参赛队员姓名：马浩宇

中学：华东师范大学第二附属中学

省份：上海

国家/地区：中国

指导教师姓名：王振堂

论文题目：**The Design of Low-cost Data Glove and Research on Medium Vocabulary Continuous Sign Language Recognition**

参赛团队声明所提交的论文是在指导老师指导下进行的研究工作和取得的研究成果。尽本团队所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果。若有不实之处，本人愿意承担一切相关责任。

参赛队员： 马浩宇 指导老师：王振堂

2017 年 **11** 月 **29** 日

The Design of Low-cost Data Glove and Research on Medium Vocabulary Continuous Sign Language Recognition

Abstract

Sign language bridges the communication between the deaf-mute and the rest of the world. With the development of computer technology, the research on sign language recognition is prospering. Currently, there are two kinds of research methods. One is based on machine vision, the other is based on wearable input devices. Comparing with machine vision, wearable devices have the advantage of being able to get real-time information of the bend of the fingers and the movement of the hand. The essay applies recent technology in the field of voice recognition, and aims at finding out the best algorithm for medium vocabulary continuous sign language recognition. By building a hybrid Deep Neural Network-Hidden Markov Model, and designing and making a low-cost digital glove, the essay successfully compares the features between Dynamic Time Warping (DTW), Gaussian Mixture Model-Hidden Markov Model (GMM-HMM) and Deep Neural Network-Hidden Markov Model (DNN-HMM) as of solving the problem of sign language recognition. Tests show that in terms of calculating the observation probability, DNN has a much better performance than HMM, especially when syntax is not provided, meaning that DNN better suits the developing trend of sign language recognition. Real-time recognition is achieved on the intelligent terminal with an accuracy of over 97%, using the trained model and a decoding program.

Key words: Low cost; DTW; GMM-HMM; DNN-HMM; Medium vocabulary; Continuous sign language recognition

Contents

Abstract 3
Chapter 1 Introduction 6
1.1 The Significance of the Research 6
1.2 Research Overview of Sign Language Recognition 6
1.2.1 Sign Language Recognition Based on Wearable Input Devices 6
1.2.2 Sign Language Recognition Based on Machine Vision 8
1.3 The Project's Aim 8
1.4 Main Innovative Points 9
Chapter 2 The Hardware Design of the Data Glove 10
2.1 Design Review 10
2.2 Conversion of Resistance to Voltage 14
2.3 Serial Port Usage 17
2.4 Arduino Data Transmission 19
2.5 Power Supply Design 22
2.6 Validation of Functions 23
Chapter 3 The Design on Data Acquisition and Recognition System 25
3.1 Design Review 25
3.2 Reception and Storage of the Data 26
3.3 Processing and Analysis of the Data 28
3.4 Recording and Display of the Sample 30
3.5 The Invocation Relationship between Classes 33
Chapter 4 The Application and Analysis of the Algorithm 35
4.1 Algorithm Design Summary 35
4.2 The Application of Dynamic Time Warping 37
4.3 The Application of the Hidden Markov Model 38
4.4 The Application of the Deep Neural Network 45
4.5 Data Analysis 47
Chapter 5 Real-time Decoding 50
5.1 The Significance of Real-time Decoding 50

5.2 Real-time Decoding for DTW 51
5.3 Real-time Decoding for HMM 53
5.4 Real-time Decoding Summary 59
Chapter 6 Conclusion and Outlooks 60
6.1 Conclusion 60
6.2 Outlooks 61
Chapter 7 Reference 62
Chapter 8 Acknowledgements 64

Chapter 1 Introduction

1.1 The Significance of the Research

Sign language is the body language of hand shape, arm movement and the expression of the thought through expression, lip movement and other body potential. It has a normative grammar, clear semantics and a complete vocabulary system. [1]Chinese sign language is mainly divided into two categories: finger language and sign language. Finger language is the description of a Chinese pinyin alphabet with the trace of the finger, and is a language based on Chinese pinyin rules. Sign language is a language using the main means of simulating the shape and movement of things, and is supplemented by gestures and expressions. The goal of sign language recognition is to translate finger language and sign language into a natural language, such as text or speech, so as to enable the communication between deaf and normal people. There are more than 20 million deaf people in China, and the research on the language recognition will undoubtedly benefit this group directly. The study will provide a faster, more natural, and more convenient way for them to communicate with the hearing people, and in this way, they can better fit into society. It will also have a positive impact for the construction of harmonious society of diverse caring.

With the rapid development of computer technology, intelligence has become one of the main directions of development. To make machines think like people and make judgment has always been the ultimate goal in the field of computer intelligence. At present, deep learning has achieved remarkable development in image recognition and speech recognition. Sign language recognition and speech recognition are closely linked together. Similar to speech, gestures have the characteristics of large amount of data and strong timing, therefore, sign language recognition can perfectly inherit the frontier research results in speech recognition.

1.2 Research Overview of Sign Language Recognition

There are two mature technologies in the study of sign language: sign language recognition based on wearable input devices and sign language recognition based on machine vision.

1.2.1 Sign Language Recognition Based on Wearable Input Devices

In sign language recognition based on wearable input devices, the input devices commonly used are data gloves and three-dimensional tracking devices. Data gloves can give information about the bending of the fingers and changes of gestures, and 3d tracking devices can give the position of objects in real time. According to this feature, by combining the data glove with the 3d tracking device, the position and posture of the hand and arm in space can be obtained with an appropriate algorithm. The sign recognition system based on the input device can be further divided into finger language recognition, sign language word recognition and continuous sign language recognition.

As a kind of static gesture word, finger language has the characteristics of being easy to recognize and convenient to try new algorithms. Many early researchers had used it as the starting point for the study of sign recognition. Typical work of finger language recognition include: Takahashi and Kishino from the Japan ATR lab designed a VPL data glove system based on the coding technique of the angle of the joints and orientation of the hands, which can identify about 34 of the 46 Japanese letters.

Compared with finger language, the expression of sign language words is a dynamic time series, which must be considered in recognition and modelling. The representative work of isolated sign language word recognition is: S.S. Fels and G.E. Hinton used a VPL data glove and a Polhemus position tracker as input device and the neural network as the gesture classifier. They built five function networks according to the movement of the hand, the direction of the hand, the movement of the hand, the offset of the hand, and the movement speed of the hand to identify 203 hand words.

In the aspect of continuous sign language recognition, the representative work includes: R.H.Liang and Ouhyoung used the hidden markov model to realize the continuous Taiwan sign language recognition translator. The system is aimed at the the basic words and practice sentences in the Taiwan sign language textbooks, and the vocabulary is 71-250. The data of sign language is collected using data gloves made by VPL Company. Gaowen of the Chinese academy of sciences [6] used the two CyberGlove data gloves and a 3d tracking device as the input device to develop a set of CSL (Chinese Sign Language) recognition system. In the user-dependent aspect, the identification rate of 5100 isolated words was about 94%. In the non-user-dependent

aspect, the data of 6 people were collected, with the same 5100 word vocabulary, and the average recognition rate was 91%.

1.2.2 Sign Language Recognition Based on Machine Vision

Because of the occlusion, projection, and light influence of 2d visual images, it is difficult to accurately track the bending information of each finger based on the visual method. Therefore, the research work on the visual sign language recognition system is concentrated between the small vocabulary and the medium vocabulary. The advantage of this method are that it can simultaneously detect the movements of the speaker's hand and facial expressions and the body gestures (such as nods, sways, etc.). It also frees user of the input devices while recognizing. Therefore, the system cost based on machine vision is relatively low.

In the aspect of sign language recognition based on machine vision, representative works are: C.Charayaphan and A. Marble used image processing to identify 31 isolated gestures in ASL, and ended up correctly identifying 27. M.H. Yang and others used the motion trajectory to extract and classify the 2d movement in the image sequence. The recognition rate of the 40 American Sign Language words reached 98.14% using the time delay neural network. K. Grobel and M. Assan from Germany used the hidden markov model to identify 262 isolate Dutch words with a recognition rate of 91.3%. The recognition method of the system is based on vision, and the method is to make the testers wear colored gloves and then extract the two-dimensional features through video.

In the aspect of continuous sign language recognition, the representative works is: T.S. Tarner of the multimedia laboratory at the Massachusetts institute of technology conducted a study on the continuous recognition of American Sign Language. They used the feature vectors of the handshape, orientation of the hand and motion track information to as the input of the hidden markov model to recognize ASL. In order to be able to track, they asked users to wear colored gloves, whereas the right glove was yellow and the left glove was orange. The test was carried out in a short sentence consisting of 40 words randomly, the recognition rate of the system was 91.3%, and the real-time recognition rate was 98% after the addition of certain grammatical constraints.

1.3 The Project’s Aim

Although there has been detailed studies of hand gesture recognition based on data glove, no product has been widely used by the deaf. One possible reason is the high cost of data gloves. The data glove on the market is mainly for VR use, so the price is generally high. Take the product of DataGlove Inc., which specializes in data gloves, as an example, the cheapest data glove costs $585.00, or about 3,800 yuan. If this technology is to be promoted, it is vital lower the costs and meanwhile maintain the same recognition accuracy.

The project aims to achieve real-time continuous sign recognition of medium vocabulary by designing low-cost sign language data gloves. In the field of sign language recognition, an available vocabulary of less than 100 words is defined as a small vocabulary, of 100-500 words is a medium vocabulary and of 500 words and above is a big vocabulary. [1] Due to the limited time, the range of vocabulary selected in this project is the medium vocabulary, and the identification range is limited to the identification of the specific person. The words identified in the project come from the book *Chinese sign language everyday conversation*, which contains 500 words and meets the standard of a medium vocabulary.

1.4 Main Innovative Points

The innovative points of this project are:

1. Designed and built the low-cost gloves for sign language using common low cost component. Tests show that the gloves can be used to collect the data.
2. Established a deep neural network - hidden Markov hybrid model for sign language recognition using the new technology in speech recognition.
3. Established the training and real time recognition system based on DTW, GMM-HMM and DNN-HMM, and made the comparison of the three different recognition methods in time complexity and accuracy.
4. Completed the real-time identification of sign language on the intelligent terminal.

Chapter 2 The Hardware Design of the Data Glove

2.1 Design Review

In order to achieve the goal of sign language recognition, the data glove needs to collect two aspects of data: finger and gesture. Finger data, recorded by the flex sensors, are used to determine the hand posture, while gesture data, recorded by a three-axis gyroscope, are used to determine the movement of the hand. In order to be able to receive process and the above two kinds of data, a single chip microcomputer is needed. In order to transmit data wirelessly to an intelligent terminal, a Bluetooth module is required.

The block diagram of the acquisition system is shown in figure 2.1.1

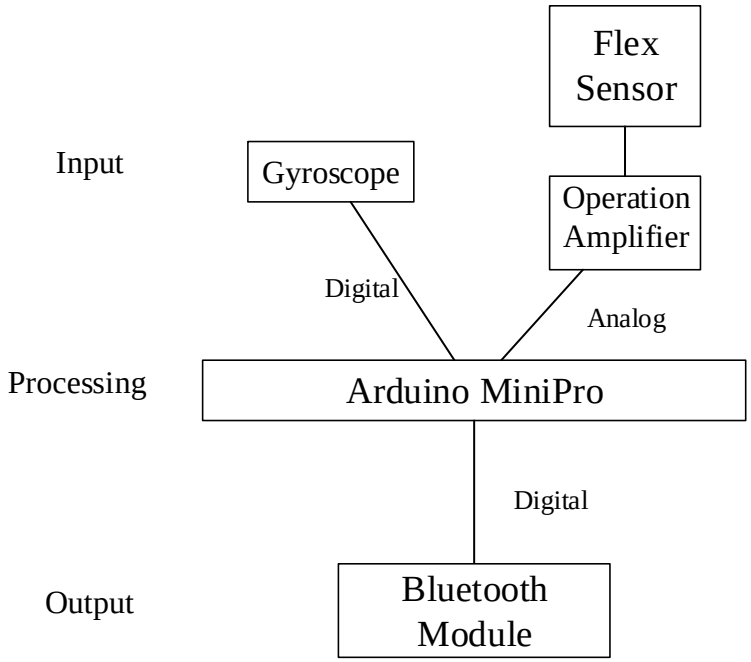

Figure 2.1.1: Block diagram of the acquisition system

Regarding the choice of the flex sensor, the common products available are produced by Spectra Symbol®, the types used in the project are 2.2’’ and 4.5", the 2.2" ones are used on the thumb and little finger, while the 4.5" ones are used on the index finger, the middle finger and the ring finger.

