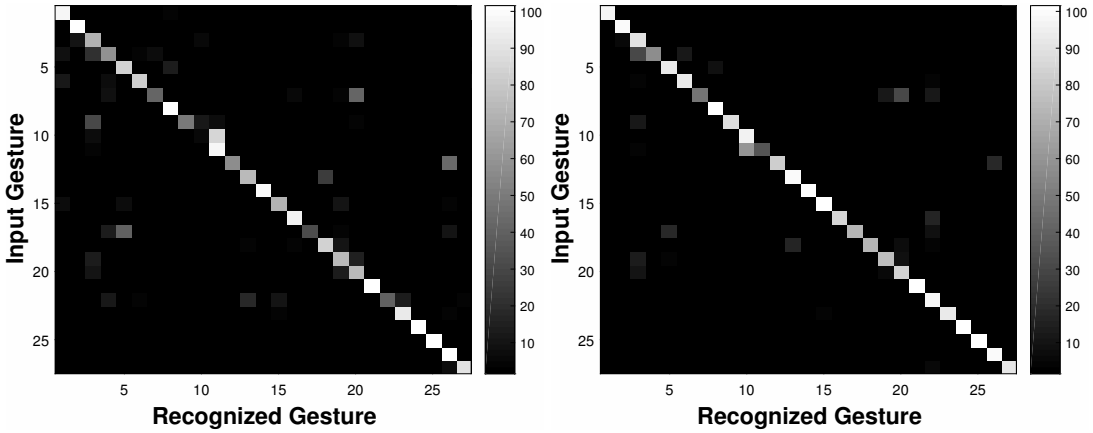


Figure 4.3: Recognition rate of the HMM. The vertical index means ”input gesture and” the horizontal index means ”gesture that input gesture recognized.”



(a) (b)

Figure 4.4: (a) Recognition rate of basic DTW. (b) Recognition rate of the probability based DTW with time-mean representation path. The vertical index means ”input gesture”, and the horizontal index means ”gesture that input gesture recognized.”

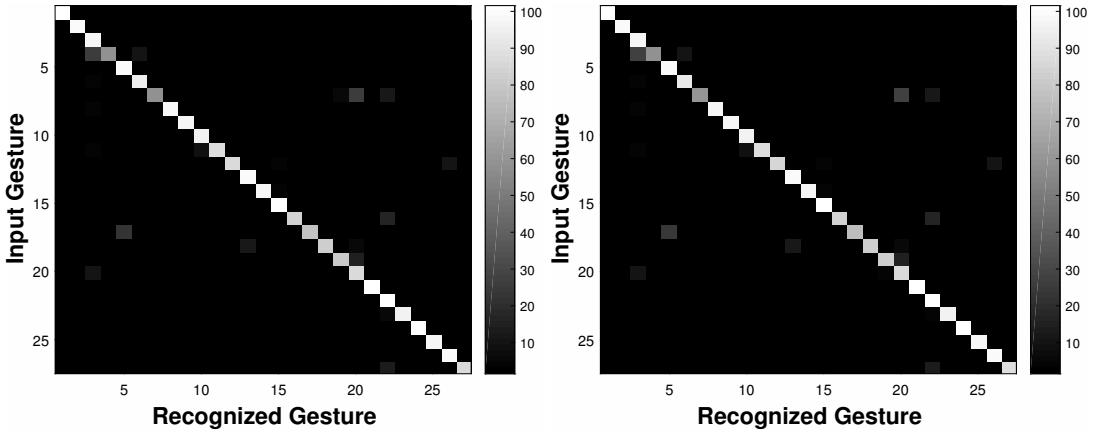
conducted a gesture recognition experiment to compare the results of Probability-based DTW. We used UTD-MHAD for the experiment as in the previous chapters. The index, name and description of the dataset are described in Appendix A.

gesture index	1	2	3	4	5	6	7
Time mean	30.90	36.65	32.07	27.64	28.32	32.91	24.17
Length mean	27.07	30.20	28.66	24.73	25.38	26.80	21.64
Repeated Warping	**25.38**	**29.77**	**26.85**	**23.91**	**25.38**	**26.51**	**21.22**
gesture index	8	9	10	11	12	13	14
Time mean	27.41	42.41	36.61	23.41	23.78	37.57	41.96
Length mean	21.59	41.81	30.61	17.70	17.59	31.81	38.88
Repeated Warping	**20.84**	**30.56**	**24.28**	**17.19**	**16.87**	**28.80**	**36.10**
gestur indexe	15	16	17	18	19	20	21
Time mean	42.94	21.44	35.65	25.97	21.96	24.81	25.44
Length mean	38.52	16.41	29.17	22.72	20.74	23.51	22.22
Repeated Warping	**36.41**	**15.94**	**28.00**	**22.71**	**19.21**	**23.28**	**21.83**
gesture index	22	23	24	25	26	27	Mean
Time mean	51.46	40.62	27.90	25.78	21.28	37.53	31.39
Length mean	54.06	45.37	22.62	**19.88**	18.87	31.14	27.77
Repeated Warping	**42.97**	**28.66**	**22.31**	19.89	**18.52**	**30.29**	**25.36**

Table 4.2: The distance of each representation path according to gesture. Time-mean representation path is the path that has the lowest average distance.

4.5 Experiment

First, in order to find out the accuracy of the representation path obtained by the three methods, the average of the distance between the paths obtained by each method, and each gesture and representation path, is shown in table 4.2. The results show that the time mean representation has the greatest distance, and the repeated warping representation has the smallest distance. Moreover, in the case of the length mean representation and the repeated warping representation, the position matching portion is added to have a relatively small distance value. This result shows why the length mean representation and the repeated warping representation path are superior, compared to the other representation paths. To analyze this, the position according to the index of each gesture was calculated, and compared with the representation path.



(a) (b)

Figure 4.5: a) Recognition rate of the probability based DTW with length mean representation path. (b) Recognition rate of the probability based DTW with repeated warping representation path. The vertical index means ”input gesture”, and the horizontal index means ”gesture that input gesture recognized.”

First, we checked each representation path. Recognition results displayed the smallest distance sum of repeated warping representation, and the greatest distance sum of time-mean representation. The reason for this is shown in figure 4.6. Figure 4.6 represents the trained representation path. Figure 4.6 (a) shows the time-mean representation. In the time-mean representation, the time-mean representation path is centered during the linear motion of the gesture, but is not represented by the movement of each gesture when the curve is drawn. The reason for this appearance is that it is twisted on the time axis. Figure 4.6(b) shows the point corresponding to the 60-th index of the three gestures. The blue gesture has not yet reached the end; the x-axis coordinate is 5.80, the orange gesture is 7.70, and the red gesture is past the point. When the data is distorted at each time axis, and different data are plotted, some of the training gestures are still inward because they have not yet reached that point, and some are located inward because they pass the point. In this case, the number of relatively inward gestures is larger than the number of outward gestures, and the average position is shifted inward. If this happens in the area where the curve is drawn, all the data will be inwardly biased.

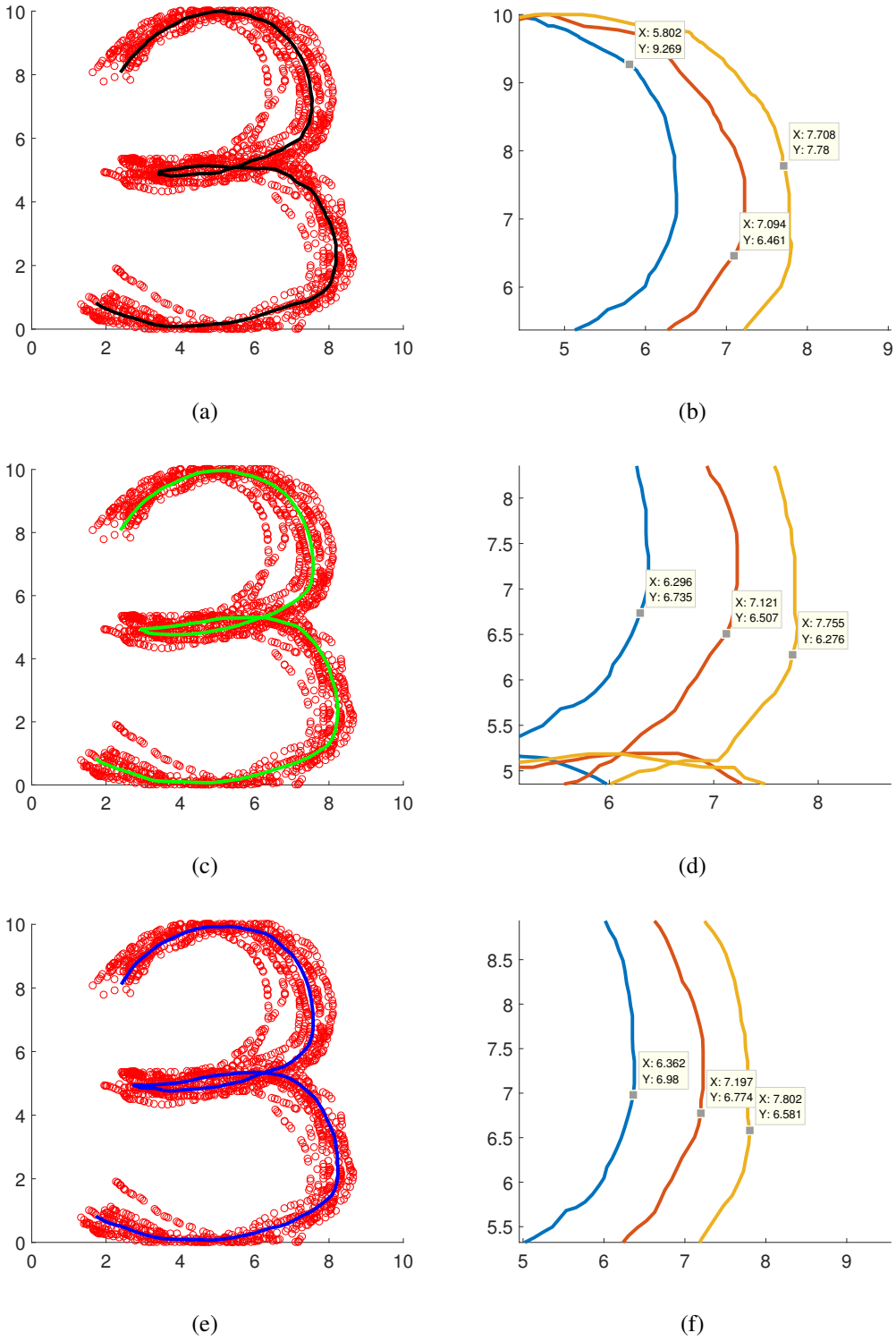


Figure 4.6: The gesture ”3” and representation path of (a) Time-mean representation, (c) Length mean representation, (e) Repeated warping representation. Each path has a similar shape for each gesture.

Next, the length mean representation is shown in Figure 4.6 (c). In the length mean representation, it can be seen that the shape that is biased inwards is relatively small compared to the time-mean representation. Figure 4.6 (d) shows a similar approximation of the approximate location as in the figure 4.6 (b). In the case of the length mean representation, each gesture is normalized according to the length, and the position of the gestures are resampled by the length of gesture. When the gestures are sampled by length, we can see that each gesture point is split in a similar place, and the result is closer to the center of the whole gesture than in figure 4.6 (b). Finally, the result of repeated warping is shown in figure 4.6 (e). Repeated warping is similar to the length mean representation. When training gestures through repeated warping, warping of the position is possible, and gesture distortion such as time mean-representation can be reduced. Next, we applied this to real gesture recognition, to check the recognition rate of the proposed algorithm. For comparison purposes, the HMM and DTW were used, and the probability based DTW was performed using each representation path. Each recognition result is shown in figure A.5, 4.5. In the recognition results, the HMM had a recognition rate of 69.71 %, DTW had a recognition rate of 76.45 %, the use of length mean representation had a recognition rate of 91.25 %, the time-mean representation had a recognition rate of 86.89 %, and the repeated warping method had a recognition rate of 90.87 %. The HMM compares the entire path and identifies the representative state without recognizing it. Particularly when the shape (or orientation) of the overall gestures were similar, for example the gesture to ”Draw circle” or ”Draw triangle”, gesture recognition failed. However, the recognition rate of DTW methods are relatively high. The basic DTW, which had the lowest recognition rate of the DTW methods, still achieved an approximately 8% higher recognition rate than the HMM. However, the recognition rate of the gesture is also lower than the probability based methods. In particular, ”draw circle” and ”draw triangle” displayed a significant decrease in recognition rate when the fitting gesture had a large variance.

In the case of length mean representation and repeated warping, 91.25% and 90.87%

were recognized, respectively. In the case of length mean representation with the highest recognition rate, each path represents a path representing a gesture. Repeated warping representation also showed a similar recognition rate to the length mean representation. Time-mean representation achieved a lower recognition rate than both the length mean representation and repeated warping representation, in gestures that have many direction changes, for example ”draw triangle”.

As a result, the recognition rate of the probability-based DTW is higher than both the DTW and HMM methods. The HMM compared some paths, and trained state probability models, rather than comparing whole paths. Also, the HMM did not train the path of the whole data, but divided some states and recognizing was based on the state. When the gesture was recognized by the state based method, the state could not represent the whole gesture path, so when an error occured near the state edge, it displayed a weakness. This reduced the gesture recognition rate.