As for the gyroscope, the model used in the project is a MPU6050 three-axis gyroscope, the data types provided are acceleration, angular velocity and angle. The origin of the coordinate

system of the gyroscope is always itself. The acceleration measured is an composition of the gravitational acceleration and the acceleration due to external forces, so the direction of the motion can be judged by the projection of the acceleration on the axis. On the other hand, the process of motion can be determined by angular velocity. Neither the data of acceleration nor the data of angular velocity is affected by orientation, which means that they can describe the trajectory effectively. The data from the gyroscope are three sequential packets of data, with 11bytes for each packet. The three packets are the acceleration packet, the angular velocity packet and the angle packet. The data format is shown in table 2.1.1, 2.1.2, 2.1.3.

Data number	Data Context	Meaning
0	0x55	packet header
1	0x51	It's an acceleration packet
2	AxL	x-axis acceleration low byte
3	AxH	x-axis acceleration high byte
4	AyL	y-axis acceleration low byte
5	AyH	y-axis acceleration high byte
6	AzL	z-axis acceleration low byte
7	AzH	z-axis acceleration high byte
8	TL	temperature low byte
9	TH	temperature high byte
10	Sum	checksum

Table 2.1.1: Data format for an acceleration packet

Data number	Data Context	Meaning
0	0x55	packet header
1	0x52	It's an angular velocity packet
2	wxL	x-axis angular velocity low byte
3	wxH	x-axis angular velocity high byte
4	wyL	y-axis angular velocity low byte
5	wyH	y-axis angular velocity high byte
6	wzL	z-axis angular velocity low byte
7	wzH	z-axis angular velocity high byte
8	TL	temperature low byte
9	TH	temperature high byte
10	Sum	checksum

Table 2.1.2: Data format for an angular velocity packet

Data number	Data Context	Meaning

0	0x55	packet header
1	0x53	It’s an angle packet
2	RollL	x-axis angle low byte
3	RollH	x-axis angle high byte
4	PitchL	y-axis angle low byte
5	PitchH	y-axis angle high byte
6	YawL	z-axis angle low byte
7	YawH	z-axis angle high byte
8	TL	temperature low byte
9	TH	temperature high byte
10	Sum	checksum

Table 2.1.3: Data format for an angle packet

The actual processing of data needs only data number 2-7. The packet headers (0 and 1) are only used to locate the data and determine the data type, and the checksum (10) is only used to test the effectiveness of the data. The size of the gyroscope is 15.24*15.24 mm. The gyroscope supports the serial port and the I2C interface, and can work under 115200/9600baud. The working voltage is 3.3v and the working current is 10mA. The encapsulation is surface mount.

In the case of single-chip microcomputer, in order to achieve the purpose of being wearable, the size of the circuit board needs to be small, limiting the overall thickness. The commonly used Arduino Uno is large and thick due to the existence of the pin headers, thus it is not suitable for the circuit board design in the project. After searching the hardware base, the project chose the smallest microcontroller with no pin headers in the Arduino series - Arduino ProMini. For the purpose of low power consumption, the selected microcontroller is a 3.3v working voltage version, with an ATmega328P as the main processor. The clock frequency is 8MHz, the size is 33* 18mm, and the power consumption is about 3.2 mA. Because there is no stamp hole on the package, it cannot be directly welded to the circuit board. Therefore, the outer edge of the microcontroller is cut so that the round holes are made into stamp holes for welding. Arduino ProMini possesses 8 analog inputs and 1 serial port, thus it can receive the data from the flex sensors in the normal manner. However, in serial communication, a new way must be found to meet the demand of the gyroscope and Bluetooth module for serial port.

In the aspect of Bluetooth module, in order to guarantee the low power consumption, the project chooses the BLE4.0 module, which is 15.1*11.2 mm, with a working voltage of 3.3V and

working current < 6mA, and is encapsulated as surface mount. When using the Bluetooth data channel, the maximum payload of each packet is 20 bytes.

2.2 Conversion of Resistance to Voltage

A flex sensor is equivalent to a changeable resistance that changes according to the bending degree, and because the processing core used is a single chip microcomputer, it is necessary to convert the resistance value to the voltage value and to access analog input for measurement. The range of resistance of the flex sensors is 7kΩ-20kΩ(4.5″) and 20kΩ-40kΩ (2.2″). There are two commonly used conversion methods, using a simple voltage division circuit to convert or using operation amplifiers to convert.

The simple voltage division circuit is shown in figure 2.2.1:

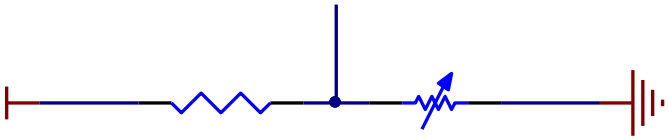

Figure 2.2.1: A simple voltage division circuit

Let the voltage across the flex sensor be U_2, then

$$U_{2max} = \frac{V_{cc}}{R_1 + R_{2max}} \times R_{2max} < V_{cc} \ (1)$$

$$U_{2min} = \frac{V_{cc}}{R_1 + R_{2min}} \times R_{2min} > 0 \ (2)$$

From the inequalities (1)(2), it can be seen that when using the simple voltage division circuit, the interval of change of voltage across the flex sensor is rather small and cannot reach VCC and GND.

The operation amplifier circuit is shown in figure 2.2.2:

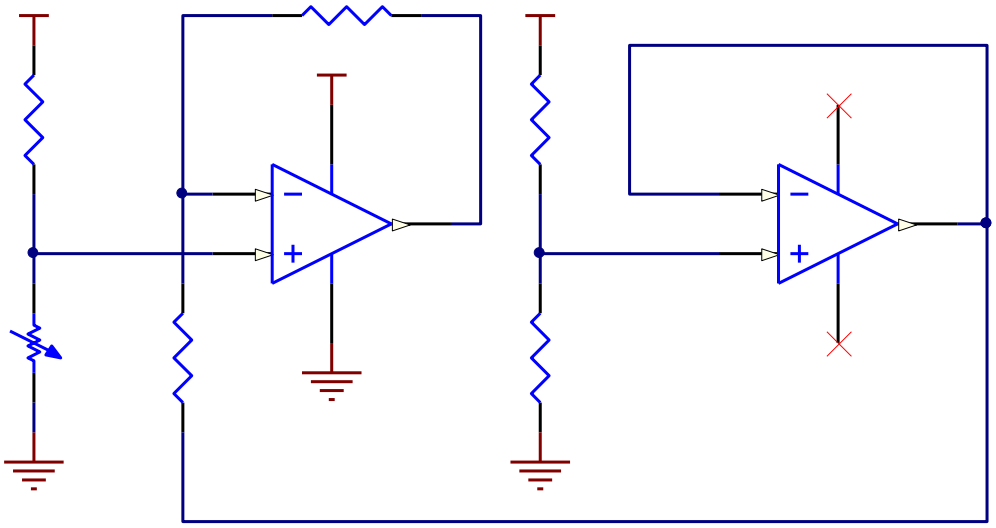

Figure 2.2.2：An operation amplifier circuit

In this circuit, pin 1 of the operation amplifier is connected to the analog input of the microcontroller. Let the voltage at each pin of the operation amplifier relative to the ground be U_1、U_2、U_3 ……U_7, the voltage across the resistors be U_{R1}、U_{R2}、U_{R3} ……U_{R6}. Using the feature of virtual short circuit and virtual open circuit of the operation amplifier, we can get

$$U_7 = U_6 = U_5 = \frac{V_{cc}}{R_5 + R_6} \times R_6 \quad (3)$$

$$U_2 = U_3 = \frac{V_{cc}}{R_1 + R_2} \times R_2 \quad (4)$$

$$U_{R4} = U_7 - U_2 = V_{cc} \times \left(\frac{R_6}{R_5 + R_6} - \frac{R_2}{R_1 + R_2}\right) \quad (5)$$

$$\mathrm{I} = \frac{U_{R4}}{R_4} = \frac{V_{cc} \times \left(\frac{R_6}{R_5 + R_6} - \frac{R_2}{R_1 + R_2}\right)}{R_4} \quad (6)$$

$$U_1 = U_2 + I \times R_3 = V_{cc} \times \left[\frac{R_3}{R_4} \times \frac{R_6}{R_5 + R_6} + \left(1 - \frac{R_3}{R_4}\right) \times \frac{R_2}{R_1 + R_2}\right] \quad (7)$$

Let $R_5 = R_6, R_3 = 3R_4,\ R_1 = R_{2max}$. Because $R_{2min} = \frac{1}{2} R_{2max}$, we can know that when R_2 takes the minimum value, $U_1 = 0$; when R_2 takes the maximum value, $U_1 = V_{cc}$. In other words, the interval of change of the output voltage U_1 can reach VCC and GND when R_2 changes.

The output voltage of the simple voltage division circuit cannot reach VCC and GND while the one of the operation amplifier circuit can, which means that the operation amplifier circuit is superior to the simple voltage division circuit in the range of output voltage. Therefore the project chose the operation amplifier circuit as the means to convert resistance to voltage.

For the selection of operational amplifier, the rail to rail operational amplifier should be selected. The input and output of the ordinary operational amplifier cannot reach VCC and ground, but the input and output of the rail to rail transit can reach the power supply voltage and the ground. For example, in this project, the power voltage is 3.3 V, and the flex sensor is a 4.2 " model. If the non-rail-to-rail operation amplifier LM324 is used, the voltage change is 1.0V-2.5V when the resistance of the flex sensor changes. When the rail-to-rail operational amplifier AD8604 is used, the voltage change is 0V-3.3 V, and there is a significant increase in the variation. The project chose AD8604 as the operational amplifier for this reason, and its current is about 0.75mA at the working voltage of 3.3 V.

2.3 Serial Port Usage

In the project, the three-axis gyroscope is used to record the gesture changes, and the Bluetooth module is used to transmit the data to the upper computer. Both modules need to connect to the microcontroller by serial port. However, the microcontroller used in the project, the Arduino ProMini, is a small single chip microcomputer and has only one serial port, so direct connection is not an option.

The initial connection method is Serial Peripheral Interface (SPI), and the two microcontrollers are respectively the master and the slave. The master connects the gyroscope while the slave connects the Bluetooth module. As the Arduino ProMini itself has no USB port, it's not convenient to debug. So Arduino Uno, that has an USB port, is used to debug. The processing speed and main vibrating frequency of Arduino Uno are both higher than Arduino ProMini, thus it is competent in the feasibility test. The results of the SPI feasibility test showed that packet dropout rates were very high. A possible is that Arduino itself has poor support of the SPI protocol, which leads to the inconsistency of clock frequency in asynchronous communication. For the above reasons, the project can only give up the SPI connection and use other methods.

After analyzing the actual situation, it can be found that the serial ports of gyroscope and Bluetooth module are divided into Tx (transmission) and Rx (receiving) interface, but actually, the gyroscope only needs to transmit data, and the Bluetooth module only needs to receive data. They both use only half of the serial port. Therefore, it is possible to divide the serial port in two, and connect the Tx of the gyroscope to the Rx of Arduino and the Tx of Arduino to the Rx of the Bluetooth module (Figure2.3.1).

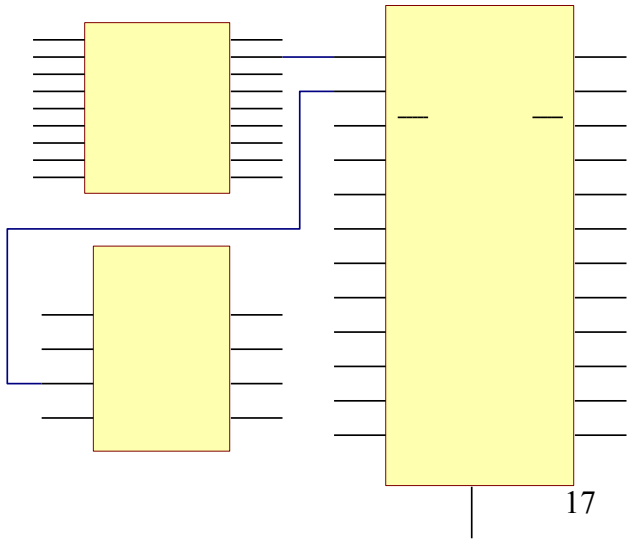

Figure 2.3.1: The gyroscope and the Bluetooth module connect to Arduino ProMini separately via serial port

Using Arduino Uno for the feasibility test, the project found that this transmission method can realize the normal transmission of the data and the packet loss rate is zero. The experiment proves that this simple method can be used for the separate use of Tx and Rx of the serial port.

2.4 Arduino Data Transmission

After completing the feasibility test on serial port use, Arduino ProMini was used in practical tests. The transmission rate was normal in the test, but there was still some loss or dislocation in the data received. After reviewing on chip data, I found that the problem may be caused by the error of baud rate.

The actual transmitting baud rate of a microcontroller is determined by its main frequency and custom parameters. The calculation formula is $BAUD = \frac{fosc}{16(UBRRn+1)}$, whereas f_{osc} is the clock speed of the microcontroller and UBRRn is an user-defined integer ranging from 0-4095.In the project, the clock speed of Arduino ProMini is 8MHz. If the common 115200Baud was used in transmission, the closest possible Baud rate would be $\frac{8\times10^6}{16\times(3+1)} = 125000Baud$, and the error would be 8.5%. After checking the chip manual, it was found that the maximum error allowed under the project's data transfer status (8 bit, no parity) was plus or minus 4.5%, and the recommended maximum error was plus or minus 2.0%. The actual error was way above the allowed maximum error, therefore packet losses occur. Since the output of the gyroscope has only 115200Baud and 9600Baud, the project considered using the lower baud rate of 9600Baud for transmission to reduce the error. When 9600 Baud was used for transmission, the closest possible Baud rate was $\frac{8\times10^6}{16\times(51+1)} = 9615Baud$, and the error was only 0.16%, within the recommended maximum error. After using the new baud rate, the packet loss detection was carried out again. The test data showed that the data transmission rate was stable, the checksum was 100% correct, and the packet loss rate was zero, which was in line with the transmission requirement.

After completing the research on the transmission method, the paper designed the data collection and preprocessing program on Arduino. The program block diagram is shown in figure 2.4.1.

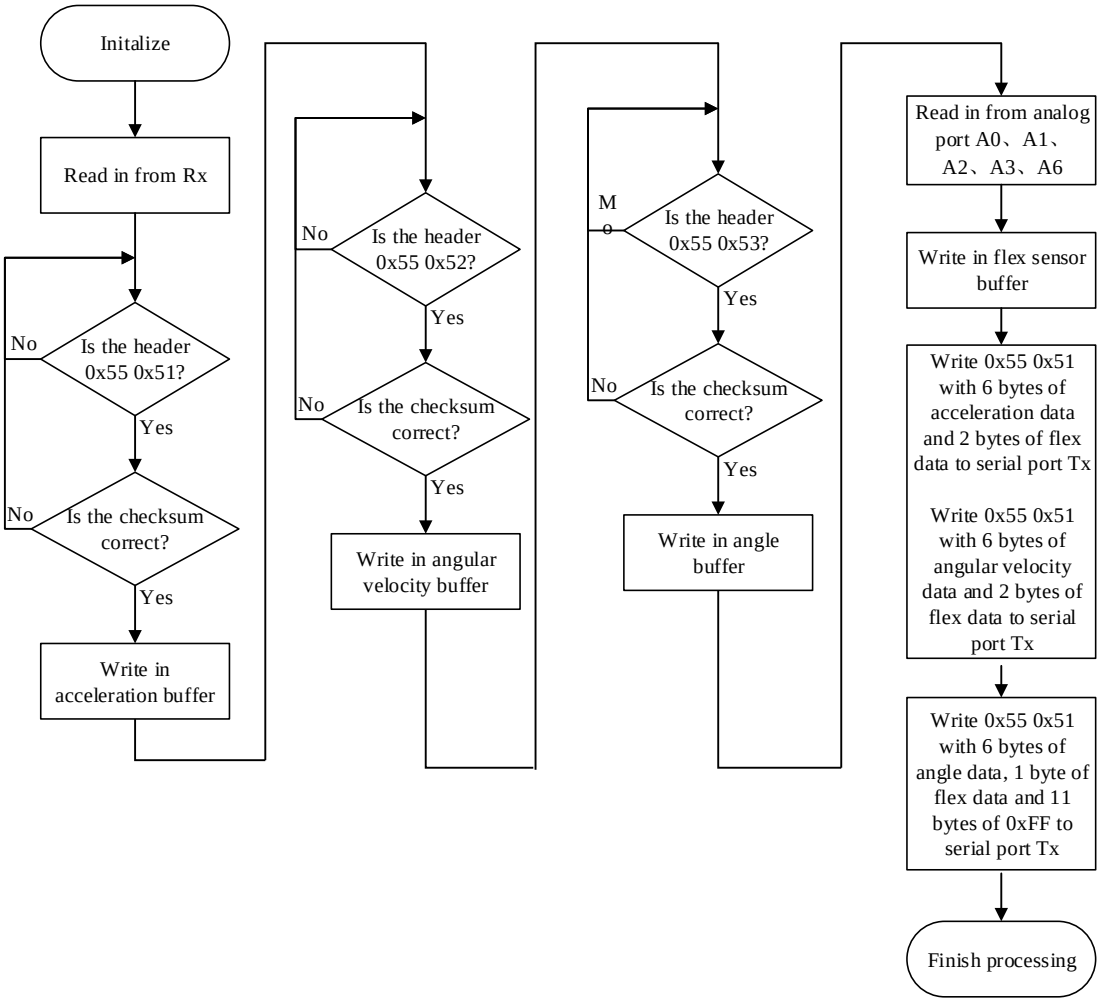

Figure 2.4.1: The block program for the data collection and preprocessing program

In the process of transmission, the program will first check the checksum of every packet of data. Because the gyroscope data itself has been packaged, to reduce the amount of data, the gyroscope data is unpacked for further processing. Each flex sensor data is a one byte hexadecimal integer. After collecting all three packets of gyroscope data, the gyroscope data, along with the data of the flex sensors are packed into two packets and sent to the Bluetooth module. The data format is shown in table 2.4.1.

Data number	Data Context	Meaning
0	0x55	packet header
1	0x51	It’s an acceleration packet
2	AxL	x-axis acceleration low byte
3	AxH	x-axis acceleration high byte
4	AyL	y-axis acceleration low byte
5	AyH	y-axis acceleration high byte

6	AzL	z-axis acceleration low byte
7	AzH	z-axis acceleration high byte
8	Flex1	flex sensor data 1
9	Flex2	flex sensor data 2
10	0x55	packet header
11	0x52	It’s an angular velocity packet
12	wxL	x-axis angular velocity low byte
13	wxH	x-axis angular velocity high byte
14	wyL	y-axis angular velocity low byte
15	wyH	y-axis angular velocity high byte
16	wzL	z-axis angular velocity low byte
17	wzH	z-axis angular velocity high byte
18	Flex3	flex sensor data 3
19	Flex4	flex sensor data 4

Data number	Data Context	Meaning
0	0x55	packet header
1	0x53	It’s an angle packet
2	RollL	x-axis angle low byte
3	RollH	x-axis angle high byte
4	PitchL	y-axis angle low byte
5	PitchH	y-axis angle high byte
6	YawL	z-axis angle low byte
7	YawH	z-axis angle high byte
8	Flex5	flex sensor data 5
9-19	0xFF	fill

Chart 2.4.1: The Bluetooth data format

2.5 Power Supply Design

The main power consumption components in the circuit are the single chip microcomputer, the Bluetooth module, the gyroscope and operational amplifier. According to the parameters of the components, when the power voltage is 3.3v, the trunk current is less than 22.2 mA. To meet the design requirement of being able to work for 10 hours and the size confinement of smaller than 30* 30mm, the final selection of the paper is a 300mAh lithium battery, with a size of 30*25mm and the nominal voltage of 3.7v. To make it easier to charge the battery, the circuit board comes with a standard micro-USB interface installed on the edge of the circuit board. To control the battery's charging or power supply, a switch is installed on the circuit board. In order to ensure the wearability of the system, the size of the switch is required for height < 1mm, length < 2mm, width < 4mm, and encapsulated as surface mount. The selected SMT switch has a height of 0.7mm, a length of 1.4mm, and a width of 3.1 mm, which meets the design requirements.

2.6 Validation of Functions

The circuit board size is 36.8*34.3 mm, and the schematic diagram is shown in fig.2.6.1.

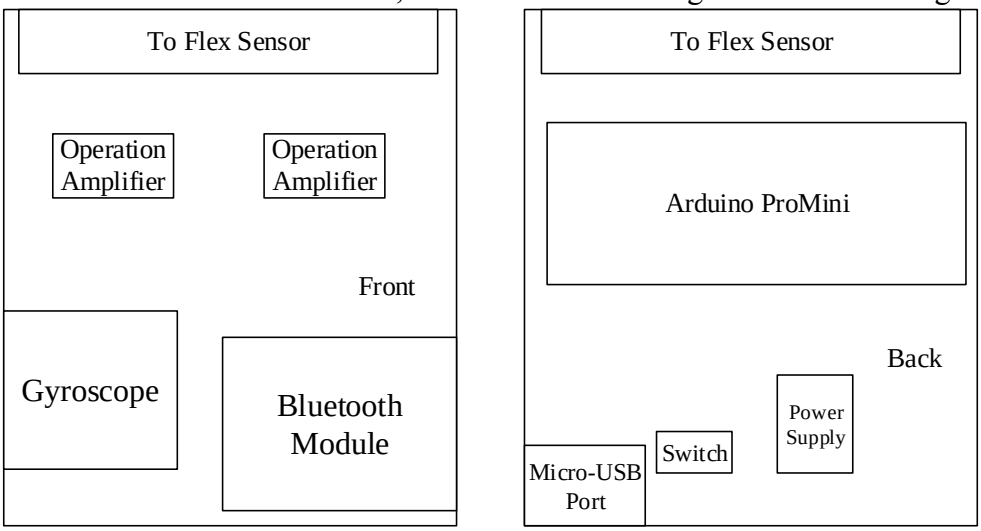

Figure 2.6.1: The schematic diagram of the circuit board

The cost of each glove is 660.5 yuan (the bill of material is shown in figure 2.6.2). The entire system (two gloves) costs 1,321 yuan, far below the price of existing data gloves.

Component list — 手语手套

Source Data From: 手语手套.SchDoc
Project: Free Documents
Variant: None

Altium — Think it, Design it, Build it™

Report Date: 2016/11/27 15:24:22
Print Date: 04-Feb-17 4:33:06 PM

#	LibRef	Description	Footprint	Quantity	Unit Price
1	Flex 4.5"	Flex sensor 4.5"	6-0805_N	3	118
2	Flex 2.2"	Flex sensor 2.2"	HDR1X2	2	92.5
3	LM324D	Quadruple Operational Amplifier	D014_N	2	7.5
4	Arduino Mini Pro	Arduino Mini Pro	Arduino Mini Pro	1	8.8
5	BLE 4.0	Bluetooth Low energy	BLE4.0	1	13.9
6	Battery	Lithium battery(small)		1	28
7	Charger	USB charger	Micro_USB	1	7
8	Charger Port	Header, 5-Pin	Micro_USB	1	0.14
9	Gyroscope	Gyroscope Module	Gyroscope	1	47.25
10	SW-SPDT	SPDT Subminiature Toggle Switch, Right Angle Mounting, Vertical Actuation	Switch	1	1.48
				14	660.57

Figure 2.6.2: Bill of material of the gloves

According to the design diagram, the circuit is set up for testing. The trunk current was 20.5 mA, a less than 10% error from the calculated 22.2mA. Considering the characteristics of the Bluetooth module, the error is within acceptable range. When bending the flex sensor, the

voltage range of the thumb, middle finger and index finger is 0v-3.3 V, and the small thumb is 0v-3.1 V, and the ring finger is 0v-2.5v, which basically meets the design requirements. The relatively small voltage variation of the ring finger may be due to the limitation of its own bending amplitude.

After making sure the data is collected, a program on the iPad is written, which uses the Bluetooth protocol to read the data and write it as a file for further processing. The files are then read using a computer to check and verify the validity of the data. In the 2679 groups of data collected, the accuracy of the checksum is 100%, and when the flex sensor bends, the change in the voltage value can be seen. The variation is the same as the results obtained using a multimeter.

The reason why iPad is used for programming is that Apple's support for the Bluetooth protocol is superior to that of android, and the data transmission is more stable. Second, apple's hardware compatibility is good, so it can be easily ported to other Apple intelligent terminals such as the iPhone.

Chapter 3 Design on the Data Acquisition and Recognition System

3.1 Design Review

There are mainly three design goals that the system needs to achieve: in order to monitor the connection and data of Bluetooth in real time, the Bluetooth data reception and storage module is required; in order to be able to determine the validity of the data and ensure that it can be found in time when the gloves are abnormal (such as the occurrence of disconnection), the data processing and analysis module is needed; in order to facilitate the recording of samples and find out about the sample content recorded, the sample recording and the corpus annotation display module is required. The interface of the system is shown in figure 3.1.1.