However, with dynamic time warping, it was advantageous to perform a 1:1 comparison between the entire data and the data. The algorithm finds the optimal matching point from the beginning to the end, then finds the error at that point. In this way, the recognition rate can be improved by taking advantage of the overall shape characteristics, as compared with the HMM using the partial part. However, the existing DTW also achieved a lower recognition rate than the probability-based DTW. For the gestures ”draw circle” and ”bowling” in particular, the larger the variance of the gesture, the greater was the difference in recognition rate. In such cases, the gesture itself spread over a wide range, but was always calculated at the same distance, and the recognition rate was lowered when the axis was deflected. However, with the probability-based DTW, it was possible to apply it to variance, and recognition was possible even if the axis deviated from a certain range. In each path of Proabability DTW, the recognition rate of Time mean Representation was the lowest, and Length mean Representation and Repeated Warping Representation showed similar recognition rates. The overall recognition rates are summarized in Table 4.3.

Method	Recognition Rate	Standard Deviation
HMM	69.54	0.91
DTW	87.25	0.45
PDTW (Time mean)	76.57	0.21
PDTW (Length mean)	90.74	0.47
PDTW (Repeated warping)	90.79	0.58

Table 4.3: Summary of gesture recognition results.

4.6 Summary

In this paper, DTW which is a method to recognize gesture is developed and applied based on probability. Utilizing probability to apply variance to the gesture makes it possible to create a difference when the axis is slightly deflected, and when the gesture path is greatly deflected. We acquired the central path to obtain the variance. We used time-mean representation as the first method to obtain the path representing the gesture. Time-mean representation samples each gesture, with fixed numbers over time, and finds the average position of each gesture at each index. The data obtained in this way are mostly in the center of the gesture, but they are distorted inward when the direction of the gesture is changed. This is because when the same gesture is drawn, the speed at which the gesture is drawn varies depending on the subject, and when the index is constant, the gesture position is changed, so the area is distorted inward.

The second method was to use the length mean representation. For the length mean representation path, the sampled gestures over time were relocated along the distance, to make the data evenly spread. The evenly spread data could solve the information imbalance problem in the data concentrated in a part. Also, since the data itself is normalized according to the position, it can be confirmed that the path indicated by each data is somewhat consistent even though the whole warping does not proceed separately. According to position, the matched data averages the center path of each data and does not deviate from the center of the data.

Finally, we used the repeated warping representation path. In the repeated warping representation path, a new path is created by the 1:1 matching of two signals, to use all the information of each path, and the newly generated path is updated by warping with the next path. As the update progressed, we changed the weight, based on the number of gestures already trained. The advantage of creating paths through repeated warping is that all points are obtained in the optimal matching state. Each point is trained in an optimal matching state, and the effect of finding the average position (between the samples at the nearest position) is obtained. Therefore, the path that is finally generated can represent each gesture. The repeated warping representation was stronger than the other representation paths in the curved part of the data, and a path representing the entire data was generated.

In addition, there was a problem that the same error accumulated due to the simple distance calculation, even where the data was dense when a similar type was displayed, due to the recognition using the simple distance (caused by the problem of dynamic time warping). We used a probability-based approach to improve this problem. With dynamic time warping using probabilities, although it may be similar, it is possible to apply a recognition that considers the dispersion from the center point of the entire path. When applying variance, fewer errors were added where the gesture training data was spread widely, and a large error was added even if the distance was the same in a narrow position. With the actual probability-based DTW, the recognition rate was greatly improved compared to the conventional DTW.

Chapter 5

Multi-point Gesture Recognition

In this chapter, we extend the gesture recognition that tracks the previously described two-dimensional position in one point to a three-dimensional one using a multipoint tracking method. . We applied and experimented with the problem of normalization and the need to select data in various ways. For the normalization, we carried out various experiments, such as fitting each person's key or applying an affine transformation, and then proceeded with the distance and dispersion to select points. This section is being prepared for submission as a paper in journal 'Sensors and Actuator'(Kwon and Kim, b).

5.1 Data set and Method

We used the University of Texas at Dallas Multimodal Human Action Dataset (UTD-MHAD) for the experiment. The University of Texas at Dallas Multimodal Human Action Dataset Twenty-seven gestures were performed by eight subjects four times. To check the recognition rate of each normalization method, we checked the recognition rate and analyzed it with the transition matrix-based FSM proposed in chapter 3. Each gesture and label are shown in Table 5.1.

gesture index	gesture name	gesture index	gesture name
1	Swipe left	15	Tennis swing
2	Swipe right	16	Arm curl
3	Wave	17	Tennis serve
4	Clap	18	Push
5	Throw	19	Knock
6	Arm cross	20	Catch
7	Basketball shoot	21	Pickup and throw
8	Draw X	22	Jog
9	Draw circle (clockwise)	23	Walk
10	Draw circle (counter clockwise)	24	Sit to stand
11	Draw triangle	25	Stand to sit
12	Bowling	26	Lunge
13	Boxing	27	Squat
14	Baseball swing		

Table 5.1: Index and names of gestures.

5.2 Problems of Three-Dimensional Multipoint Data

The biggest problem when applying three-dimensional multipoint data is how to match the initial positions and points of each point. Figure 5.1 shows subject 1 and subject 2 without any gesture execution. Figure 5.1 is part of the RGB camera that captured subject 1 and subject 2; the blue skeleton at the right represents subject 1, whereas the red skeleton represents subject 2. In this case, the two skeletons show two major problems. The first problem is that the distance between the points of the skeletons is different.

The size and ratios of each subject are different on the whole screen in Figure 5.1 (left image). When the two skeletons are extracted from the RGB image, it is necessary to match the distances between the points because the height of each person is different

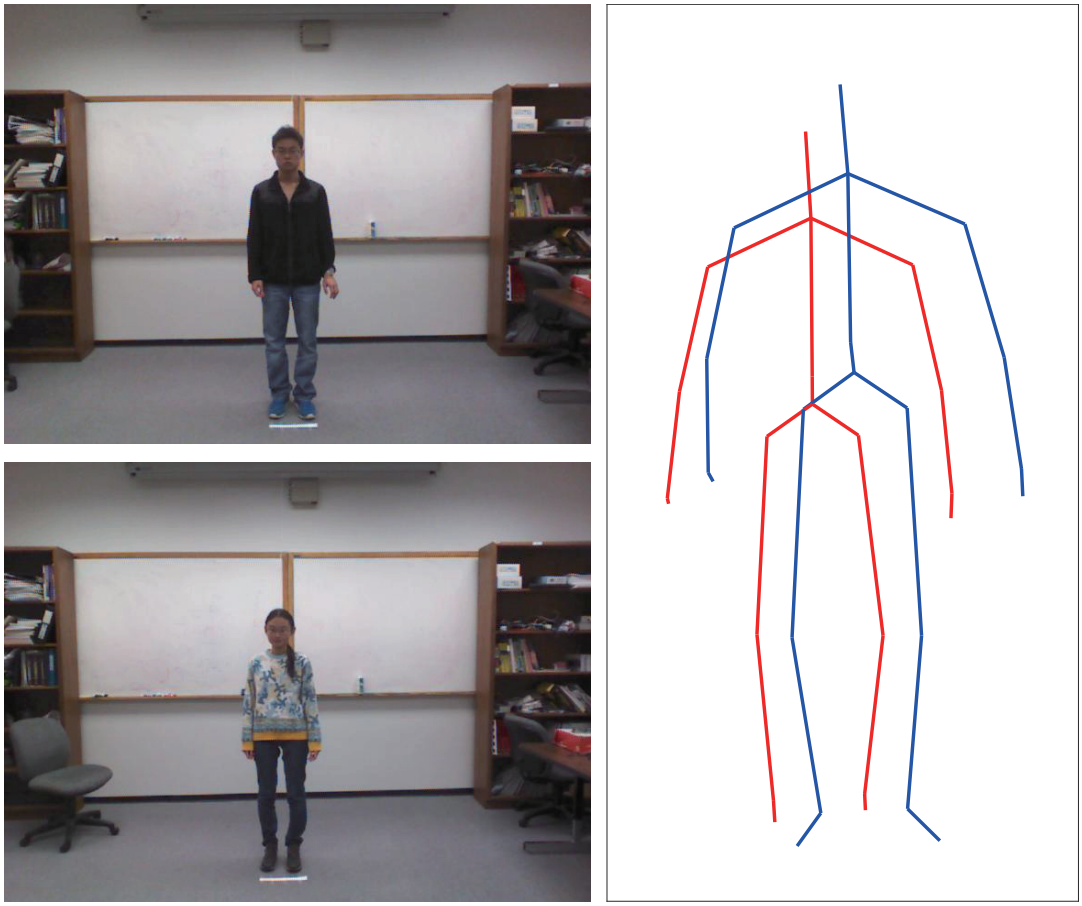


Figure 5.1: The skeleton of two subjects. There are unmatch at each skeleton and joint length. This is because every subject has different stature.

and the length of the joint is also different. The second problem is that the starting point of each skeleton gestures are different. If we look at the location of each point, we can see that all the positions of the points are slightly different. This is one of the problems derived from the first problem, in which the length of the whole joint is different and there is a difference in the distance between each skeleton point, and, thus, the initial position is different. These two problems may not make much difference to people's visual perception, but they can be a major factor in the failure in gesture recognition based on the actual location.

We show another problem in Figure 5.2.

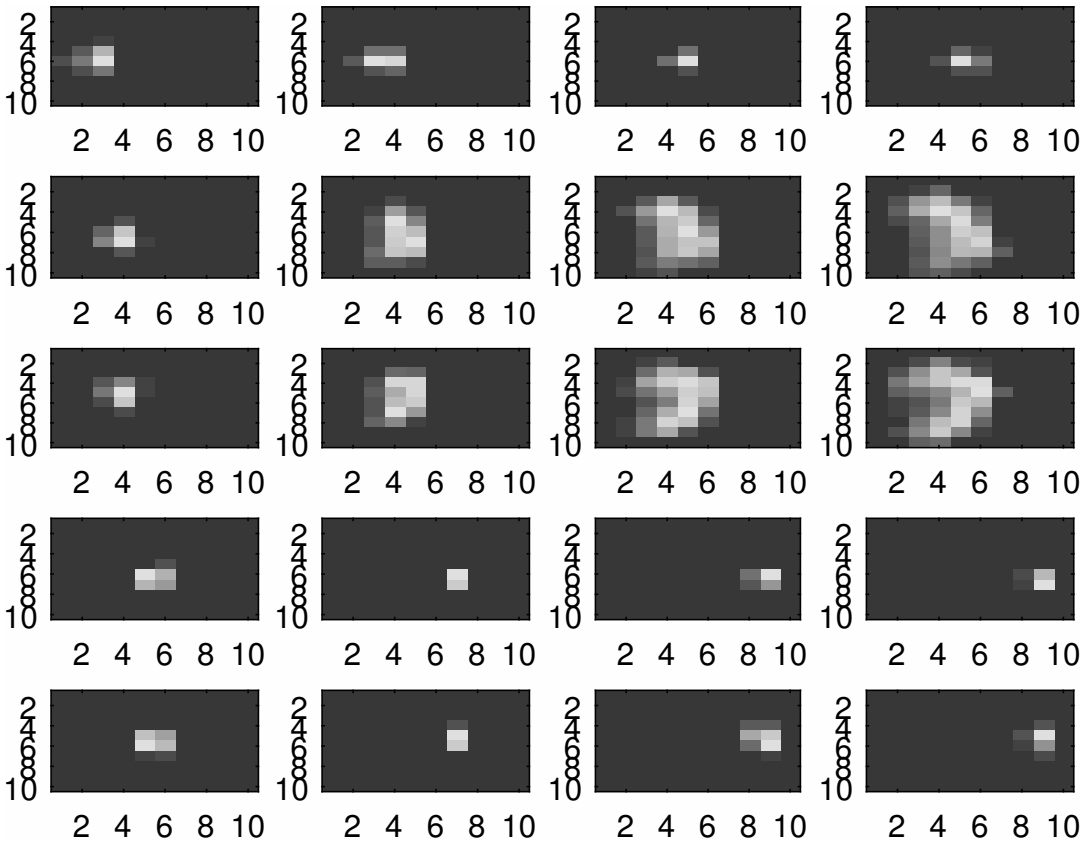


Figure 5.2: The 2 dimensional histogram of gesture 'Baseball Swing'. Each Image means accumulated image of each skeleton point.

Figure 5.2 shows a two-dimensional histogram of each skeleton of the gesture "baseball swing". There are 20 points for gesture recognition, although we actually need only the points that have a large amount of information for each gesture. Thus, the amount of information that each point has is uneven. The main piece shows that the actual moving points are spread over a wide range by the starting position, size, etc. Some points are also stationary, with little movement. Consequently, it is necessary to match the selection process and location to the point. To solve the above problem, we will study various gesture fitting methods and use data selection in this chapter.