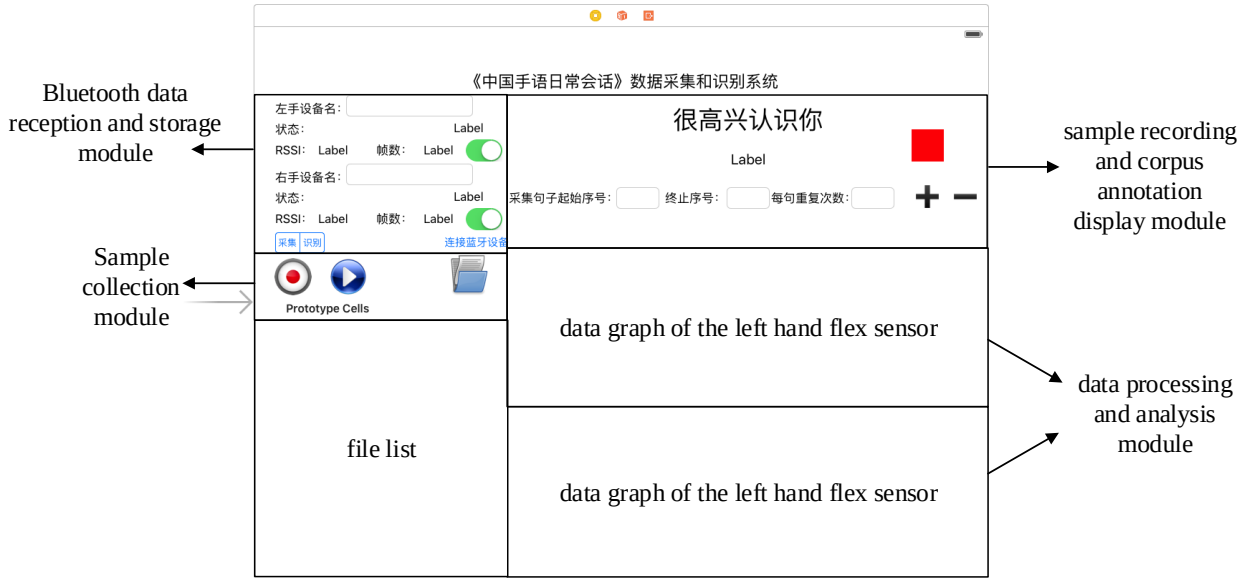

Figure 3.1.1: The interface of the Data Acquisition and Recognition System

3.2 Reception and Storage of the Data

The program designed by the project receives Bluetooth data through the CBCentralManagerDelegate protocol, and the block diagram is shown in figure 3.2.1.

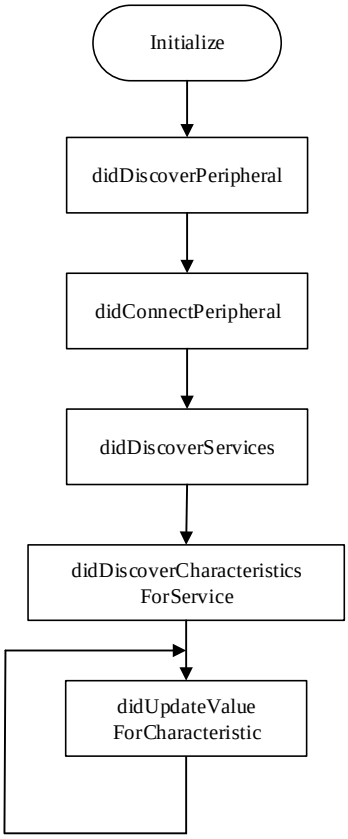

Figure 3.2.1: The block diagram of the reception of the data

When the CBCentralManager calls any function, the program updates the state of the system and displays it on the user interface. The specific process is: Peripheral Discovered→Peripheral Connected→Services Discovered→Characteristic Discovered。

The output parameters of the didDiscoverPeripheral function contains (RSSI*) RSSI, which records the signal strength values at the time when the function is called and displays in the user interface.

The output parameters of the didUpdateValueForCharacteristic function contains (CBCharacteristic*) characteristic, namely the Bluetooth data that is wrapped and packaged into

a CBCharacteristic object. Each time the function is called, 1 is added to the frame number which appears on the user interface.

Because the communication mode between Bluetooth and Arduino ProMini is asynchronous communication, it might happen that a packet of data sent by Arduino is split by the Bluetooth module and sent in two packets. If the data is stored without processing, the data loss and dislocation will occur in the subsequent identification, affecting the overall recognition effect. The custom DataBuffer class is used to handle this situation.

The external interface provided by DataBuffer is

- (void)writeBuffer:(NSString*) incomingStr;

- (NSString*)readBuffer;

The writeBuffer function uses the loop array to implement the temporary storage of the Bluetooth data, while the readBuffer function extracts the complete packet from the loop array through the lookup method of the regular expression and outputs as NSString *.

3.3 Processing and Analysis of the Data

It is found in the test that the left hand Bluetooth module transmits data faster than the right hand one. Therefore, in data reception, situations like the left hand has sent 3 packets of data and the right hand only 1 package can occur. If you simply put a packet of the left and a packet of the right data together, there might be a left hand data overflow. The program uses a custom MessageBuffer class to handle this problem.

The external interface provided by MessageBuffer is

```
- (MessageBuffer*)init;
- (NSString*)getFullMessage:(NSString*)message forHand:(NSString*)hand;
```

The init function is used to generate counter and the ring array for the cache. The getFullMessage function caches the data from both hands and joins them together. When the left data overflows, it no longer uses the real time data from the right hand, but uses the latest data from the right hand. The result of the splicing is output as NSString *.

After the data is pieced together, the data is unwrapped. The program defines a structure called gloveData and the definition is as follows

```
typedef struct{
   double acceleration[6];
   double angularVelocity[6];
   double angle[6];
   int flex[10];
}GloveData;
```

The structure consists of four arrays, storing data of acceleration, angular velocity, Angle and flex sensor values respectively. Unpack is done in view controller. The process separates acceleration, angular velocity, angle, and flex sensor data from the data processed by the MessageBuffer, and uses the defined structure to store the data. After separating the data, the actual value of the data is obtained by using the given algorithm. The drawing is done using the DataShowView class.

The DataShowView does not provide an external interface, and only contains the drawRect function that the system calls automatically, which is used to retrieve data from the gloveData structure and draw the line graph that is shown in the user interface, as shown in figure 3.3.1.

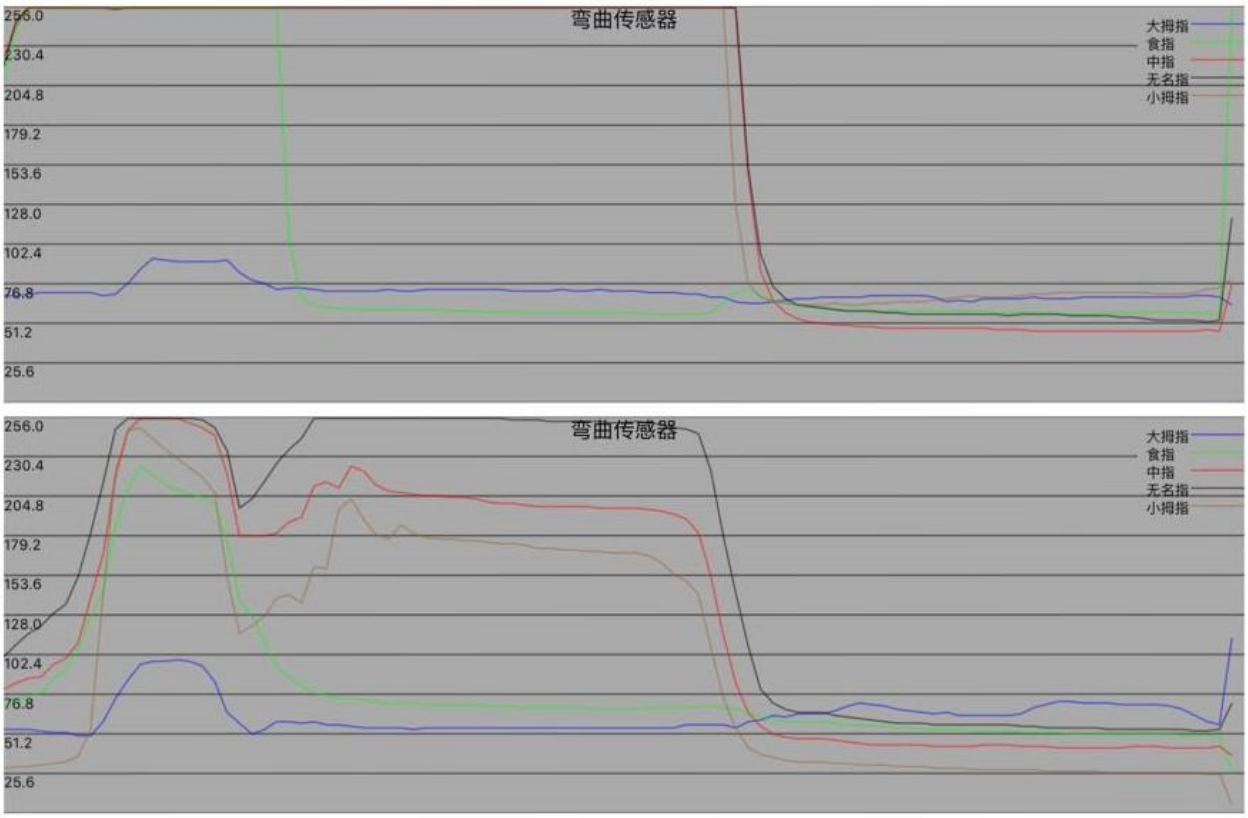

Figure 3.3.1: The line graph of the flex sensors

3.4 Recording and Display of the Sample

In the process of recording the samples, the features that need to provided include displaying and reading out the sentences of the current record, show/modify repetition times, show/modify serial number of the recorded sentence, erasing the wrong data from the previous action and determining the user's record state. The implementation functions of the first three functions are in view controller, and the Read function is implemented using the AVSpeechSynthesizerDelegate protocol. The last two functions are implemented using a custom ActivityDetection class.

The methods of determining actions in the program are as follows: for the flex sensor and the angular velocity data, the values in the static state should be close to zero, so whether the maximum value of the data exceeds the threshold determines whether it is an action state; for the acceleration data, processing is done through a sliding window, with a window size of 5. Since there is gravity acceleration in the stationary state, whether the difference between the maximum and minimum values of the data in the window exceeds the threshold determines whether it is an action state. If any of the three aspects of the data (angular velocity, acceleration and flex sensor) is judged to be in the action state, then the data will be judged as the action data.

In class, five possible recording states are defined through enumeration, namely INIT (initialization), SENTENCE_START (action start), SENTENCE_ONGOING (action ongoing), SENTENCE_END (action end) and IDLE. The status transition diagram is shown in figure 3.4.1.

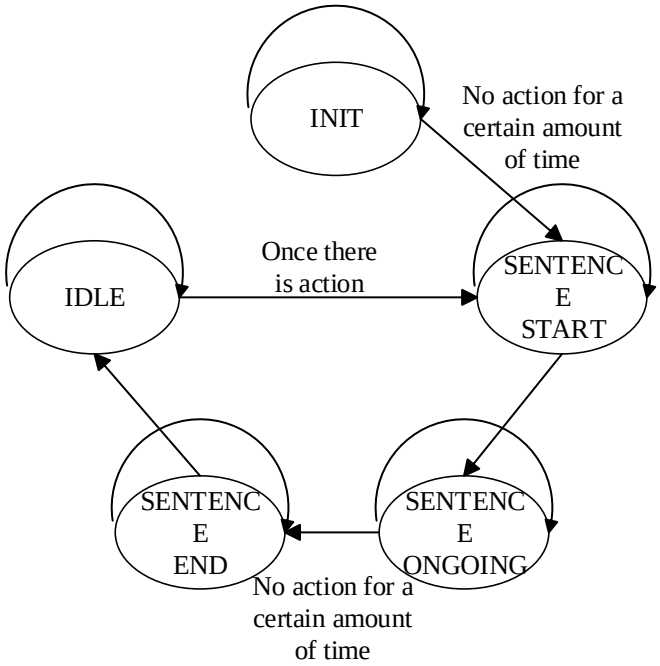

Figure 3.4.1: The status transition diagram

It is to take a certain amount of time to get into the SENTENCE_START state after INIT. This is because in the solo recording process, it takes a certain amount of time for the hands to return to the body side after pressing the record key, in order to not record the data in this period, there needs to be a certain delay. In contrast, there is no delay in entering the SENTENCE_START state from the IDLE state once action is detected. This is to ensure not missing out any data. In different states, a block in the user interface displays different colors to remind users.

The external interface provided by ActivityDetection is

- (ActivityDetection*)init;

- (NSInteger)detectActivity:(GloveData)gloveData;

- (void)resetState;

The function of the init function is to generate the threshold value of various data action decisions for initialization. The function of the resetState function is to reset the state to INIT, which is not only used for initialization, but also for erasing the previous error data. Because each sample need to be repeated many times, errors are unavoidable. When an error occurs, press the record button again to erase the data from the previous action, and the resetState function is

called to restore the state to INIT. The role of detectActivity is to determine whether a piece of data is an action data and to update the status based on the decision.

3.5 The Invocation Relationship between Classes

The invocation relationship between classes in the program is shown in figure 3.5.1.

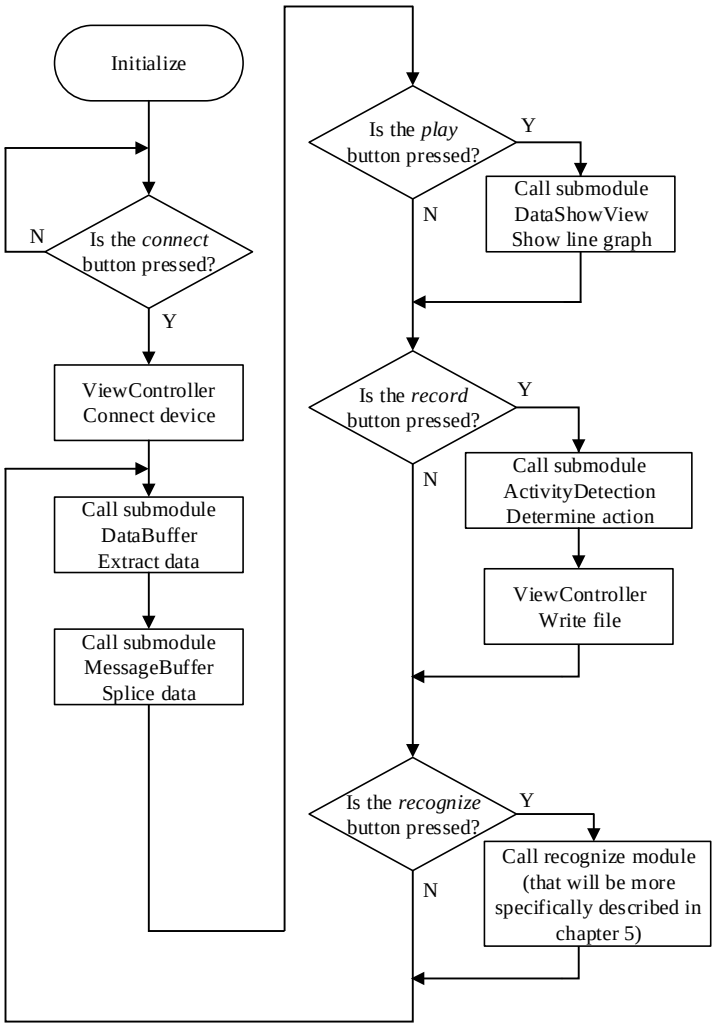

Figure 3.5.1: The invocation relationship between classes

When writing a file, the file name is GloveData+4 character starting sentence number (including) +4 characters terminating sentence number (excluding) +4 character sentence repetition times, as a TXT file type. Take the record of the first sentence as an example, the number of repetitions is 30 times, and the filename is shown in figure 3.5.2.

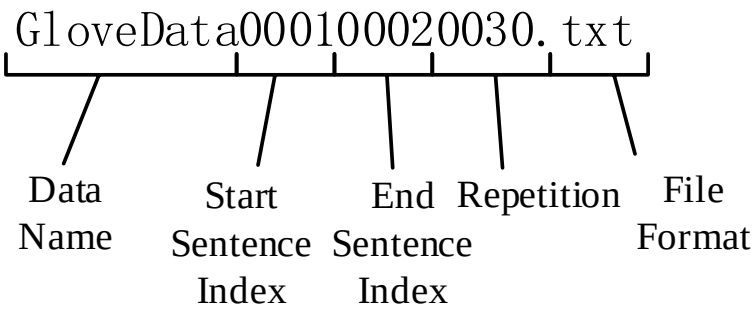

Figure 3.5.2 File name example

The contents of the file are spliced by the left and right data, with 128 characters per line, as in table 3.5.1

Data number	Data Context	Meaning
1-4	sentenceIndex	Index of the sentence
5-8	repetitionNumber	Repetition number
9-68	leftData	Left hand data
69-128	rightData	Right hand data

Table 3.5.1 File data format

In this paper, 335 sentences were recorded using the sign language glove and the acquisition program, and the number of sentences repeated was 30 times.

Chapter 4 Application and Analysis of the Algorithm

4.1 Algorithm Design Summary

Nowadays, the algorithms for sign language recognition mainly include dynamic time warping (DTW), deep neural network (DNN) and hidden markov model (HMM). The paper aims to compare the complexity and accuracy of the three algorithms to find the most suitable algorithm for the continuous sign language recognition.

Prior to the application of the algorithm, the sample data collected was preprocessed. The original data contained the header 0x55 0x51/0x52/0x53 that was used for positioning and was in the format of txt, which made it inconvenient to analyze the data directly. Therefore, the header was removed in processing and the remaining data was written as a binary file in the MFC format specified by Htk. [11]In the subsequent data analysis, 85 percent of the 30 sets of data recorded previously was used as training data, and 15 percent was used as test data.

In addition, the paper also divided all the sentences in the book *Chinese sign language everyday conversation,* that is to split apart all the sign language words in a sentence. The project made a *dictionary* handDict according to the extracted hand words, with the meaning of the sign language word as the *key*, and the serial number of sign language words, obtained according to the basic Chinese sign language standard GBT24435-2009, as the corresponding *value*. Because the HMM library Htk used in the paper did not support Chinese, the paper also made the needed wordAlias.

As for the dimension of the data, there are 5 dimensions for the flex sensors on each hand, 3 dimensions for acceleration and 3 dimensions for angular velocity, making the total data dimension of each hand 11 and of both hands 22.

After analyzing the gesture in sign language, it was found that in sign language, the left hand was mainly used as the auxiliary hand, and the curvature of the fingers might have little influence on the identification process. If the left hand flex sensor can be removed without affecting the final identification result, not only can the complexity of the algorithm be reduced, but the cost of the system can also be greatly reduced, which is in line with the research purpose of low cost.

To verify this hypothesis, the results with the left hand flex sensor (22 dimensions) and without the left hand bending sensor (17 dimensions) will be presented in the analysis of the data, and the comparison will be made.

4.2 The Application of Dynamic Time Warping

Dynamic time warping (DTW) is an algorithm based on dynamic programming (DP), which solves the problem of template matching with different pronunciation. By using the time scale function which satisfies a certain condition, the time corresponding relationship between the test template and the reference template can be described, and the corresponding regulation function of the smallest cumulative distance of the two template matching can be found. 错误!未找到引用源。。

Because only one template was needed in the DTW algorithm, one of the 25 training data was randomly selected as a template. The test data was then compared with all the templates and the template with the smallest corresponding distance was selected as the identification result. Because of the randomness of the template selection in the test, the selected template may not be the most typical one, thus there might be fluctuations in the accuracy. The paper carried out 5 rounds of tests in 22 and 17 dimensions, taking the mean as the test result.

The mean accuracy was 99.17 per cent in the 22 dimension test, which means there were 14 errors on average in the 1687 sets of data. The mean accuracy of the 17 dimensional test was 99.20%, which means there was 13.5 errors on average in the 1687 sets of data. It can be concluded from the data above that in the sign language recognition based on DTW, the recognition accuracy of the 22-dimensional data is similar to the recognition accuracy of the 17-dimensional data, with the latter slightly higher by 0.03 percent. Therefore in the sign language recognition based on DTW, the 17 dimensional data can replace the 22 dimensional data without affecting the recognition accuracy.

4.3 The Application of the Hidden Markov Model

The hidden markov model (HMM) is a statistical model used to describe a markov process with unknown parameters. The specific algorithm is to determine the implicit parameters of the process from the observable parameters, and then use these parameters for further analysis [9]。

In this paper, the training and identification of mono-phoneme and tri-phoneme hidden markov models was done using the Htk toolkit. The parameters required for training included training configuration, HMM model, model list, dictionary (previously prepared wordDict), training corpus annotation, and training file list. The recognition method was Viterbi decoding, and the additional parameter needed was the syntax.

The difference between the training configuration and the default values is shown in table 4.3.1

TARGETKIND = MFCC
CEPLIFTER = 22*
NUMCEPS = 22*
FORCECXTEXP = T
ALLOWXWRDEXP = T

Table 4.3.1: The training configuration

The above configuration declares the dimension of training data and support for tri-phoneme recognition. If the 17-dimensional data is used, the number before * should be replaced by 17.

In the paper, each hand word was modeled, and the model used in this paper is a Gaussian Mixture Model. The number of states of each model is 5, the model name is the corresponding number of the sign language words, and the list of mono-phoneme models is made, as shown in table 4.3.2.

G100011
G100034
……
G998
sil

Table 4.3.2: A sample of mono-phoneme model lists

A list of corpus annotation for a mono-phoneme training is made using the previously prepared wordAlias and the corresponding serial number of each sign language word. Take the sentence "很高兴认识你", which means “Glad to know you” in English as an example, the left side is the corpus annotation on the word level, and the right side is the corpus annotation on the phoneme level, as shown in table 4.3.3.

SENT-START	sil
hen3	G172
gao1xing4	G1856
ren4	G269
shi2	G209
ni3	G166
SENT-END	sil
.	.

Table 4.3.3: Corpus annotation on the word and phoneme level

The training syntax is a list of sentences written according to Htk, as in table 4.3.4.

$sentence-body=
hen3 gao1xing4 ren4shi2 ni3
| hen3 gao1xing4 ren4shi2 ni3 men2
……
| zi4zhu4 you2 hai2 shi4_2 can1jian1 lv3you2 tuan2
;
(SENT-START $sentence-body SENT-END)

Table 4.3.4: The training syntax sample

Using the software tool provided by Htk, the following script was wrote in Matlab to implement the training and recognition of the HMM and output the test results.

```
system(['HCompV -C ','myConfig -f 0.01 -m -S ','myTrain_Mono.scp -M ','Hmms
','myProto']);
system(['HERest -C ','myConfig -I ','myLabels_Mono -t 250.0 150.0 1000.0 -S
','myTrain_Mono.scp -H ', ...
```

```
        'Hmms/macros_Mono -H ','Hmms/hmmdefs_Mono -M ','Hmms/Mono
','myPhones_Mono']);
    for n=1:4
        system(['HERest -C ','myConfig -I ','myLabels_Mono -t 250.0 150.0 1000.0 -S
','myTrain_Mono.scp -H ', ...
            'Hmms/Mono/macros_Mono -H ','Hmms/Mono/hmmdefs_Mono ','myPhones_Mono']);
    end
    system(['HHEd -H ','Hmms/Mono/macros_Mono -H ','Hmms/Mono/hmmdefs_Mono
','G0.hed ','myPhones_Mono']);
    for n=1:36
        system(['HERest -C ','myConfig -I ','myLabels_Mono -t 250.0 150.0 1000.0 -S
','myTrain_Mono.scp -H ', ...
            'Hmms/Mono/macros_Mono -H ','Hmms/Mono/hmmdefs_Mono ','myPhones_Mono']);
    end
    system(['HVite -H ','Hmms/Mono/macros_Mono -H ','Hmms/Mono/hmmdefs_Mono -S
','myTest_Mono.scp -l * -i ', ...
        'myResult_Mono.mlf -w ','myWordNet_Mono -p 0.0 -s 5.0 ','myDict_Mono
','myPhones_Mono']);
    system(['HResults -I ','myTestLabels_Mono ','myPhones_Mono ','myResult_Mono.mlf']);
    system(['HVite -H ','Hmms/Mono/macros_Mono -H ','Hmms/Mono/hmmdefs_Mono -S
','myTest_Mono.scp -l * -i ', ...
        'myResult_MonoNoContext.mlf -w ','myWordNet_MonoNoContext -p 0.0 -s 5.0
','myDict_Mono ','myPhones_Mono']);
    system(['HResults -I ','myTestLabels_Mono ','myPhones_Mono
','myResult_MonoNoContext.mlf']);
```

The test results of the 22-dimensional mono-phoneme model were as table 4.3.5 (no syntax) and table 4.3.6 (with syntax).

```
====================== HTK Results Analysis =======================
  Date: Tue Jan 31 14:43:25 2017
  Ref : D:/gestureRecognize22/myTestLabels_Mono
  Rec : D:/gestureRecognize22/myResult_MonoNoContextFix.mlf
```

```
---------------------- Overall Results -------------------------
SENT: %Correct=24.36 [H=411, S=1276, N=1687]
WORD: %Corr=84.27, Acc=69.89 [H=9241, D=252, S=1473, I=1577, N=10966]
================================================================
```