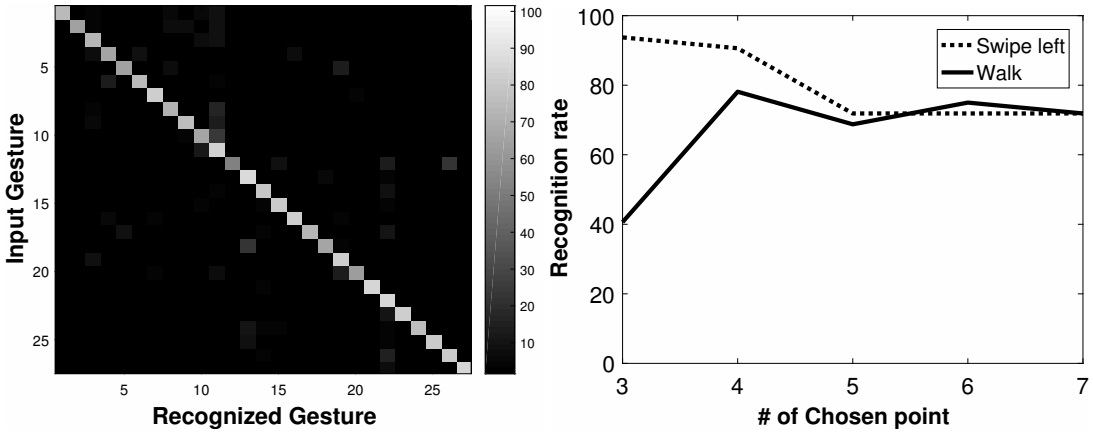
5.3 Gesture Point Selection

As shown in the previous section, there is an imbalance in the information between the part where the gesture actually operates and the part that does not work. There are gestures that use the whole body, and there are gestures that use only the upper and lower body or the left hand or right hand. Therefore, in this section, we select a gesture point with many movements and apply it to actual gestures.

5.3.1 Variance selection

For the first method for choosing a gesture, we used the variance of each point. We will assume that the larger the gesture movement is, the more information it has, and that the more the gesture moves, the wider the range is and the greater the variance is. Let us look at Figure 5.2. The gesture “Baseball swing” is a movement that involves the left and right hands from left to right. Thus, the data in the left hand will be in a wide range (top 3, left 2), as shown in Figure 5.2, and will have a large variance. On the other hand, the gesture only involves the movement of the left hand, and, therefore, the movement of the head is relatively small and the variance is also small. In this way, the larger the movement, the larger the variance that is used to select more moving points. To check the results, we show the result of the selected three point and the increase in the number of selected points from 3 to 7 in the figure.

Figure 5.3 (a) shows that the recognition result was shaky. To analyze the reason for this result, we show the results of the gestures “Swipe left” and “sit to stand” in Figure 5.3 (b). The results show that the recognition rate decreased as the number of points increased in the gesture “Swipe left” and that the recognition rate decreased as the number of points used in the gesture “Sit to stand” decreased. The gesture “swipe left” is a gesture that moves the left hand from left to right. The actual number of points to be moved is two, and as the number of points increases, the recognition rate decreases.



(a) (b)

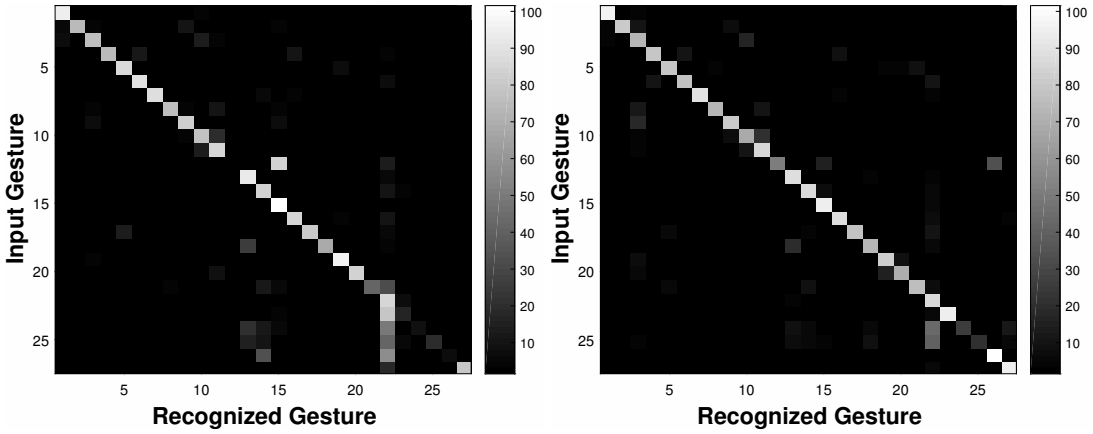
Figure 5.3: The result of Variance based point choose. (a) The result of gesture recognition. (b) The result of gesture 'Swipe left' and 'Sit to stand' with change number of selected point. When the number of selected points increases, gesture 'Swipe left' decreases recognition rate but gesture 'Sit to stand' increases.

In contrast, the gesture "sit to stand" involves transitioning from a sitting to a standing position. The actual sitting position is the movement of the whole body, and there are many parts to move. Therefore, the greater the number of points used, the better the recognition rate is.

To compensate for these shortcomings, we decided to focus not on the number of points selected but on the size of the variance. The variance threshold was set and recognized using points having more than a certain threshold. The recognition results are shown in Figure 5.4 (a).

It can be seen that the recognition result had a higher recognition rate compared to the previous result. The number of points selected for each gesture is shown in Table 5.2.

If we look at Table 5.2, we can see that the number of points selected for each gesture was different. However, in the gesture "Walk," it can be seen that it has fewer selected points compared to the 'Jog" gesture, even though the former gesture moves only the left hand. This is due to the small movement of the overall gesture compared to the threshold of the selection criteria. If we lower the threshold to correct this, we need to



(a) (b)

Figure 5.4: The result of Variance based point choose with threshold. Result show better than fixed point choose.

gesture index	1	2	3	4	5	6	7	8	9
Chosen number	3	3	3	4	3	5	6	3	3
gesture index	10	11	12	13	14	15	16	17	18
Chosen number	3	3	3	20	6	6	3	6	6
gesture index	19	20	21	22	23	24	25	26	27
Chosen number	3	3	13	4	1	14	14	20	6

Table 5.2: The selected number of point in each gesture.

apply the points that do not need other gestures to the recognition, and the result is that the recognition rate also decreases.

Therefore, we have to select points with a certain rate of motion based on the point with the largest variance. In general, for small gestures, it is possible to lower the standard of the point used and to raise the standard when there is a certain level of movement, thus enabling a flexible selection. The result of applying it to the gesture recognition is shown in Figure 5.4 (b).

It can be seen that the recognition rate improved as the selection of actual gestures became relatively accurate. In particular, the recognition rates of the gestures “Walk” and “Jog,” which perform small movements across the body, increased dramatically.

By increasing the possibility of choosing a gesture that has little absolute movement, the method was able to improve its recognition rate. As a result, the number of dots chosen to hold a large amount of information is reduced, and a large number of points are used to select the actual gesture; however, the information may be insufficient. Many points can be selected on the basis of the maximum movement instead of the absolute standard. Gestures with small movements are less likely to have more than an absolute standard.

5.3.2 Speed summation

For the method of selecting a gesture, we used variance to select the gesture movement in the previous subsection and applied it to the gesture recognition. However, the use of variance does not distinguish between gestures that move continuously over a small range. Let us assume a gesture that has a repetitive motion in a narrow range. When we use variance to judge a gesture movement, we get a small variance by taking repetitive movements in a narrow range. Therefore, the gestures that have repeated movement in a small range are not chosen as a gesture that has a large amount of information. However, there are many repeated movements with only a small range. When we have repetitive motion in a narrow space, we have a lower variance compared to a single motion in a wide range. To compensate for these drawbacks, we measured the amount of information by accumulating the speed of the gesture and finally moving the distance. In this way, it is possible to pick out the important points of the gesture of a repetitive motion in a small range. We applied it to actual gesture recognition and confirmed the result. As in the previous subsection, we selected points that move more than a certain amount based on the movement distance of the movement point. The gesture recognition result is shown in Figure 5.5. The recognition rate was higher than that using the previous variance. Moreover, the number of points selected by the method using variance and speed for each gesture is shown in Table 5.3.

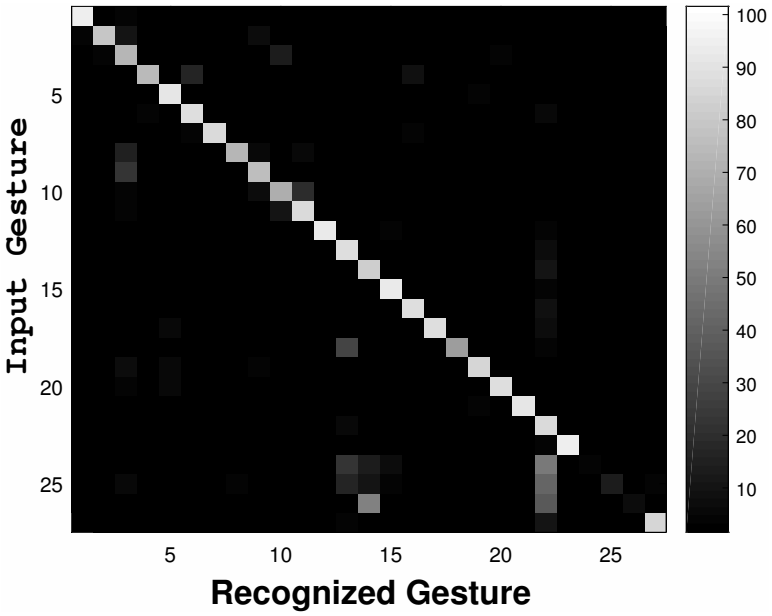


Figure 5.5: The recognition rate of speed based gesture selecting.

gesture index	1	2	3	4	5	6	7	8	9
Variance	2	2	2	4	2	3	4	2	2
Speed	2	2	2	4	3	4	5	2	2
gesture index	10	11	12	13	14	15	16	17	18
Variance	2	2	10	4	4	2	4	4	4
Speed	2	2	3	6	5	3	4	5	6
gesture index	19	20	21	22	23	24	25	26	27
Variance	2	2	2	4	4	11	14	3	4
Speed	2	3	3	4	4	14	15	20	6

Table 5.3: The distance of each representation path. Time mean representation path is path that has the lowest average distance.

From the results of each extraction, it can be seen that a speed-based gesture extracts the operating point better. For instance, in the case of the gesture “Lunge,” it is a full-bodied gesture. Therefore, gestures should be able to extract the entire body. Regardless of the total distance traveled, however, it is necessary to determine the distance away from the center of gravity. Therefore, it does not represent the whole movement. On the other hand, when the speed method is used, the moving distance from the start

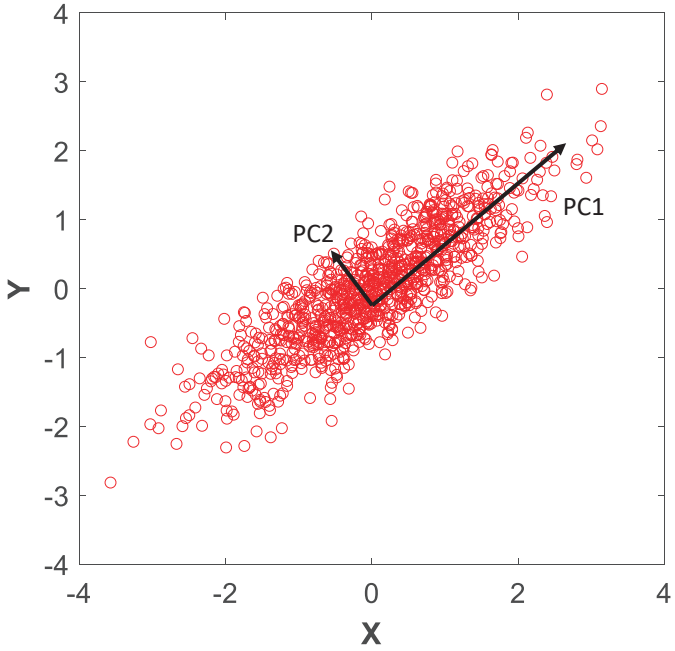


Figure 5.6: Gaussian random data. Arrow of this figure is good to represent the data. When we use the axis with arrow, we can reduce the dimension and we call this arrow as a 'Principal Compornant'

point to the ending point is obtained to represent the information amount of the signal. For the method that uses speed, the exact number of points to be extracted is 20; for the variance, however, only 4 points are extracted. As a result, it can be confirmed that the highest recognition rate was obtained when speed was used.

5.3.3 Principal component analysis

PCA is a method of extracting and applying information about important dimensions to N-dimensional data. For example, suppose we have the data from Figure 5.6.