Table 4.3.5: Test results for 22-dimensional mono-phoneme model (no syntax)

```
====================== HTK Results Analysis =======================
  Date: Tue Jan 31 13:35:30 2017
  Ref : D:/gestureRecognize22/myTestLabels_Mono
  Rec : D:/gestureRecognize22/myResult_Mono.mlf
---------------------- Overall Results -------------------------
SENT: %Correct=97.93 [H=1652, S=35, N=1687]
WORD: %Corr=99.73, Acc=99.64 [H=10936, D=15, S=15, I=9, N=10966]
================================================================
```

Table 4.3.6 Test results for 22-dimensional mono-phoneme model (with syntax)

The test results of the 17-dimensional mono-phoneme model were as table 4.3.7 (no syntax) and table 4.3.8 (with syntax).

```
====================== HTK Results Analysis =======================
  Date: Tue Jan 31 12:18:18 2017
  Ref : D:/gestureRecognize/myTestLabels_Mono
  Rec : D:/gestureRecognize/myResult_MonoNoContextFix.mlf
---------------------- Overall Results -------------------------
SENT: %Correct=32.13 [H=542, S=1145, N=1687]
WORD: %Corr=88.34, Acc=69.65 [H=9672, D=83, S=1193, I=2047, N=10948]
================================================================
```

Table 4.3.7 Test results for 17-dimensional mono-phoneme model (no syntax)

```
====================== HTK Results Analysis =======================
  Date: Sun Jan 29 13:28:52 2017
  Ref : D:/gestureRecognize/myTestLabels_Mono
```

Rec : D:/gestureRecognize/myResult_Mono.mlf
---------- Overall Results ----------
SENT: %Correct=97.51 [H=1645, S=42, N=1687]
WORD: %Corr=99.63, Acc=99.53 [H=10908, D=9, S=31, I=12, N=10948]
==

Table 4.3.8 Test results for 17-dimensional mono-phoneme model (with syntax)

Whereas H stands for correct number, N stands for total test number, D, S, I stand for errors, namely, D stands for deletion error (for example “很高兴认识你”→“很高兴认你”), S stands for replacement error (for example “很高兴认识你”→“很高兴认识他”), I stands for insertion error (for example “很高兴认识你”→“很高兴认识你们”). SENT stands for sentence accuracy, any word error in the sentence makes the sentence an error. In the WORD part, Corr stands for word accuracy not considering insertion errors, and Acc stands for word accuracy considering insertion errors [11].

It can be seen that the syntax has a great influence on the GMM - HMM sign language recognition. The main reason is that the syntax can reduce the incorrect paths in Viterbi decoding, thus reducing the possibility of error.

After completing the mono-phoneme test, the paper carried out the tri-phoneme test. The three-phoneme HMM gesture recognition took into account the former phoneme of a phoneme and the latter phoneme, so the differences in preparation of the documents were mainly reflected in the model list and the corpus annotation.

The model list format is as table 4.3.9:

sil
sil-G172+G1856
……
G136-G509+G2131
G509-G2131+sil

Table 4.3.9: A sample of tri-phoneme model lists

The corpus annotation format is as table 4.3.10:

sil
sil-G172+G1856
G172-G1856+G269
G1856-G269+G209
G269-G209+G166
G209-G166+sil
sil
.

Table 4.3.10: Tri- phoneme corpus annotation

The test results of the 22-dimensional tri-phoneme model were as table 4.3.11 (no syntax) and table 4.3.12 (with syntax).

====================== HTK Results Analysis ======================
Date: Tue Jan 31 15:00:28 2017
Ref : D:/gestureRecognize22/myTestLabels_Mono
Rec : D:/gestureRecognize22/myResult_TriPhone.mlf
---------- Overall Results ----------
SENT: %Correct=99.70 [H=1682, S=5, N=1687]
WORD: %Corr=99.97, Acc=99.95 [H=10963, D=0, S=3, I=2, N=10966]
==

Table 4.3.11 Test results for 22-dimensional tri-phoneme model (with syntax)

====================== HTK Results Analysis ======================
Date: Sun Jan 29 15:05:57 2017
Ref : D:/gestureRecognize/myTestLabels_Mono
Rec : D:/gestureRecognize/myResult_TriPhone.mlf
---------- Overall Results ----------
SENT: %Correct=99.29 [H=1675, S=12, N=1687]
WORD: %Corr=99.95, Acc=99.86 [H=10942, D=2, S=4, I=9, N=10948]
==

The test without syntax couldn't be carried out for the tri-phoneme HMM. Suppose there are *n* mono-phoneme models, then there will be n^3 tri-phonemes models. These combinations do not necessarily exist in the training data, if one or more of these combinations does not exist, the

Table 4.3.12 Test results for 17-dimensional tri-phonemes model (with syntax)

necessary decoding parameters cannot be generated, and there will be errors when decoding, thus only through clustering method can this approximation be done. Because of the limited time, research hadn't been carried out on this part.

Comparing table 4.3.6 with table 4.3.8, table 4.3.7 with table 4.3.9, table 4.3.11 with table 4.3.12, it can be concluded that the recognition accuracy of 17-dimensional data and 22-dimensional data in the GMM-HMM model is basically the same. In the case of syntax, the accuracy of the 17-dimensional data was slightly lower than that of the 22-dimensional data by 0.43% (mono-phoneme) and 0.41% (tri-phoneme), which is neglectable. In the case of mono-phoneme without syntax, the accuracy of the 17-dimensional data was even higher than the 22-dimensional data by 31.90%. Therefore, in the GMM-HMM model, the simpler 17-dimensional data could be used instead of 22-dimensional data without affecting the recognition accuracy. This not only reduced the data complexity, but also reduced the total cost of the system from 1,321 yuan to 782 yuan, down 40.8 percent.

4.4 The Application of the Deep Neural Network

Deep neural network (DNN) is a mathematical model of distributed parallel information processing by abstracting the human neural network from the angle of information processing. This network relies on the complexity of the system to deal with the purpose of processing information by adjusting the interconnection between large number of internal nodes [12][14]. Although the neural network method has the characteristics of classification and anti-interference, it has only been applied to the recognition of static gestures because of its weakness in processing time series. [13]. Considering that DNN is better at dealing with static problems, the paper considers using DNN instead of GMM to complete the sample observation probability calculation. The current relatively common DNN-HMM combination is divided into series and hybrid, and the subject chose a hybrid model because of the limited time.

To get better results in the training, the samples were expanded. The original 1 sample point was expanded to a neighboring 9 sample points (the four sample points before the sample, the sample point itself, and the four sample points after the sample). The sample points for starting and termination are replenished by repeating the data of the start or termination sample four times. The training data number is $190 \times 335 \times 25 = 1591250$ in total, each data contains $9 \times 17 = 153$ dimensions of the sample and 1182 dimensions of the label. The data is too large to use the neural network for a one-time training. Since MATLAB did not support batch training of neural network, and the Htk version that supports neural network hadn't released its windows version, the paper chose the Python based Keras library which supported batch neural network training for DNN training. The neural network used was multi-layer perceptron (MLP), with a dropout rate of 10% and a three-layer configuration. There were 1500 neurons on the first and second layers, and the activation function is *relu*, while the third layer had 1182 neurons, and the activation function is *softmax*. The number of neurons in the third layer was the total number of states of all sign language words.

Because the data needed to be passed between Matlab (HMM training), Python (DNN training) and C ++ (Viterbi decoding), it was necessary to store data in a format that was supported by all three. The paper selects the h5 (hdf5) format. The file includes the 17 dimensional maximum and minimum values of the data, the total number of the states of the sign language words, the training data, the training data label, the test data, and the test data label. The test results are as table 4.4.1 (no syntax) and table 4.4.2 (with syntax).

```
====================== HTK Results Analysis =======================
  Date: Sun Jan 29 16:27:51 2017
  Ref : D:/gestureRecognize/myTestLabels_Mono
  Rec : D:/gestureRecognize/viterbiDecodeResultFix
------------------------ Overall Results --------------------------
SENT: %Correct=85.93 [H=1447, S=237, N=1684]
WORD: %Corr=97.39, Acc=95.71 [H=10635, D=77, S=208, I=183, N=10920]
================================================================
```

Table 4.4.1 DNN-HMM test results (no syntax)

```
====================== HTK Results Analysis =======================
  Date: Mon Jan 30 10:21:01 2017
  Ref : D:/gestureRecognize/myTestLabels_Mono
  Rec : D:/gestureRecognize/viterbiDecodeResult
------------------------ Overall Results --------------------------
SENT: %Correct=97.92 [H=1649, S=35, N=1684]
WORD: %Corr=99.75, Acc=99.67 [H=10893, D=12, S=15, I=9, N=10920]
================================================================
```

Table 4.4.2 DNN-HMM test results (with syntax)

4.5 Data Analysis

After completing the application of various algorithm models, the recognition accuracy of various models was summarized as figure 4.5.1.

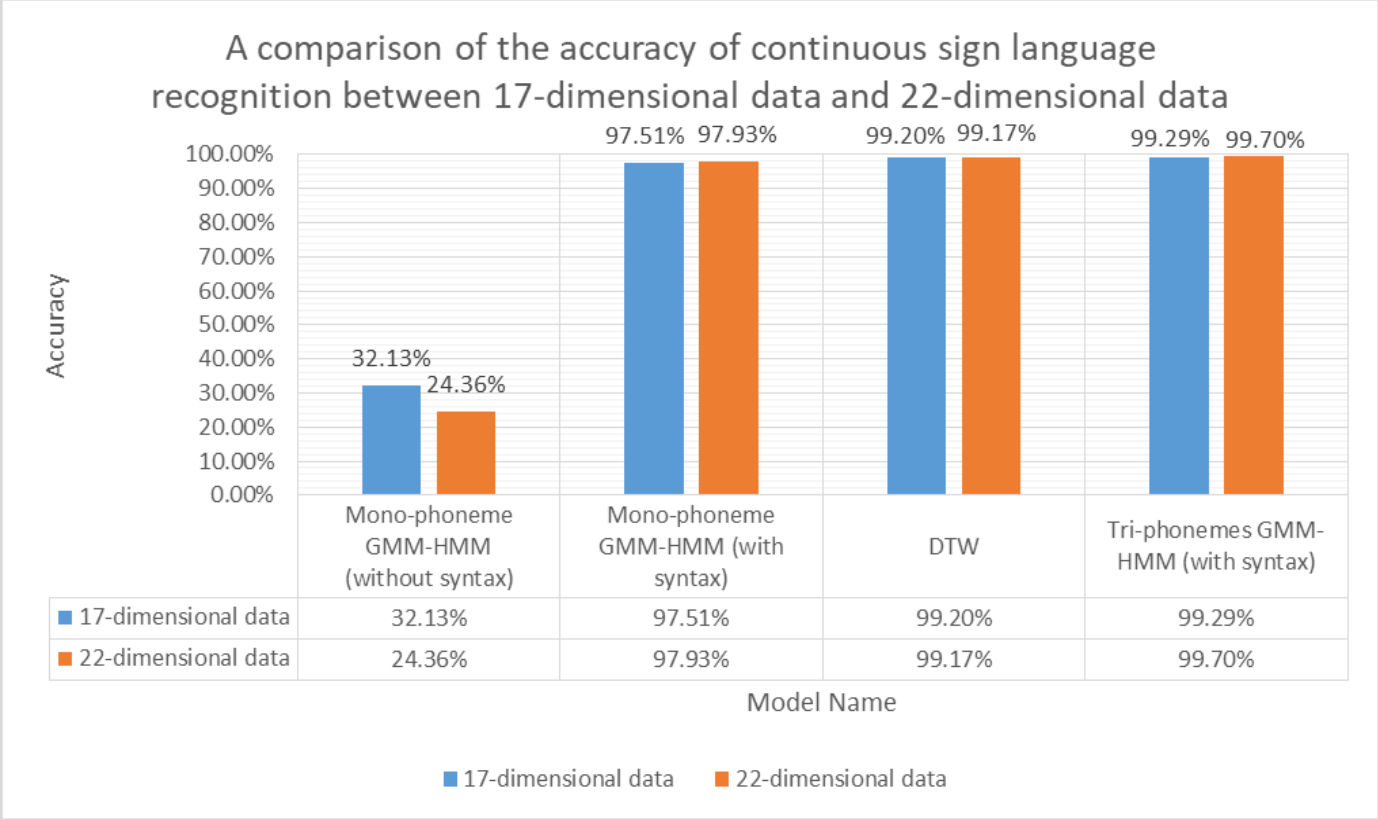

Figure 4.5.1: A comparison of the accuracy of the continuous sign language recognition between 17-dimensional data and 22-dimensional data

Figure 4.5.1 shows that in the recognition of DTW and HMM, the accuracy result obtained by using the 17-dimensional data is similar to that of using 22 dimensional data, and the maximum difference is no more than 0.50%. In DTW and mono-phoneme no syntax HMM recognition, a higher accuracy can even be obtained by using the 17-dimensional data. The above data shows that within the scope of this paper, the 17-dimensional data can replace the 22 dimensional data for model training and recognition, without affecting the final accuracy. Therefore, the 17-dimensional data is used in subsequent analysis.