The given data are two-dimensional data. However, the distribution of the actual data is almost linear. In this case, if the data are transformed in the direction of the widest spreading, the distribution of the data can be described as one dimensional. For example, let us look at the two lines shown in Figure 5.6. These two lines are the best representation of two-dimensional data, which are estimated through the distribution

of the data. We can describe the data in two dimensions, and we can get the best dimension when we represent each data as one dimension with the widest spreading dimension. These two dimensions are called a principal component (PC) and are denoted by PC_1, PC_2 to describe the data well. The PCA is a method of replacing given data with meaningful dimensions in a given dimension in this way. PCA is used to find the PC that best represents the given data to reduce the dimension. PCA is a method of replacing given data with meaningful dimensions in a given dimension in this way. Therefore, finding the right PC is very important. In this case, for the method of obtaining the PC, an eigenvector of the covariance matrix and a corresponding eigenvalue are obtained. The covariance matrix shows the distribution characteristics of a given data. Through each correlation, the correlations between the dimensions can be known and the form of the data can be abstractly known. The formula of the covariance matrix is as follows:

$$\Sigma_{ij} = E[(x_i - \bar{x_i})(x_j - \bar{x_j})] \tag{5.1}$$

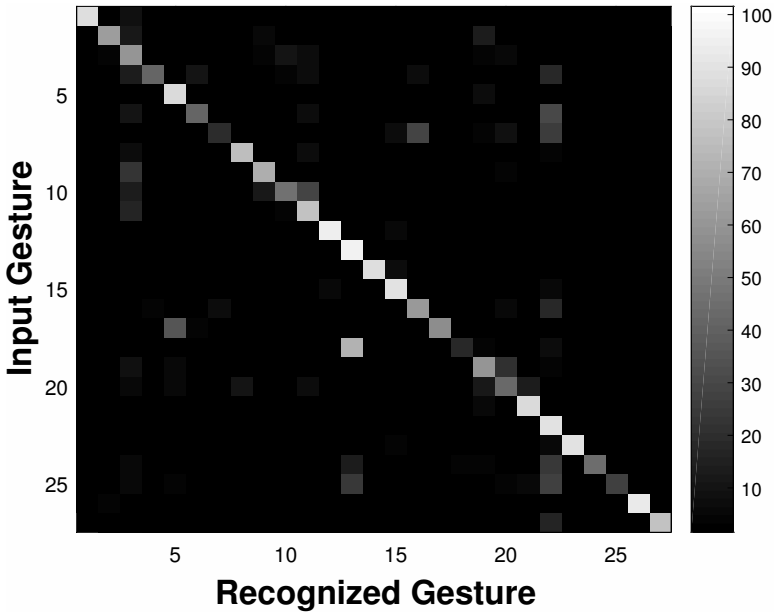
Therefore, the PC can be obtained by analyzing the covariance matrix. If we obtain the eigenvector of the covariance matrix, each eigenvector is a PC that can represent the data, and the corresponding eigenvalue is the amount of information that represents the data. For example, in Figure 5.6, the direction of each line segment is an eigenvector and its size is an eigenvalue. In this case, each eigenvalue can be used to determine the degree to which each PC represents information. This can reduce the dimensionality of the data. The gesture used in this paper has 20 three-dimensional data and has 60-dimensional data in total, although fewer data were actually used. Therefore, the PC of the data was obtained and the eigenvalue corresponding to the PC was analyzed to reduce the number of data. Table 5.4 shows the computed eigenvalue corresponding to the PC of the gesture “swipe left”. The sum of the eigenvalues obtained here was 2.79. For each eigenvalue value, it can be seen that most of the eigenvalues, except

PC	PC_1	PC_2	PC_3	PC_4	PC_5	PC_6	PC_7	PC_8	PC_9	PC_{10}
value	1.39	0.68	0.24	0.18	0.11	0.06	0.03	0.03	0.02	0
PC	PC_{11}	PC_{12}	PC_{13}	PC_{14}	PC_{15}	PC_{16}	PC_{17}	PC_{18}	PC_{19}	PC_{20}
value	0	0	0	0	0	0	0	0	0	0
PC	PC_{21}	PC_{22}	PC_{23}	PC_{24}	PC_{25}	PC_{26}	PC_{27}	PC_{28}	PC_{29}	PC_{30}
value	0	0	0	0	0	0	0	0	0	0
PC	PC_{31}	PC_{32}	PC_{33}	PC_{34}	PC_{35}	PC_{36}	PC_{37}	PC_{38}	PC_{39}	PC_{40}
value	0	0	0	0	0	0	0	0	0	0
PC	PC_{41}	PC_{42}	PC_{43}	PC_{44}	PC_{45}	PC_{46}	PC_{47}	PC_{48}	PC_{49}	PC_{50}
value	0	0	0	0	0	0	0	0	0	0
PC	PC_{51}	PC_{52}	PC_{53}	PC_{54}	PC_{55}	PC_{56}	PC_{57}	PC_{58}	PC_{59}	PC_{60}
value	0	0	0	0	0	0	0	0	0	0

Table 5.4: The PCs of gesture 'Swipe left' and each eigenvalue. Most of eigenvalues are vary small. The data was rounded to the third decimal place.

for nine points, were not even 1%. A PC with a small eigenvalue cannot represent data. Therefore, the eigenvalues are summed until the sum of the eigenvalues of each PC is 99% of the total sum and only the eigenvectors corresponding to the eigenvalues are removed. After recovering the covariance matrix with the selected PC and its corresponding eigenvalue again, we can see that most of the values were restored by calculating the error ratio and the covariance matrix. When PCA was used to reduce the dimension, the recognition rate is shown in Figure 5.7.

If we check the recognition rate, we can see that the result is not higher than the actual idea. The reason for this result is that the actual gesture is related to the x, y, and z axes, respectively. Conventional methods include information about the three axes, such that a negative decision is possible when the selected point should not move on one axis. However, in the case of PCA, it is removed and the recognition rate is decreased.



(a)

Figure 5.7: Result of PCA based dimension reduction.

5.3.4 Variance selection and principal component analysis

In this subsection, we try to summarize the relationship between PCA and the point extraction method using variance. In addition, we obtained the prototype value of each PC and the corresponding PC to identify an important dimension in real data through each PC. To obtain the corresponding value of each term, we used the following equation

$$D_i = \sum_{n=1}^{N} eigenvalue_n |PC_n(i)| \tag{5.2}$$

where $PC_n(i)$ is the value corresponding to the i th axis in PC_n and N is the number of all selected PCs. The reason for using the absolute value here is that the eigenvector is directional and has negative and positive numbers; however, the absolute value has the same direction as the eigenvector. It can be seen from the obtained values that some terms had a large value, whereas others had a small value depending on the gesture. If

we extract only large values here, they will have the same shape as the important points extracted using the variance. Actual PCA looks at the covariance of the entire data and gives a high value to the important axis in each PC. Therefore, the same result can be obtained by using the covariance.

5.4 Gesture Normalization and Fitting

5.4.1 Zero starting

Next, a study was made to match the starting point of each gesture with the distance between each gesture joint. The simplest way to fit a gesture is to ignore the relationship between each joint and to set the starting point of all the points to zero. By setting the starting point of each gesture to zero, we eliminate the influence of the starting point disparity. The recognition rate of the gesture that matched the starting point is shown in Figure 5.8 (a).

Figure 5.8 (a) shows the recognition rate of each gesture. The result was 44.7%, which is actually a very low recognition rate. The reason for the lower recognition rate can be seen in Figure 5.9 (b). The figure shows that each gesture has a similar shape overall. However, the difference in the arms' length and in the size of the movement was different for each person, and, therefore, the operation range of the gesture was changed and actually deviated greatly. In this case, even though the whole form looks similar, it failed to be recognized because it does not show the same gesture.

5.4.2 Zero to ten fitting

In the previous subsection, we changed the starting point to 0 for the fitting of the gesture; however, the size of the gesture to be taken for each person was different, which caused the recognition to fail. Therefore, to normalize the size, we obtained the

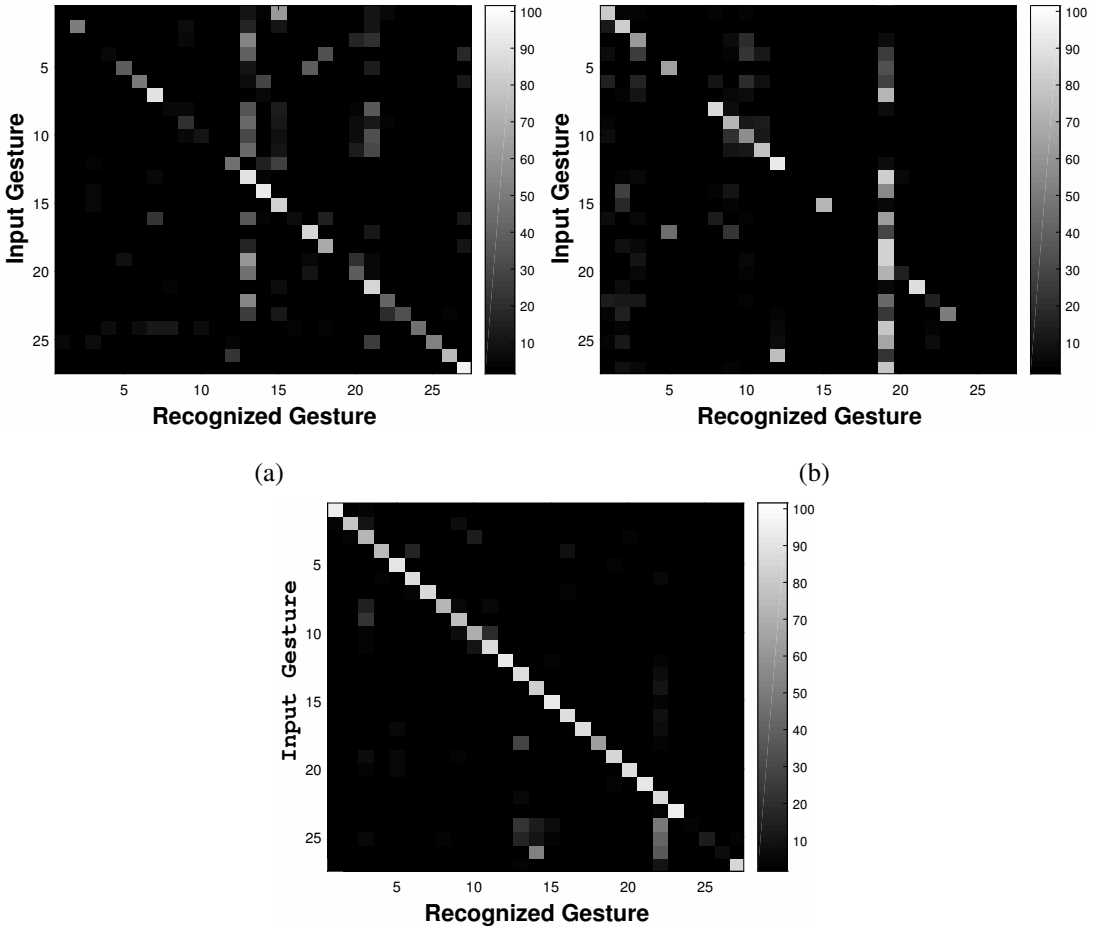


Figure 5.8: Gesture recognition result of each fitting method. (a) Zero starting. (b) Zero to ten fitting. (c) Rate fitting.

x, y, and z axis minimum and maximum values of the entire gesture and fitted them from 0 to 10. A fitting gesture can fit the entire gesture, reducing the effect of length variation or gesture size. The recognized result is shown in Figure 5.8 (b). Gestures fitted from 0 to 10 showed low recognition rate. This reduced the effect of distance by fitting the entire gesture; however, it also distorted the shape of each gesture. For example, Figure 5.9 (a) shows the gesture before fitting, whereas Figure (c) shows the fitting gesture. The data show different shapes before the fitting; however, the fitting shapes the whole shape from 0 to 10, such that the other gestures were drawn in the same shape. Therefore, confusion occurred between two gestures, and the recognition rate decreased.

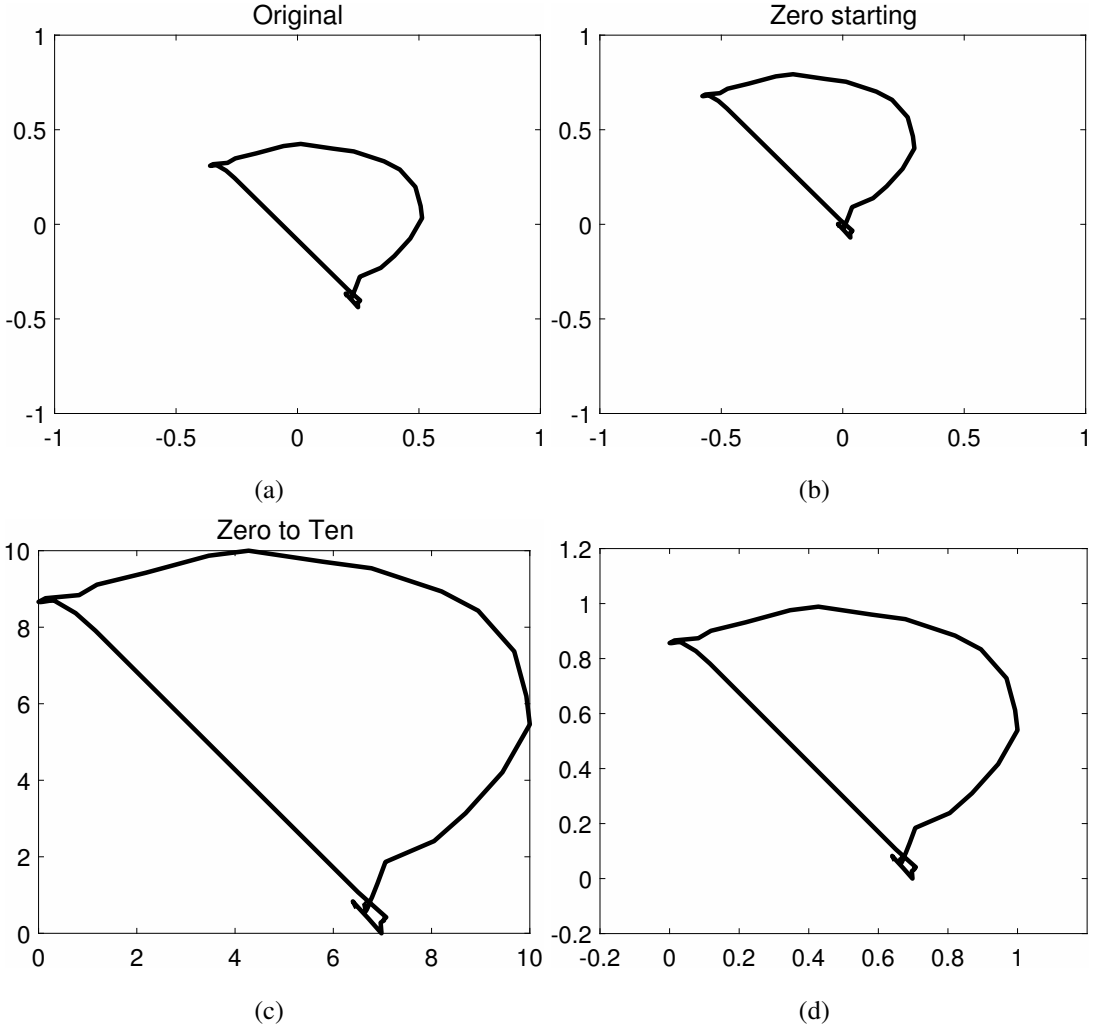


Figure 5.9: Each fitting Methods. (a) Original signal. (b) Zero starting signal. (c) Zero to ten fitting. (d) Rate fitting

5.4.3 Rate fitting

A gesture can be sized from 0 to 10 fittings; however, other shapes can come in a similar shape. Therefore, to maintain the overall shape, we set the largest axis among the three axes, x, y, and z, from 0 to 1, and we fitted the gesture according to the ratio. Then, we obtain the following equation:

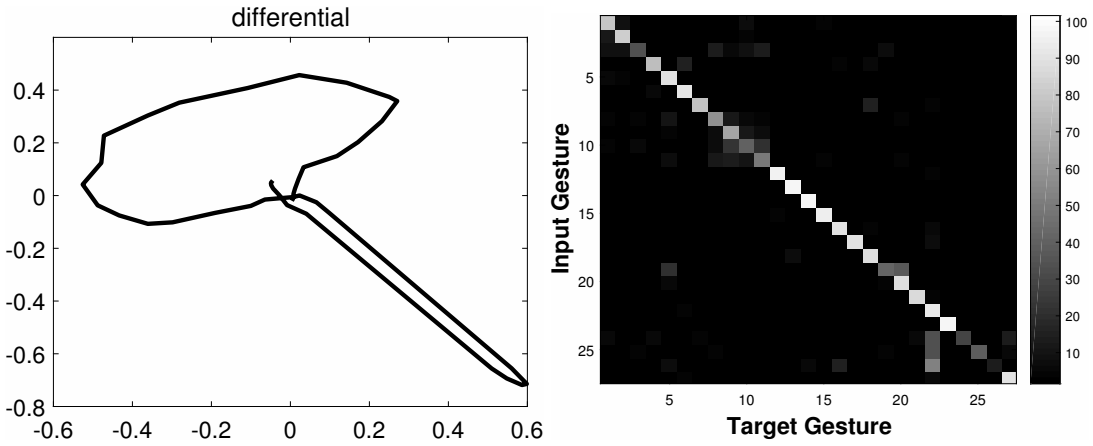
$$\begin{aligned} m_g &= max_d(sig_d^g) \\ \bar{sig} &= \frac{sig}{m_g} \end{aligned} \tag{5.3}$$

where sig_d is the dth-dimensional data of gesture "g". The gesture thus obtained can maintain the shape and improve the recognition rate of the gesture. The result of the recognition is shown in Figure 5.8 (c) whereas the fitting gesture is shown Figure 5.9 (d). As a result, it can be confirmed that the overall recognition rate was 75.6%, which was higher than that obtained by other fitting methods. The rate coding raised the recognition rate by eliminating the length problems caused by other methods and the distortion of the overall rate. However, it still has a $75\% \sim 80\%$ recognition rate. The reason for this is the removal of the stationary point. Non-motivated points can conduct a negative decision to different gestures. However, the movement of the dots that are not moving as a result of the fitting-up process is enlarged, and, accordingly, the dots that have small movements should have unnecessary information. Thus, if a gesture is included in another gesture, the recognition rate will be reduced. For example, the gestures "Sit to stand", "Stand to sit", and "Lunge" include the action of bending the knees as in the gesture "Jog". As a result, the rate of perception decreases, as it is recognized as a matrix, known to better follow the form of the entire gesture, thus reducing the recognition rate.

5.4.4 Differential

Next, differentials were used to determine the recognition rate of the relative motion of each skeleton point. If the gesture is recognized through the differentials, it can be recognized through the direction of movement without needing to initialize the position of the gesture. After differentiating each gesture, fitting was performed to normalize the size. The differential gesture is shown in figure 5.10 (a) and the recognition result is shown in (b).

The overall recognition rate was 72.96%. If we show the recognized results, we can see that the result is similar to the previous fitting method. The reason for this result is that the shape of the whole gesture is matched, but as in the previous methods, there is



(a) (b)

Figure 5.10: Diifferential data and its result. (a) Differential gesture. (b) Result.

a limitation in the fitting to the size, and the recognition fails.

5.5 Recognition Using a Neural Network

The previous selection showed problems with gesture recognition using a skeleton. The skeleton changes every time depending on the position of the person, the length of the joint, and the size of the operation. In particular, if the size of the gestures and the starting points are not matched, the recognition rate significantly decreases. We can also see that different gestures behave differently, reducing the recognition rate. As a way to complement this, we chose to have a large amount of information in this thesis and matched the starting point with the moving size. However, this method ignores the information about stationary points. For instance, the gesture “wave” is used to maximize the information in the left hand by moving it over a continuous left arm. Therefore, only the dots on the actual left hand are selected and the rest are ignored. However, it can serve as a piece of information that indicates that there is no movement in a gesture. For instance, in the case of the gesture “Walk”, there is movement of the legs, but there is no “Wave”. Therefore, if there is movement of the legs, it is not possible to determine the formation of the “Wave”. A negative decision

	Image Type	Dependency	Number of Layer
Network 1	Binary	X	1
Network 2	Binary	X	2
Network 3	Binary	O	1
Network 4	Probability Gray	X	1

Table 5.5: The layer of the CNN.

can be done through a stationary point. However, if the fitting used in this paper is adjusted, the points that are stationary may also grow. Consequently, it is difficult to apply. We applied a multilayer perceptron (MLP) and a convolutional neural network (CNN) to modify it. Each method is influenced slightly by the location or size of the image. For this purpose, the gestures of each dot were changed to a 20×20 binary image. The network went through two ways of combining the results after recognizing each joint and adding dependencies between the points. In another experiment, when constructing an image, we created gray image of the probability that a point exists at each position, and then conducted training and testing. The equation for constructing the image is as follows.

$$image'(i,j) = s^{p}_{(i,j)} * image(i,j) \quad s^{p}_{(i,j)} \text{ is the probability that point exist at pixel } (i,j) \text{ in joint } p. \tag{5.4}$$

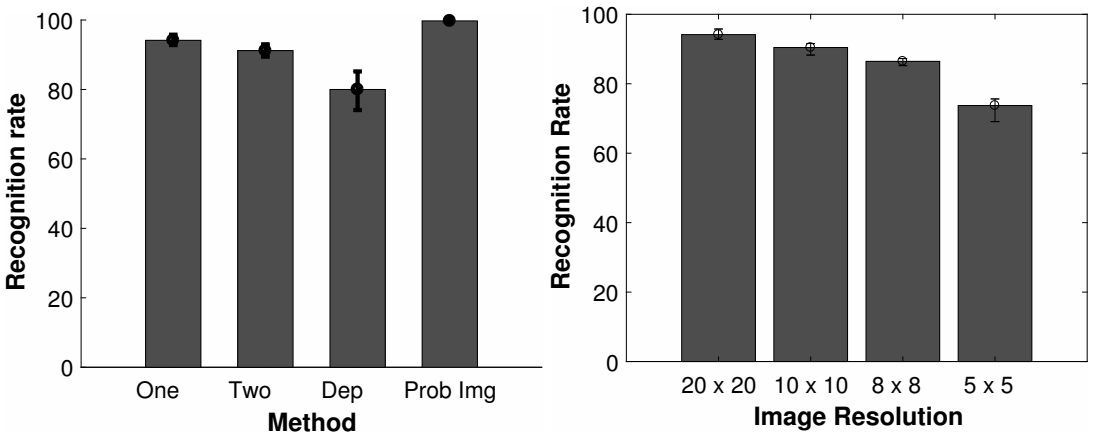
The configuration of the Network is shown in Table 5.5.

Recognition proceeded by constructing four networks.

5.5.1 Convolutional neural network

First, we made this perception using a CNN. The recognition results are presented in Figure 5.11 (a).

If we check the recognition rate, we can see that there is a higher recognition rate when



(a) (b)

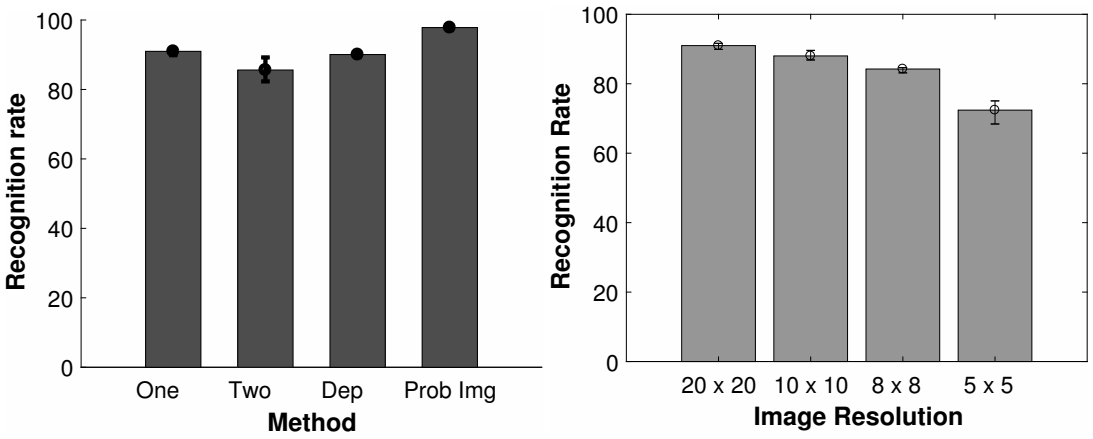
Figure 5.11: Gesture recognition rate using CNN (a) Recognition result. (b)Recognition result according to image resolution.

there is one layer. In addition, recognition rate decreases when adding dependency to Networks, and recognition rate increases when using probability gray image. Next, the perception was checked when the image was reduced from 20×20 to 10×10, 8×8, and 5×5. Only layers were applied to confirm the results. The recognition rate is indicated in Figure 5.11 (b). When we check our perception, we can see that our perception decreased as the resolution of the image decreased.

5.5.2 Multilayer perceptron

Similarly, we developed our perception using MLP. At this point, when the image size used as input was n×n, it was changed to an image size of (n×n) for application as an input to the perceptron. The recognition rate is shown in Figure 5.12.

If we check the recognition rate, like in the CNN, we see higher recognition rates when only layers are applied and lower recognition rates when the resolution is reduced.



(a) (b)

Figure 5.12: Gesture recognition rate using perceptron (a) Recognition result. (b)Recognition result according to image resolution. As image resolution decrease recognition rate be lower.

5.5.3 Neural Network Result

We can see two things by checking the recognition using MLP and CNN. First, we found that, as the number of layers increased, the recognition rate decreased. This is an overfitting, a problem with neural networks, where a simple image of 20×20 is applied to two layers, making it less common to model a transport set. Therefore, if the number of layers is small, the recognition rate is higher. Also, the recognition rate decreased when the image dependency was added. The reason for this result is that the complexity increases when the whole images are combined into one, and the dimension of the network increases. As a result, the distribution of data becomes complicated and the probability of falling to the local minimum increases, failing to reach the global minimum. Therefore, recognition rate is greatly shaken and many gestures are not recognized. Next, the results of the probability gray image are shown. When the probability gray image is applied, the recognition rate is increased. By applying this result to the probabilistic gray image, it is possible to fix the gesture 's operation area to small region, and to improve the recognition rate by removing the motion in the area where no motion occurs. We can also see that the perception decreased as the resolution of

the image decreased. It gives a precise description of the form of the gesture, leading to a higher recognition rate. The one-layer CNN had the highest recognition rate of the overall result, and most of the neural network methods had a higher recognition rate than that of the PFSM. From this, we can verify that perception using the neural network is possible without being affected by the location or size. This is a higher recognition rate compared to that of the PDTW or PFSM as it does not require the sizing of actual gestures. On the other hand, this method takes longer to train compared to the two methods.