The accuracy comparison of continuous sign language recognition based on different models is shown in figure 4.5.2.

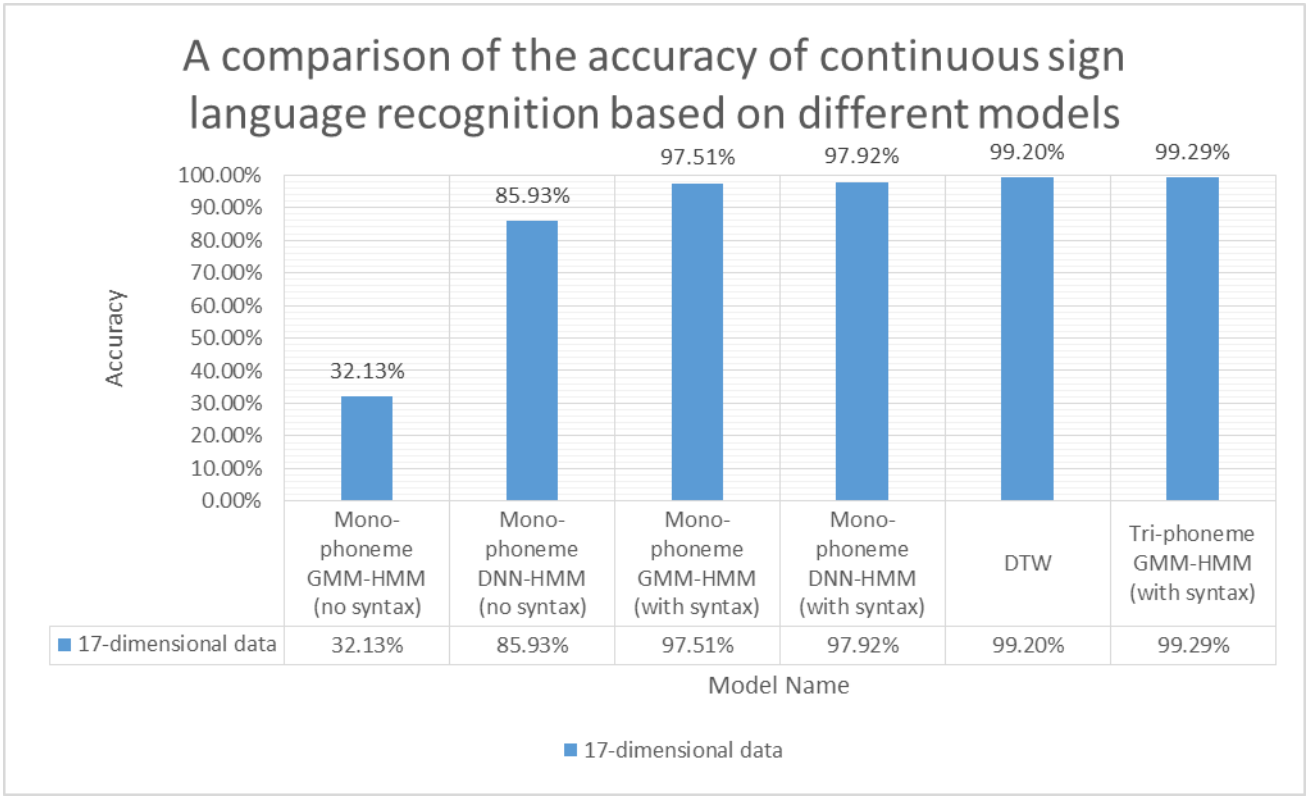

Figure 4.5.2: A comparison of the accuracy of continuous sign language recognition based on different models

It can be concluded from figure 4.5.2 that using DNN instead of GMM to calculate the observation probability of the hidden markov model can improve the recognition accuracy. In the absence of syntax, the improvement effect is more significant, which is an increase of up to 167.44%, but the improvement of accuracy is relatively small in the case of syntax, which is a mere 0.42%. In the case of syntax, increasing the number of phonemes can improve the accuracy

of recognition more effectively. It can be seen from the comparison between the recognition accuracy of mono-phoneme and tri-phoneme that the tri-phonemes can improve the accuracy of 1.82% in GMM-HMM model compared to mono-phoneme.

Chapter 5 Real-time Decoding

5.1 The Significance of Real-time Decoding

Because the purpose of this paper is to complete the real-time continuous sign language recognition, after the training model parameters are entered into the intelligent terminal, real-time decoding on intelligent terminals is required to achieve the purpose of identification. In the process of real-time decoding, this paper will focus on the complexity of decoding algorithm and test on the actual processing time. The transmitting rate of the data in the paper is 20 frames per second, or 50 milliseconds per frame. For HMM, the actual processing time for each frame must be less than 50 milliseconds to ensure real-time data processing. As DTW requires a complete sentence of data to start processing, the requirements for processing time can be slightly reduced, but no more than 1 second for each sentence.

In this paper, two kinds of decoding methods are used, one is for the decoding of DTW, and the other is Viterbi decoding for HMM. The DNN-HMM model and the GMM-HMM model mentioned above both uses Viterbi decoding, and the only differences is how the observation probability is read, so they are classified together.

5.2 Real-time Decoding for DTW

In the program, the DTW class is used for real-time decoding of DTW, and the external interface is as follows

-(DTW*)init;

-(int)recognize:(NSMutableArray *)testData;

-(void)dealloc;

The init function is used to initialize, the dealloc function is used to release the memory that is applied through malloc, and the recognize function is used for real-time identification.

The real-time decoding block diagram of DTW is shown in figure 5.2.1.

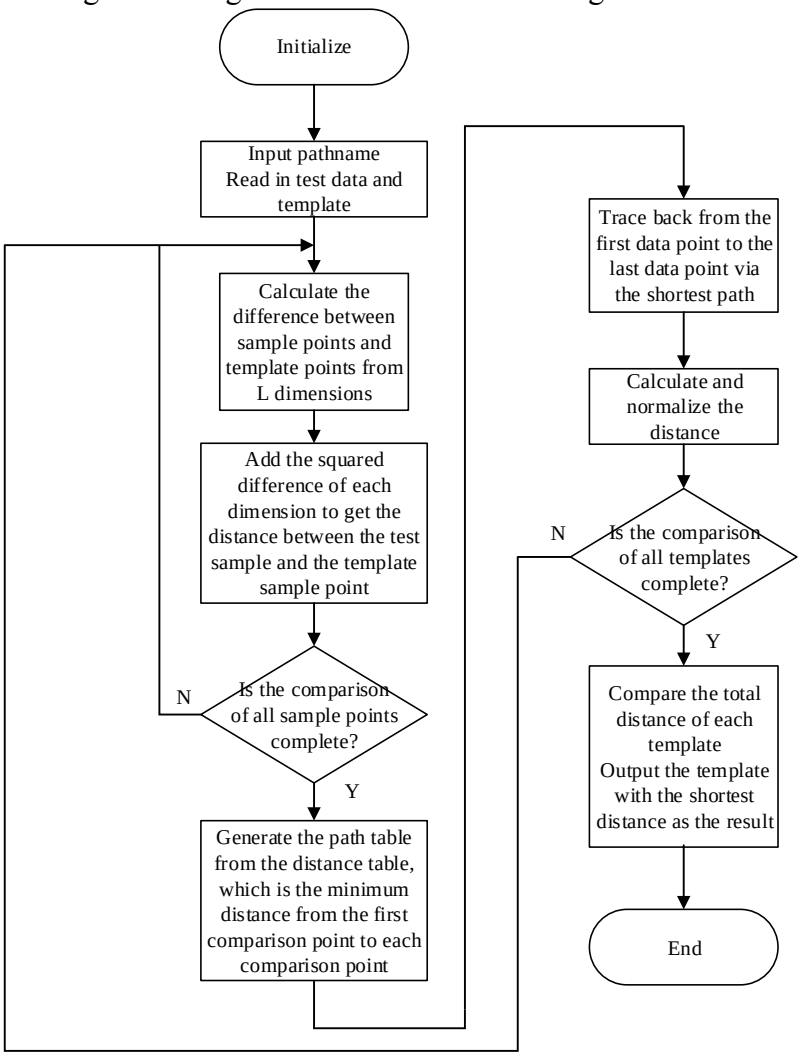

Figure 5.2.1: The real-time decoding block diagram of DTW

It can be inferred from the block diagram that if the length of the template is n, then the exact value of complexity for the decoding program is $3 \times n^2 \times SampleDimension \times TemplateNumber \approx 6 \times 10^8$, whereas 3 is the number of operands that needs to be done in every regulation, n is the number of sample points in the sample, and the average of n is 190, SampleDimension is the number of the dimensions (17) of the data, and TemplateNumber is the total number of the templates, 335 in the paper.

In the test, the processing of 1683 sentences containing 323287 sample points took 1410.091588 seconds, with an average of 0.837844 seconds per sentence, 0.004410 seconds, or 4.410 milliseconds per sample.

5.3 Real-time Decoding for HMM

In the program, the HViterbi class is used for the real-time decoding of HMM and the external interface is as follows

-(HViterbi*)init;

-(void)initLattice:(double *)obs type:(NSUInteger)recognizeType;

-(void)updateLattice:(double*)obs sampleIndex:(NSUInteger)sampleIndex type:(NSUInteger)recognizeType;

-(NSString*)decode:(NSUInteger)sampleNumber;

-(NSMutableArray *)readMFC:(NSString*)fileName;

-(double *)getMaxVector;

-(double *)getMinVector;

-(void)dealloc;

The init function is used to initialize. The dealloc function is used to release the memory that is applied through malloc. The readMFC function is used to read the data from the file. The getMaxVector and getMinVector function are used to find the maximum value of the data in each dimension, which is then used by the neural network after normalization.

To speed up decoding, the paper defines and uses a structure called Lattice. The structure contains the name of grid, serial number of the sign language word, the status number, observation density of the current grid, the linked list to the next grid (containing the serial number of the next possible grid and the transfer matrix and the most likely path number from the previous grid point to the current grid point. The definition is as follows:

```
struct latticeNode{
   double logTransP;
   unsigned int latticeIndex;
   struct latticeNode *next;
};
typedef struct latticeNode LatticeNode;
struct latticeList{
   LatticeNode *head;
};
typedef struct latticeList LatticeList;
```

```
struct lattice{
   char *latticeName;
   char *word;
   unsigned int wordIndex;
   unsigned int stateIndex;
   double prob;
   LatticeList *nextLatticeListP;
   bool used;
   bool nextUsed;
   double currentMaxProb;
   int maxLatticeIndex;
};
typedef struct lattice Lattice;
```

The paper also defines the array of structures used to hold all of the Lattice structures, and all subsequent operations are performed on the structure and array. After initializing the Lattice structure through the initLattice function, the program calculates the probability of each Lattice point reaching the next Lattice by the updateLattice function, and then goes back through the decode function to find the best path.

The block diagram of the decoding program is shown in figure 5.3.1.