5.6 Conclusion

In this chapter, we studied how to select or fit gesture points to improve the recognition rate of gestures. The three-dimensional skeleton gesture has a large number of points, and the number of points having a large amount of information per gesture is different. Moreover, the same gesture can be perceived differently depending on the person performing it. In this paper, we applied variance, speed, and PCA to select the point where the gesture moves considerably. To adjust the size of the gesture, we moved the position or changed the ratio of the moving distance. The best result was achieved when the gesture was recognized according to the original ratio, showing a recognition rate of approximately 75.6%. Moreover, the greater the points that can be selected flexibly, the better the recognition rate is. Each gesture has a different degree of movement, and the number of moving points is different. We also conducted recognition using various Neural Networks. Recognition using the Neural Network showed high recognition rates without being affected by problems with gesture recognition.

Problem	Method	Recognition Rate
Choosing	Variance	73.17
	Speed	75.58
	PCA	67.73
Fitting	Zero Starting	44.73
	Zero to Ten Fitting	37.22
	Rate Fitting	75.58
Both	Differential	72.96
	CNN(1 Layer)	94.16
	CNN(2 Layer)	91.20
	MLP(1 Layer)	90.99
	MLP(2 Layer)	85.60
	CNN(Binary image)	97.82
	MLP(Binary image)	94.16

Table 5.6: The recognition rate of each method. We fix the starting point and the end point by three conditions, suggest a step size that moves at once, and make the matching point not to be backward, and to be able to match the whole signal.

5.7 Summary

We conducted a study on two problems with gesture recognition. Gestures have found that the starting position and the range of motion vary significantly depending on the person's height, shooting environment, and size of motion. For the gesture recognition, we chose to have a large amount of fittings and information. We confirmed that the recognition rate varies greatly depending on the various fitting methods and on the methods of selecting points. For the selection of a gesture that has a large amount of information, it was evaluated with cumulative speed and distribution. It also enabled flexible choices based on point choices and gestures based on fixed travel distances. Several experiments were performed in practice, and we found that the better the selection criteria, the better the perception will be. We also conducted a study on the fitting method of gestures. EEach gesture had a different beginning and a different end

point. To recognize the gesture, we conducted a fitting in various ways. We found that the rate of perception varied significantly with the position and size settings of the resulting gesture. Lastly, we conducted gesture recognition using a neural network. We conducted a perception using CNN and MLP and obtained a high recognition rate. Both methods can automatically modify the position of the gesture, making it more likely to be recognized compared to the methods proposed in this thesis. The solutions applied in this chapter and the corresponding recognition rate are shown in Table 5.6. Thus, the importance of selection of fitting and information for gesture recognition was determined.

Chapter 6

Conclusions

Gesture recognition has been studied and applied to HCI and HRI. People can communicate their intentions by a gesture and can judge status or situations through unconscious behaviors. In recent years, research based on deep learning has become popular and machine learning recognition rates have improved, and deep learning is expected to be applied to various fields involving future science. We studied algorithms to develop FSM and DTW to recognize gestures, and recognized a gesture using various fittings and dimension reductions. The existing FSM-based method considers the fact that only one PATH, which is a disadvantage of the existing method, is remembered. . We also apply probability to DTW to obtain the variance from the center of the gesture and to adjust the distance according to a distribution. By studying gesture fitting and dimension reduction, we improved the recognition rate by controlling unnecessary information and solved the problem according to size.

6.1 State Transition Probability-Based Finite State Machine

In Chapter 3, we proceeded with research on how to develop FSM. To develop the FSM, we applied forward and backward algorithms, which are used to modify gestures

in HMM. In addition, we recognize the gesture as a path classification using a transition matrix to supplement the FSM which can recognize a gesture easily but cannot recognize a gesture with various paths. When recognizing a gesture based on probabilities, various combinations were possible by distinguishing paths from gestures' states, and it was possible to show a high recognition rate even with sparse training data. Furthermore, as the proposed PFSM can recognize a gesture when training data is sparse, it is confirmed that the recognition rate can be improved by more sophisticated recognition when the number of states increases. While there are advantages over existing methods, there is a disadvantage in that it is impossible to distinguish between repeated gestures. Standard FSM memorizes the entire path, and it is possible to classify the number of repeated gestures in a certain interval, but this is not possible with the proposed method. The recognition rate of a gesture was up to 85% according to the number of STATEs, which was much higher than the recognition rate of HMM provided by the data set (e.g.,68%). Furthermore, PFSM exhibits faster computation than other methods.

6.2 Probability-Based Dynamic Time Warping for Gesture Recognition and Signal Warping

In Chapter 4, we developed a recognizer that can consider the distribution of gestures by combining DTW with a probability distribution. To generate the probability distribution, a representative path was determined in various ways. The representative path is a time mean representation that finds the center point by matching the number of samples according to the execution time of the gesture, a length mean representation that divides the interval of the gesture according to the length, or a repeated warping representation obtained by repeating DTW. Recognition rate is considered in the time axis, the center of the length mean representation, and repeated warping representation achieves high recognition rates. Recognition rates were 76%, 91% and 91%, respec-

tively. However, the operation speed is slower than standard HMM or DTW.

6.3 Multi-Point Gesture Recognition

Finally, we conducted research to solve fitting and dimensional problems in gesture recognition. A non-refined gesture is not well recognized because it differs greatly in various points, such as starting point and size. Thus, fitting was conducted in various ways and features were changed and tested. We also studied how to reduce the dimensions involved in multipoint gesture recognition. Actual skeleton gestures represent whole body gestures, but not all gestures use all of the body. When there is a non-moving point, a rank problem may occur in the probability, and the information unevenness problem arises. Therefore, dimensions were reduced and recognition was evaluated. PCA was used to reduce the dimensions, and the amount of information was determined by the distribution of gesture points or the accumulated speed. Among the various experiments, the highest recognition rate was obtained when the maximum distance was recognized as 1 to maintain the shape, and the gesture was selected based on the accumulation of the speed. However, most methods did not obtain good recognition rates and even the highest recognition rate showed loss of information. The recognition rate of the methods proposed in this paper and the results of previous gesture recognition using UTD-MHAD are shown in Table 6.1.

6.4 Future Work

6.4.1 Algorithm and feature improvement

As a method in future research, we will further study the angle and differential data of the skeleton. Angle-based recognition can focus on the relationships between joints, so that more information can be applied than in the method of recognizing each point

Paper	Method	Recognition Rate
Chen et al. (2015)	DMM, HMM	66.1
Chen et al. (2016)	DMM, CRC	74.7
Zhou et al. (2014)	ELC-KSVD	76.19
Zhang et al. (2017)	3DHoTs, MBC	84.4
Hou et al. (2016)	CNN	86.97
Ours	HMM(Left-Right)	65.19
Ours	HMM(Ergodic)	67.41
Ours	PFSM	75.58
Ours	PDTW (Length mean)	90.74
Ours	PDTW (Repeated warping)	90.79
Ours	CNN	94.16
Ours	MLP	90.99
Ours	CNN (with Dependency)	80.00
Ours	MLP (with Dependency)	90.05
Ours	CNN (Probability image)	94.16
Ours	MLP (Probability image)	97.82

Table 6.1: The compare of recognition rate with other algorithms.

separately. It is also expected that using derivatives will reduce the effect of the starting point. To improve the algorithm, we will focus on supplementing the shortcomings of the current algorithm. Although PDTW has a high recognition rate, it requires extensive computations and long processing time, and PFSM has a gesture with a particularly low recognition rate. If these shortcomings are complemented, the algorithms should be applicable to a wider range of problems.

6.4.2 Extend to HCI applications

We will also study the application of the proposed algorithm. We present the current algorithm, but we intend to develop auxiliary equipment that can analyze a user's gesture first, and develop a robot to analyze necessary operations and provide corresponding

services. A gesture contains information about the motion involved. If the robot recognizes the gesture first and grasps the intention of a gesture, it is expected that the robot can train itself without direct manipulation by the user. Moreover, it is expected that a similar technique can be applied to the development of auxiliary equipment that helps a user to perform manual labor using less power through prediction of the operation to be carried out by recognizing user gestures.

Appendix A

UTD-MHAD

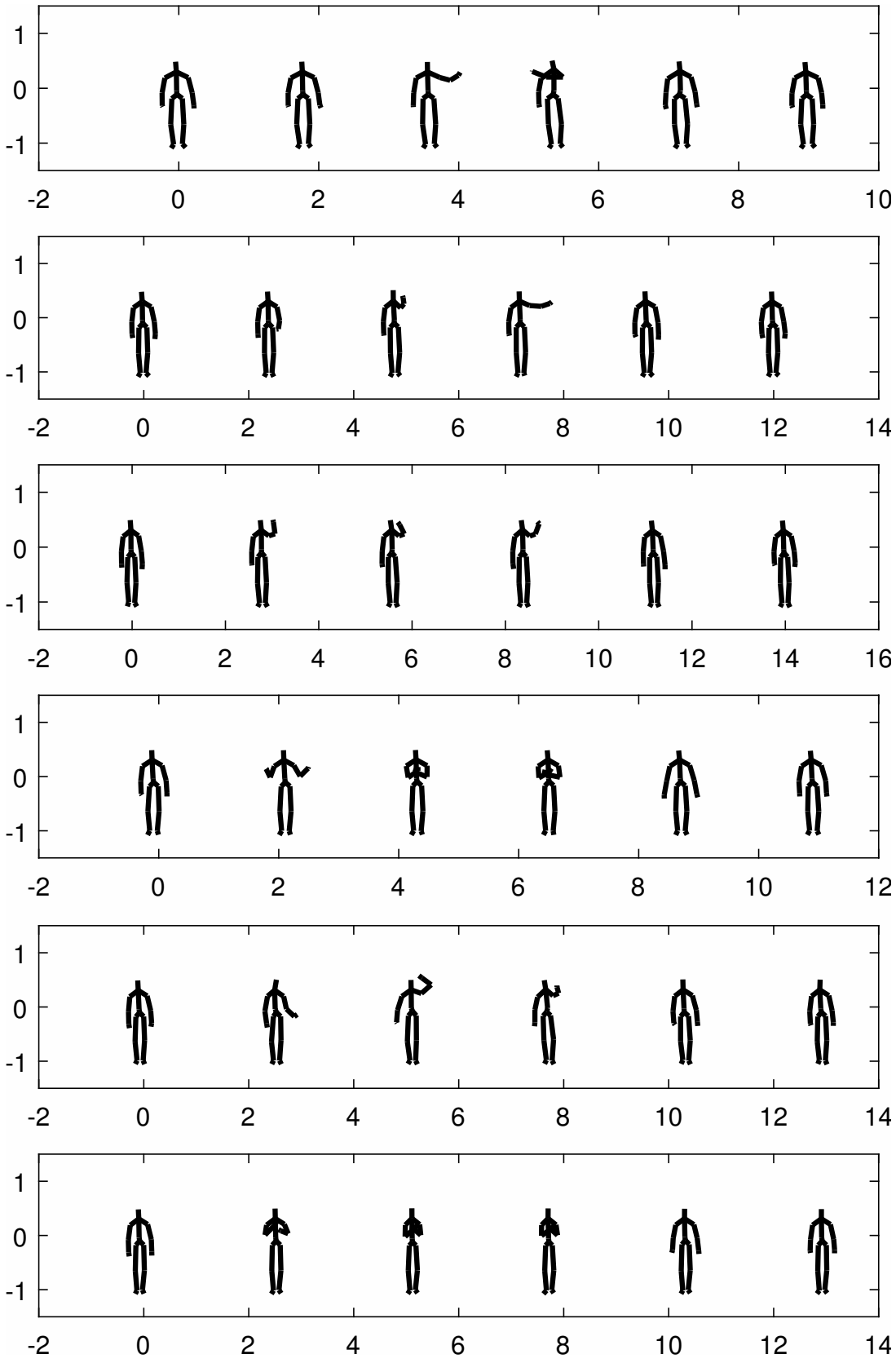


Figure A.1: Gesture of UTD-MHAD which index 1 to 6. The higher up in the figure, the lower the index is.

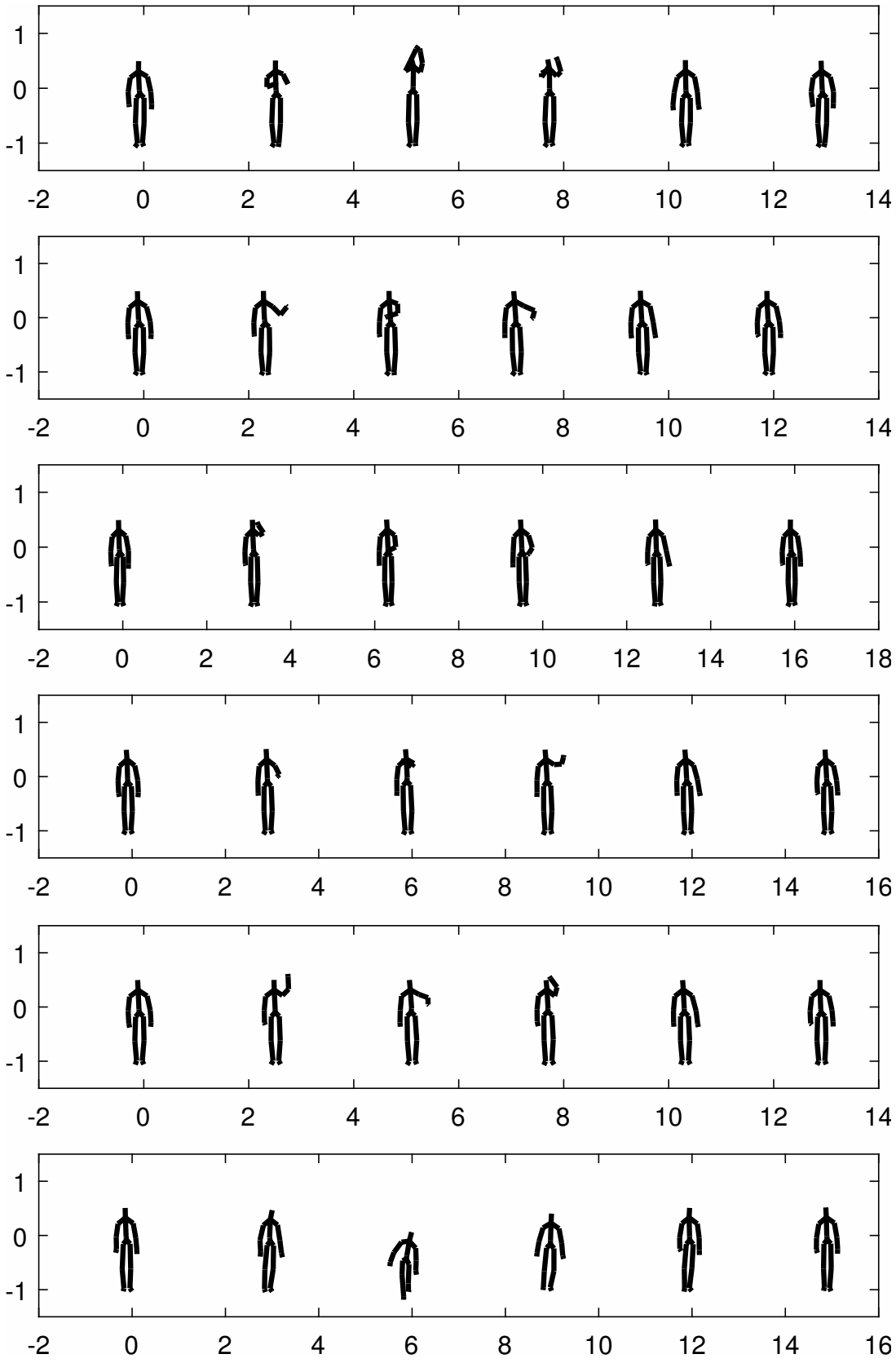


Figure A.2: Gesture of UTD-MHAD which index 7 to 12. The higher up in the figure, the lower the index is.

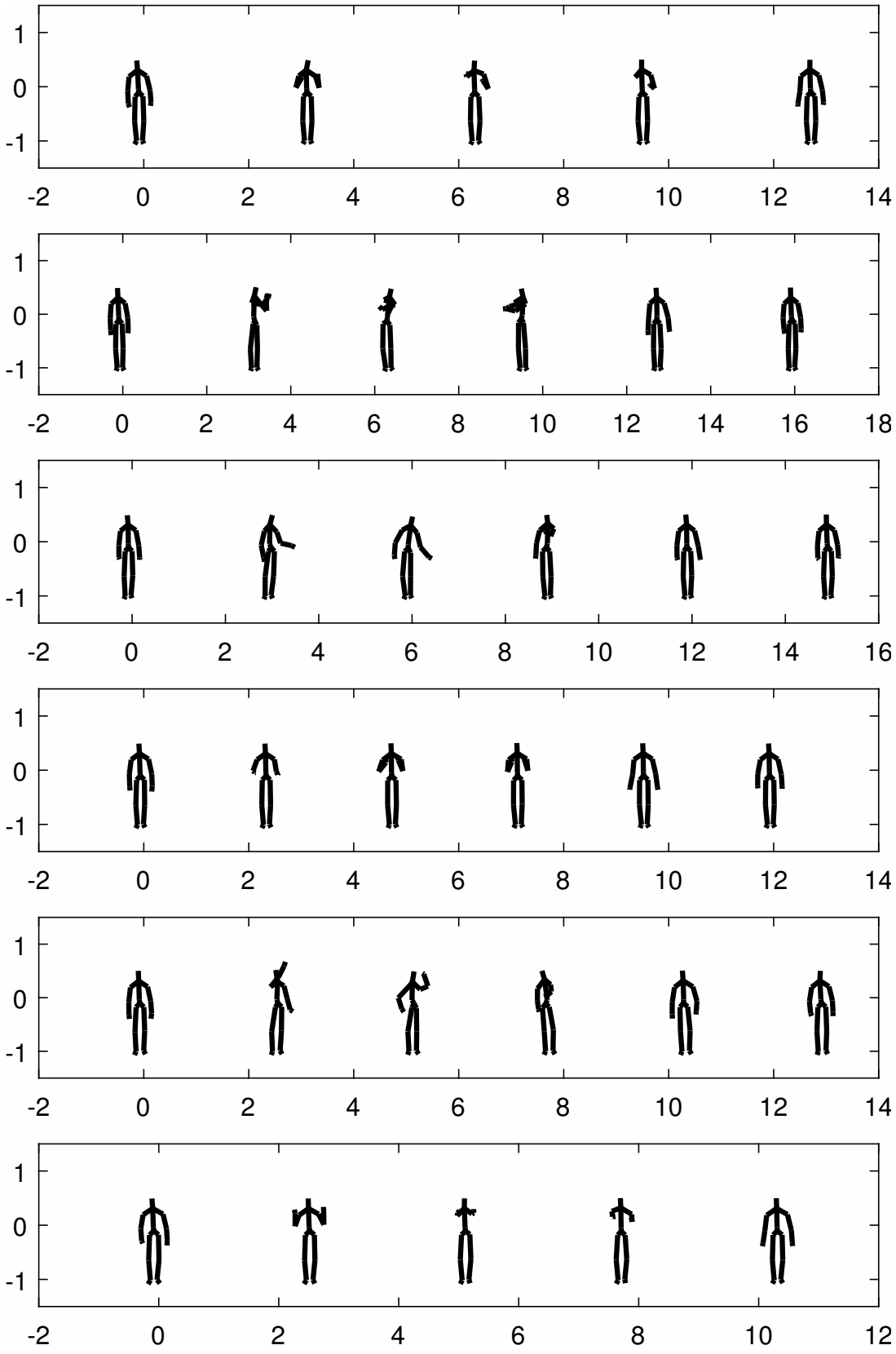


Figure A.3: Gesture of UTD-MHAD which index 13 to 18. The higher up in the figure, the lower the index is.

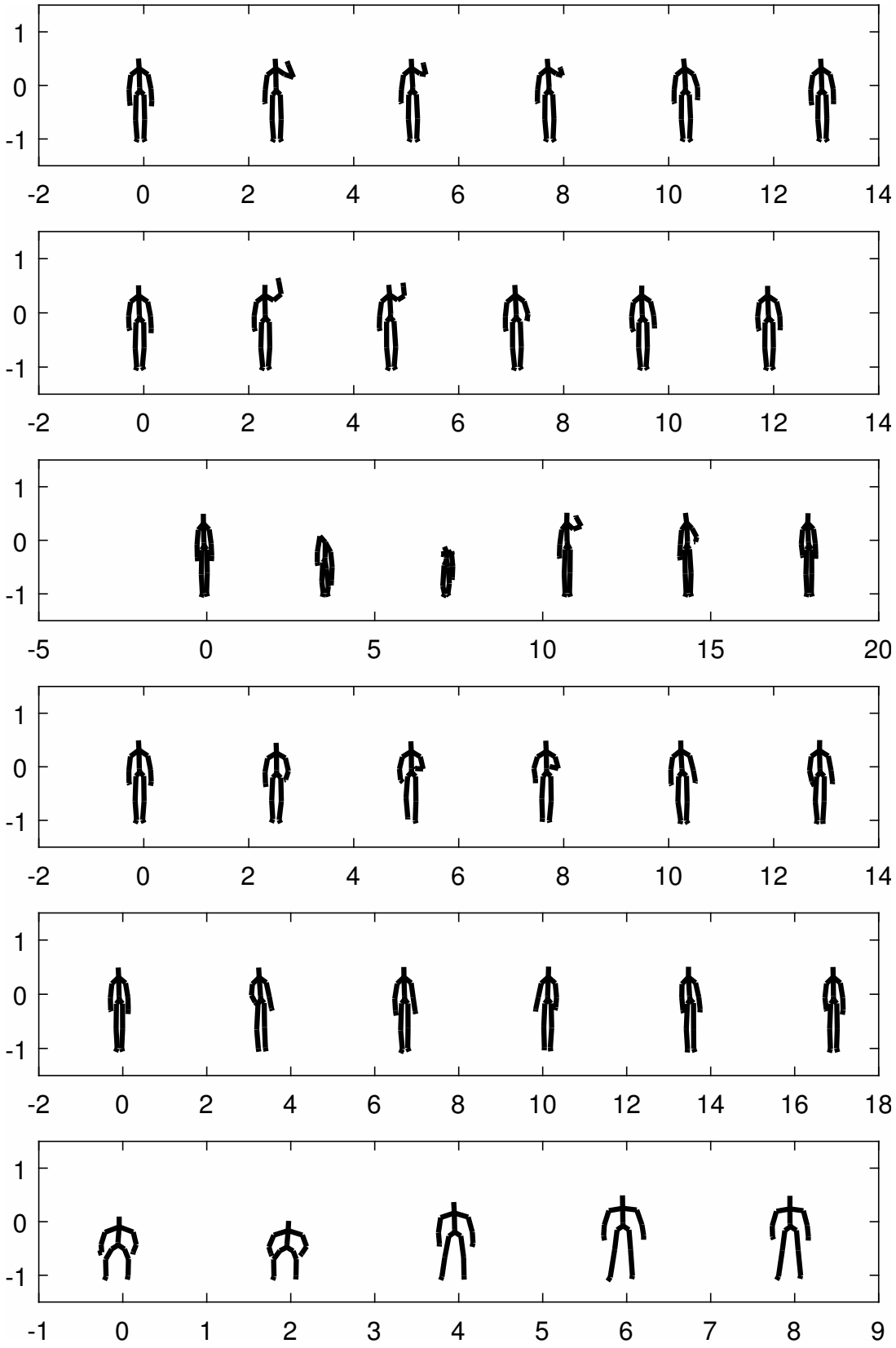


Figure A.4: Gesture of UTD-MHAD which index 19 to 24. The higher up in the figure, the lower the index is.

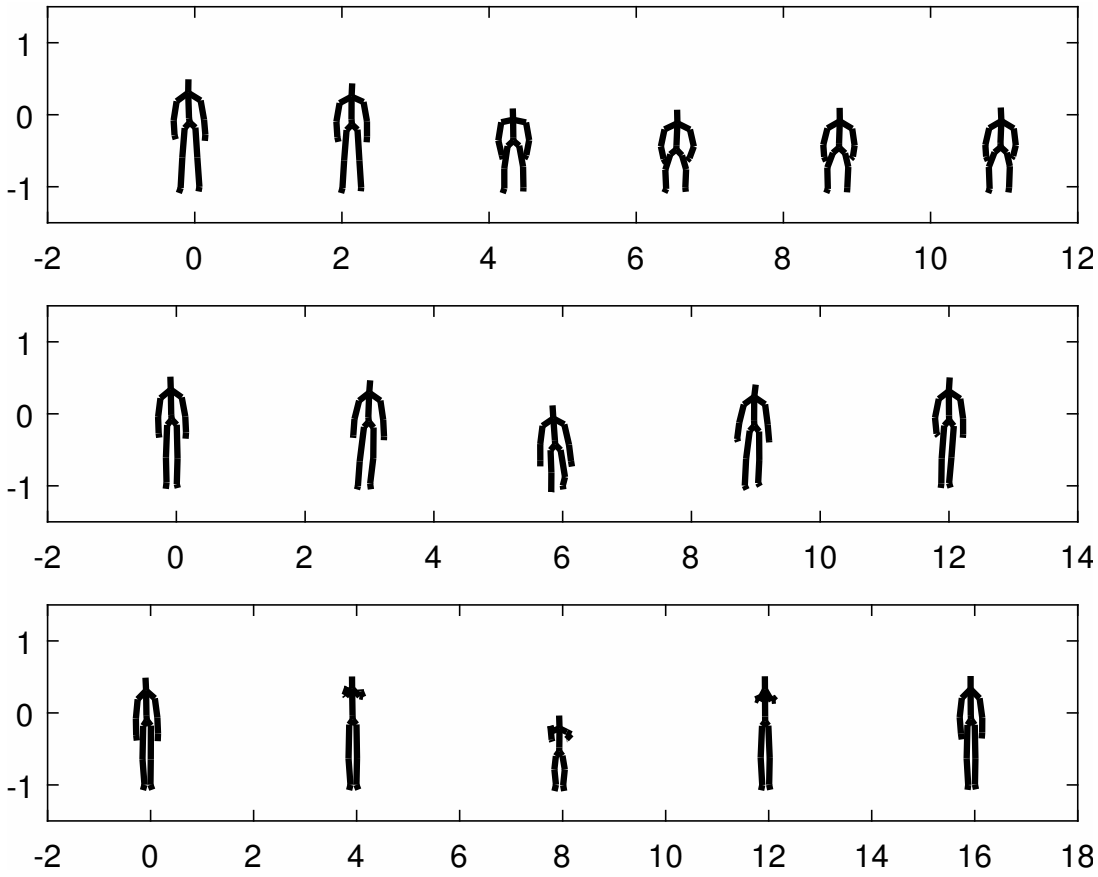


Figure A.5: Gesture of UTD-MHAD which index 25 to 27. The higher up in the figure, the lower the index is.