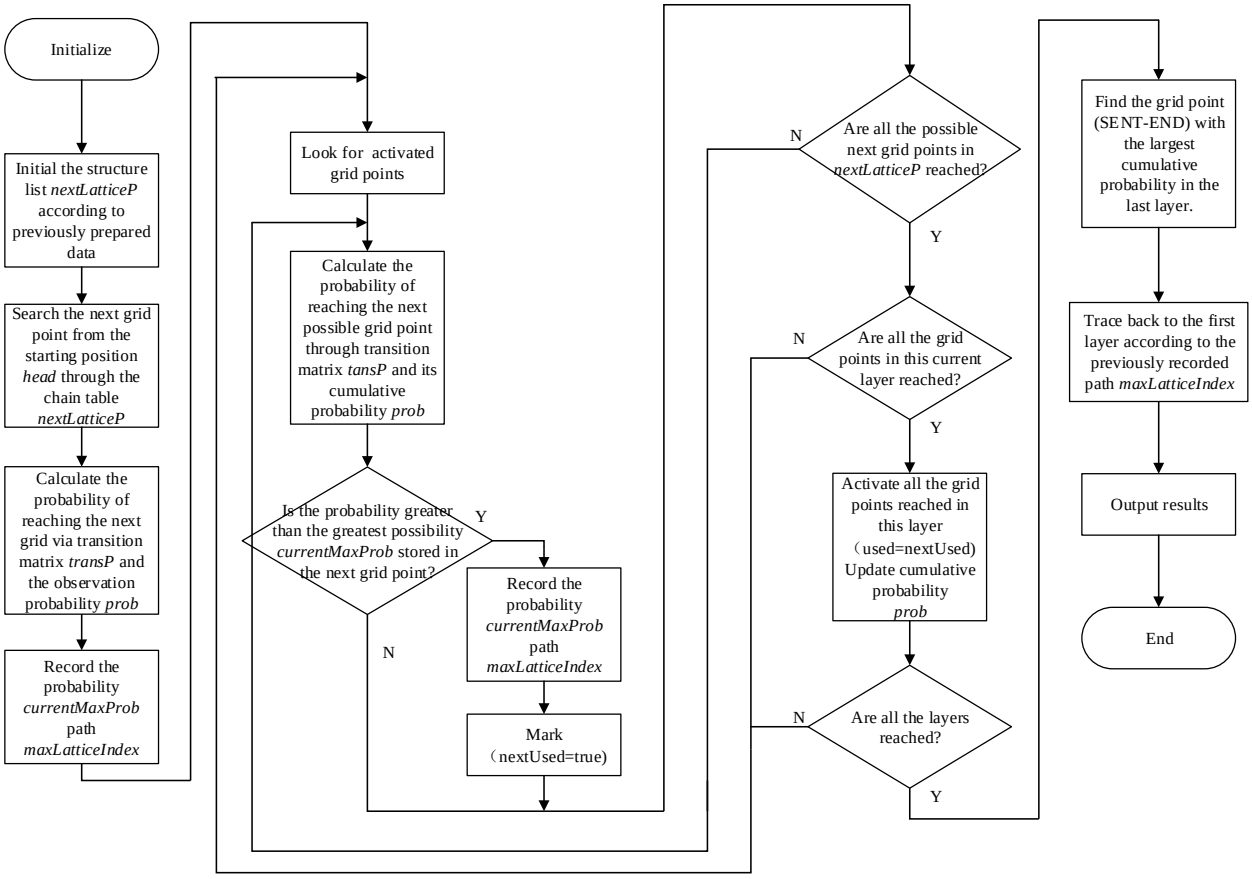

Figure 5.3.1 The block diagram of the decoding program for HMM

The main difference between DNN-HMM and GMM-HMM is the calculation of the observation density of the current lattice. The calculation method of DNN-HMM is based on *weight* and *bias* in the training results. In the program, the relu and softmax function are run to calculate the specific values of the observed probability. The calculation method of GMM-HMM is based on the Mean, Variance and the Gaussian model constant (Gconst). The observation density is calculated by using formula (8).

$$\mathrm{N}(\mathrm{o};\mu,\Sigma) = \frac{e^{-(o-\mu)'\Sigma^{-1}(o-\mu)}}{\sqrt{(2\pi)^n|\Sigma|}} \quad (8)$$

The core code of GMM-HMM is as follows

```
double *temp=(double *)malloc(sizeof(double)*_vectorLength);
    double *temp1=(double *)malloc(sizeof(double)*_vectorLength);
```

```
    for(int i=0;i<_stateNumber;++i){
        int meanStartIndex=i*(_vectorLength*2+1);
        int varStartIndex=meanStartIndex+_vectorLength;
        int gConstIndex=varStartIndex+_vectorLength;
        _stateProb[i]=0;
        vDSP_vsubD(data, 1, _stateArray+meanStartIndex, 1, temp, 1, _vectorLength);
        vDSP_vsqD(temp, 1, temp1, 1, _vectorLength);
        vDSP_dotprD(temp1, 1, _stateArray+varStartIndex, 1, _stateProb+i, _vectorLength);
        _stateProb[i]*=-0.5;
        _stateProb[i]-=0.5*_stateArray[gConstIndex];
    }
    free(temp);
    free(temp1);
```

The core code of DNN-HMM is as follows

```
double *input=data;
    double zero=0;
    for(int i=0;i<_layerNumber-1;++i){
        double *result=(double *)malloc(sizeof(double)*_row[i]);
        double *temp=(double *)malloc(sizeof(double)*_row[i]);
        vDSP_mmulD(_weight[i], 1, input, 1, result, 1, _row[i], 1, _column[i]);
        vDSP_vaddD(result, 1, _bias[i], 1, temp, 1, _row[i]);
        vDSP_vthrD(temp, 1, &zero, result, 1, _row[i]);
        free(temp);
        if(input!=data)
            free(input);
        input=result;
    }
    double *temp=(double *)malloc(sizeof(double)*_row[_layerNumber-1]);
```

```
vDSP_mmulD(_weight[_layerNumber-1], 1, input, 1, _stateProb, 1, _row[_layerNumber-
1], 1, _column[_layerNumber-1]);
vDSP_vaddD(_stateProb, 1, _bias[_layerNumber-1], 1, temp, 1, _row[_layerNumber-1]);

double sumProb=0;
for(int j=0;j<_row[_layerNumber-1];++j){
   sumProb+=exp(temp[j]);
}
sumProb=-log(sumProb);
vDSP_vsaddD(temp, 1, &sumProb, _stateProb, 1, _row[_layerNumber-1]);
free(input);
free(temp);
```

The Shared core code is as follows

```
for(int i=0;i<_latticeNumber;++i){
    if(_latticeArray[i].used){
        LatticeNode *nextLatticeP=_latticeArray[i].nextLatticeListP->head;
        while(nextLatticeP){
            Lattice *nextLattice=_latticeArray+nextLatticeP->latticeIndex;
            if(nextLattice->currentMaxProb<_latticeArray[i].prob+nextLatticeP-
>logTransP){
                nextLattice->nextUsed=true;
                nextLattice->currentMaxProb=_latticeArray[i].prob+nextLatticeP->logTransP;
                nextLattice->maxLatticeIndex=i;
            }
            nextLatticeP=nextLatticeP->next;
        }
    }
}
```

Suppose the length of the template is n, then the complexity of the GMM-HMM decoding program is $n \times (6 \times \frac{List}{Lattice} \times Lattice + StateNumber \times VectorLength \times 3) \approx 2.3 \times 10^7$,

Whereas 6 and 3 are respectively the number of commands for updating the lattice data and calculating the probability of observation. $\frac{List}{Lattice}$ is the average number of states that a lattice can reach, namely$\frac{10831}{5247}$. $Lattice$ is the number of the grids in total, namely 5247. $StateNumber$ is the number of states in total, namely 1182. $VectorLength$ is the dimension of the data, namely 17. n is the number of sample points in the sample and the average value is 190. The complexity of DNN-HMM decoding program mainly focuses on the calculation of observation probability by using DNN training parameters and the complexity of the shared core code is negligible. The specific value is $n \times (1500 \times 1500 \times 2 + 1182 \times 1500 \times 2) \approx 1.5 \times 10^9$, whereas 1500 and 1182 stands for the number of neurons in the second layer and in the third layer respectively.

In the test, the maximum time needed by GMM-HMM was 0.003427 seconds, or 3.427 milliseconds. The maximum time required for DNN-HMM was 0.025529 seconds, or 25.529 milliseconds, all within the specified design requirement of 50 milliseconds.

5.4 Real-time Decoding Summary

The algorithm complexity and processing time of DTW, GMM-HMM and DNN-HMM are compared in table 5.4.1。

Algorithm Name	Algorithm Complexity (number of commands per sentence)	Processing Time (millisecond per frame)
DTW	6×10^8	4.410 (on average)
GMM-HMM	2.3×10^7	3.427
DNN-HMM	1.5×10^9	25.529

Table 5.4.1 Algorithm complexity and processing time table of different algorithms

It can be seen from the table that the GMM-HMM and DNN-HMM algorithm can complete the processing of one frame of data within the prescribed 50 milliseconds, and the processing speed of the DTW algorithm is also within the expected time. All three algorithms can be used in real-time sign language recognition in this paper.

The advantage of the DTW algorithm is that the training samples are small and the recognition accuracy is high. The disadvantages are that the algorithm is high in complexity and cannot process the data in real time, and the malleability is poor. The prerequisite for identification using DTW is that the identified data must be fully aligned with the template, otherwise the recognition effect will be compromised. The advantage of the GMM-HMM algorithm is that the algorithm is low in complexity and high in malleability. The disadvantage is that the training sample is large, and the recognition accuracy is very low in the absence of syntax (as mentioned in 4.5). DNN-HMM is characterized by its ductility and good recognition accuracy in situations with or without syntax. In the future, the trend of sign recognition will be the larger vocabulary and the less syntax, and for this reason, DNN-HMM has a brighter development prospect than DTW and GMM-HMM.

Chapter 6 Conclusion and Outlooks

6.1 Conclusion

1. The project reaches the goal of designing and producing the low-cost gloves for sign language using common low cost components, such as Bluetooth module, gyroscope and flex sensors, at a total cost of only 782 yuan, which is far less than currently available data gloves with similar functions.
2. In the models involved in this thesis, The 17 dimensional data (without the left hand flex sensors) can replace the 22 dimensional data (with the left hand flex sensors) in model training and real-time recognition, and will not affect the final accuracy results. This not only reduces the complexity of the algorithm, but also reduces the cost.
3. The DNN - HMM hybrid model can achieve a higher recognition accuracy than then GMM - HMM model, and the difference is greater in the context-free cases, which conforms to the trend of the development of sign language recognition, and means that the DNN-HMM hybrid model is a more promising method for future sign language recognition.
4. The purpose of real-time recognition on an intelligent terminal is realized. The real-time decoding system can complete the data processing within the specified time, which guarantees the real-time performance.

6.2 Outlooks

There are still many shortcomings in the project. Afterwards, the project can advance in these following aspects.

1. Use a better algorithm. Due to the limited time, the paper did not study the tri-phonemes DNN-HMM, which may have better results. The paper only focused on the Multi-layer Perceptron (MLP). New technology in speech recognition, such as the convolutional neural network (CNN) and the recurrent neural network (RNN) can be used as a substitute for MLP.
2. Dig into the context-free situation. In the paper only the context-free mono-phonemes HMM model is studied. Clustering methods such as SOM or kmeans can be used in the case of tri-phonemes.
3. Improve the circuit board interface. In the project, jumper wires are used to connect the flex sensors to the circuit board, but this connection is extremely vulnerable, and the wires disconnect frequently during the tests. A possible solution is to find a smaller surface mounting connector to connect the flex sensors to the board, which will make the connection more stable.

Chapter 7 Reference

[1] 张良国，陈熙霖.手语识别研究综述[J].信息技术快报.2009，7(3):28-41

[2] 范会敏，王浩.模式识别方法概述[D].西安.西安工业大学计算机科学与工程学院，2012

[3] 江勇军.基于 Kinect 的孤立词手语识别系统研究[D].合肥.中国科学技术大，2015:9-14

[4] 余晓婷，贺荟中.国内手语研究综述[J].中国特殊教育，2009，4：36-41

[5] 陈振华，余永权，张瑞.模糊模式识别的几种基本模型研究[J].计算机技术与发展，2010，20(9)：32-35

[6] 吴江琴，高文.基于 ANN/HMM 的中国手语识别系统[J].计算机工程与应用，1999，37(9):1-4

[7] 倪训博，赵德斌，姜峰，程丹松. Viterbi 和 DTW 算法的关系分析——在非特定人手语识别中的应用[J].计算机研究与发展，2010，47(2):305-317

[8] 张露.基于 DTW 的单个手语识别算法[J].现代计算机，2016(8):77-80

[9] Sakoe H, Chiba S. Dynamic programming algorithm optimization for spoken word recognition[J]. IEEE Transactions on Acoustics Speech & Signal Processing, 1978, 26(1):43-49.

[10] Rabiner L R. A tutorial on hidden Markov models and selected applications in speech recognition[J]. Proceedings of the IEEE, 1989, 77(2):267-296.

[11] Young S, EvermDNN G, Gales M, et al. The HTK book (for HTK version 3.4)[J]. 2006.

[12] Fels.S.S.,Hinton.G.E.,GloveTalk:A neural network interface between a dataglove and a speech synthesizer[J], IEEE Trans On Neural Network，1993，4(1)：2—8.

[13] 邹伟. 中国手语单手词汇识别方法和技术研究[D]. 中国科学院自动化研究所, 2003.

[14] Michael A. Nielsen. "Neural Networks and Deep Learning" [M].Determination Press. 2014

Chapter 8 Acknowledgements

The project was completed under the guidance of my supervisor. In the planning phase of this project, he gave me a lot of advice on how to implement the project. He was very concerned about our research process and tried his best to create the best conditions for us throughout the project.

I would also like pay thanks to my family and friends. I have come across a lot of difficulties in the project and it is for their support and love that I can successfully complete it.