Bibliography

Acero, A., Deng, L., Kristjansson, T. T., and Zhang, J. (2000). HMM adaptation using vector taylor series for noisy speech recognition. In *INTERSPEECH*, pages 869–872.

Alsharif, O., Ouyang, T., Beaufays, F., Zhai, S., Breuel, T., and Schalkwyk, J. (2015). Long short term memory neural network for keyboard gesture decoding. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 2076–2080. IEEE.

Barczak, A. L. and Dadgostar, F. (2005). Real-time hand tracking using a set of cooperative classifiers based on haar-like features.

Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA.

Bhuyan, M., Ghosh, D., and Bora, P. (2005). Threshold finite state machine for vision based gesture recognition. In *INDICON, 2005 Annual IEEE*, pages 379–382. IEEE.

Chang, J. Y. (2014). Nonparametric gesture labeling from multi-modal data. In *ECCV Workshops (1)*, pages 503–517.

Charniak, E. (1996). *Statistical language learning*. MIT press.

Chen, B., Hua, C., Han, J., and He, Y. (2017). A novel real-time gesture recognition

algorithm for human-robot interaction on the uav. In *International Conference on Computer Vision Systems*, pages 518–526. Springer.

Chen, C., Jafari, R., and Kehtarnavaz, N. (2015). Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 168–172. IEEE.

Chen, C., Jafari, R., and Kehtarnavaz, N. (2016). A real-time human action recognition system using depth and inertial sensor fusion. *IEEE Sensors Journal*, 16(3):773–781.

Chen, F.-S., Fu, C.-M., and Huang, C.-L. (2003). Hand gesture recognition using a real-time tracking method and hidden markov models. *Image and vision computing*, 21(8):745–758.

Chen, Q., Georganas, N. D., and Petriu, E. M. (2007). Real-time vision-based hand gesture recognition using haar-like features. In *Instrumentation and Measurement Technology Conference Proceedings, 2007. IMTC 2007. IEEE*, pages 1–6. IEEE.

Chuan, C.-H., Regina, E., and Guardino, C. (2014). American sign language recognition using leap motion sensor. In *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*, pages 541–544. IEEE.

Davis, J. and Shah, M. (1994). Visual gesture recognition. *IEE Proceedings-Vision, Image and Signal Processing*, 141(2):101–106.

Demuth, H. B., Beale, M. H., De Jess, O., and Hagan, M. T. (2014). *Neural network design*. Martin Hagan.

Elons, A., Ahmed, M., Shedid, H., and Tolba, M. (2014). Arabic sign language recognition using leap motion sensor. In *Computer Engineering & Systems (ICCES), 2014 9th International Conference on*, pages 368–373. IEEE.

Escalera, S., Athitsos, V., and Guyon, I. (2017). Challenges in multi-modal gesture recognition. In *Gesture Recognition*, pages 1–60. Springer.

Federgruen, A. and Tzur, M. (1991). A simple forward algorithm to solve general dynamic lot sizing models with n periods in 0 (n log n) or 0 (n) time. *Management Science*, 37(8):909–925.

Forney, G. D. (1973). The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Freeman, W. T. and Roth, M. (1995). Orientation histograms for hand gesture recognition. In *International workshop on automatic face and gesture recogn ition*, volume 12, pages 296–301.

Funasaka, M., Ishikawa, Y., Takata, M., and Joe, K. (2015). Sign language recognition using leap motion controller. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, page 263. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Gales, M. J. (1998). Maximum likelihood linear transformations for HMM-based speech recognition. *Computer speech & language*, 12(2):75–98.

Gerling, K., Livingston, I., Nacke, L., and Mandryk, R. (2012). Full-body motion-based game interaction for older adults. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1873–1882. ACM.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. (1995). *Markov chain Monte Carlo in practice*. CRC press.

Hong, P., Turk, M., and Huang, T. S. (2000a). Constructing finite state machines for fast gesture recognition. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 3, pages 691–694. IEEE.

Hong, P., Turk, M., and Huang, T. S. (2000b). Gesture modeling and recognition

using finite state machines. In *Automatic face and gesture recognition, 2000. proceedings. fourth ieee international conference on*, pages 410–415. IEEE.

Hou, Y., Li, Z., Wang, P., and Li, W. (2016). Skeleton optical spectra based action recognition using convolutional neural networks. *IEEE Transactions on Circuits and Systems for Video Technology*.

Keogh, E. (2002). Exact indexing of dynamic time warping. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 406–417. VLDB Endowment.

Keogh, E. J. and Pazzani, M. J. (2001). Derivative dynamic time warping. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pages 1–11. SIAM.

Kollorz, E., Penne, J., Hornegger, J., and Barke, A. (2008). Gesture recognition with a time-of-flight camera. *International Journal of Intelligent Systems Technologies and Applications*, 5(3-4):334–343.

Kwon, H. and Kim, D. Gesture recognize using regenerated paths and multipath FSM machine. *Sensors and actuator*.

Kwon, H. and Kim, D. Multipoint skeleton gesture recognition and preprocessing method. *Sensors and actuator*.

Kwon, H. and Kim, D. Probability based dynamic time warping for gesture recognition and signal warping. *Sensors and actuator*.

Kwon, H. and Kim, D. 상태 천이 확률을 이용한 제스처 인식 알고리즘. *제어로봇 시스템 공학회*.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pages 396–404.

Lee, H.-F., Lu, Y.-S., Huang, J.-L., and Hu, C.-L. (2015). A skeleton and gesture based user authentication system. In *Ubi-Media Computing (UMEDIA), 2015 8th International Conference on*, pages 254–258. 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, K.-F. and Hon, H.-W. (1989). Speaker-independent phone recognition using hidden markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(11):1641–1648.

Liang, R.-H. and Ouhyoung, M. (1995). A real-time continuous alphabetic sign language to speech conversion vr system. In *Computer Graphics Forum*, volume 14, pages 67–76. Wiley Online Library.

Liang, R.-H. and Ouhyoung, M. (1998). A real-time continuous gesture recognition system for sign language. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 558–567. IEEE.

Lim, G. H., Pedrosa, E., Amaral, F., Lau, N., Pereira, A., Dias, P., Azevedo, J. L., Cunha, B., and Reis, L. P. (2017). Rich and robust human-robot interaction on gesture recognition for assembly tasks. In *Autonomous Robot Systems and Competitions (ICARSC), 2017 IEEE International Conference on*, pages 159–164. IEEE.

Lissier, E. (1995). Markov models and hidden markov models: A brief tutorial. *Introduction to Artificial Intelligence at the University of California, Berkeley*.

Liu, Y., Dong, M., Bi, S., Gao, D., Jing, Y., and Li, L. (2016). Gesture recognition based on kinect. In *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2016 IEEE International Conference on*, pages 343–347. IEEE.

McCormick, J., Vincs, K., Nahavandi, S., Creighton, D., and Hutchison, S. (2014). Teaching a digital performing agent: Artificial neural network and hidden markov model for recognising and performing dance movement. In *Proceedings of the 2014 International Workshop on Movement and Computing*, page 70. ACM.

Mohandes, M., Deriche, M., and Liu, J. (2014). Image-based and sensor-based approaches to arabic sign language recognition. *IEEE transactions on human-machine systems*, 44(4):551–557.

Monnier, C., German, S., and Ost, A. (2014). A multi-scale boosted detector for efficient and robust gesture recognition. In *ECCV Workshops (1)*, pages 491–502.

Motion, L. (2015). Leap motion controller. *URl: https://www. leapmotion. com*.

Neverova, N., Wolf, C., Taylor, G., and Nebout, F. (2016). Moddrop: adaptive multi-modal gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(8):1692–1706.

Ohn-Bar, E. and Trivedi, M. M. (2014). Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE transactions on intelligent transportation systems*, 15(6):2368–2377.

Palma, C., Salazar, A., and Vargas, F. (2016). HMM and DTW for evaluation of therapeutical gestures using kinect. *arXiv preprint arXiv:1602.03742*.

Peng, X., Wang, L., Cai, Z., and Qiao, Y. (2014). Action and gesture temporal spotting with super vector representation. In *ECCV Workshops (1)*, pages 518–527.

Pitsikalis, V., Katsamanis, A., Theodorakis, S., and Maragos, P. (2017). Multimodal gesture recognition via multiple hypotheses rescoring. In *Gesture Recognition*, pages 467–496. Springer.

Potter, L. E., Araullo, J., and Carter, L. (2013). The leap motion controller: a view on sign language. In *Proceedings of the 25th Australian computer-human inter-*

action conference: augmentation, application, innovation, collaboration, pages 175–178. ACM.

Rabiner, L. and Juang, B. (1986). An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Rose, R. C. and Paul, D. B. (1990). A hidden markov model based keyword recognition system. In *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pages 129–132. IEEE.

Salvador, S. and Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580.

Schlömer, T., Poppinga, B., Henze, N., and Boll, S. (2008). Gesture recognition with a wii controller. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 11–14. ACM.

Schwarz, L. A., Mkhitaryan, A., Mateus, D., and Navab, N. (2012). Human skeleton tracking from depth data using geodesic distances and optical flow. *Image and Vision Computing*, 30(3):217–226.

Soltani, F., Eskandari, F., and Golestan, S. (2012). Developing a gesture-based game for deaf/mute people using microsoft kinect. In *Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on*, pages 491–495. IEEE.

Starner, T. E. (1995). Visual recognition of american sign language using hidden markov models. Technical report, DTIC Document.

Stone, E. E. and Skubic, M. (2015). Fall detection in homes of older adults using the

microsoft kinect. *IEEE journal of biomedical and health informatics*, 19(1):290–301.

Tamura, Y., Umetani, T., Kashima, N., and Nakamura, H. (2014). Dynamic gesture classification using skeleton model on RGB-D data. In *Journal of Physics: Conference Series*, volume 490, page 012103. IOP Publishing.

Tollenaere, T. (1990). Supersab: fast adaptive back propagation with good scaling properties. *Neural networks*, 3(5):561–573.

Tsai, G. (2010). Histogram of oriented gradients. *University of Michigan*.

Vaananen, K. and Bohm, K. (1993). Gesture driven interaction as a human factor in virtual environments-an approach with neural networks. *Virtual reality systems*, pages 93–106.

Wilson, A. D. and Bobick, A. F. (1999). Parametric hidden markov models for gesture recognition. *IEEE transactions on pattern analysis and machine intelligence*, 21(9):884–900.

Wu, Y. and Huang, T. S. (1999). Vision-based gesture recognition: A review. In *Gesture Workshop*, volume 1739, pages 103–115. Springer.

Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 379–385. IEEE.

Yi, B.-K., Jagadish, H., and Faloutsos, C. (1998). Efficient retrieval of similar time sequences under time warping. In *Data Engineering, 1998. Proceedings., 14th International Conference on*, pages 201–208. IEEE.

Yim, H. (2013). A state-based approach to the gesture recognition. *School of Electrical and Electronic Engineering, Yonsei University*.

Ying, W. and LIU, Z.-d. (2016). Kinect gesture recognition method using track sequence. *DEStech Transactions on Computer Science and Engineering*, (cmee).

Yu, S.-Z. and Kobayashi, H. (2003). An efficient forward-backward algorithm for an explicit-duration hidden markov model. *IEEE signal processing letters*, 10(1):11–14.

Zhang, B., Yang, Y., Chen, C., Yang, L., Han, J., and Shao, L. (2017). Action recognition using 3d histograms of texture and a multi-class boosting classifier. *IEEE Transactions on Image Processing*, 26(10):4648–4660.

Zhang, C. and Tian, Y. (2012). RGB-D camera-based daily living activity recognition. *Journal of Computer Vision and Image Processing*, 2(4):12.

Zhao, X., Li, X., Pang, C., Sheng, Q. Z., Wang, S., and Ye, M. (2014). Structured streaming skeleton–a new feature for online human gesture recognition. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11(1s):22.

Zhou, L., Li, W., Zhang, Y., Ogunbona, P., Nguyen, D. T., and Zhang, H. (2014). Discriminative key pose extraction using extended lc-ksvd for action recognition. In *Digital lmage Computing: Techniques and Applications (DlCTA), 2014 International Conference on*, pages 1–8. IEEE.