論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Deep Regression Model-based Real-Time Control Strategy for Myoelectric Prosthetic Hand
著者(和文)	秦梓軒
Author(English)	Zixuan Qin
出典(和文)	学位:博士(学術), 学位授与機関:東京工業大学, 報告番号:甲第12480号, 授与年月日:2023年3月26日, 学位の種別:課程博士, 審査員:小池 康晴,金子 寛彦,SLAVAKIS KONSTANTINOS,小尾 高史,八木 透
Citation(English)	Degree:Doctor (Academic), Conferring organization: Tokyo Institute of Technology, Report number:甲第12480号, Conferred date:2023/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

東京工業大学
Tokyo Institute of Technology

School of Engineering
Department of Information and Communications Engineering

Deep Regression Model-based Real-Time Control Strategy for Myoelectric Prosthetic Hand

Zixuan Qin

Thesis submitted to the Tokyo Institute of Technology for the degree of
Doctor of Philosophy (*Ph.D.*)

March, 2023

Zixuan Qin. (2023). *Deep Regression Model-based Real-Time Control Strategy for Myoelectric Prosthetic Hand.* Doctor of Philosophy (Ph.D.). Tokyo Institute of Technology.

Supervisors
Prof. Yasuharu Koike
Prof. Natsue Yoshimura
Prof. Kaneko Hirohiko
Prof. Kostas Slavakis
Prof. Yagi Tohru
Prof. Obi Takashi

Location:
Tokyo, Japan

Abstract

Upper limb amputees will lose their ability to work and even have trouble maintaining a normal life. To support the daily life for amputees, myoelectric prosthetic hands which have ability to perform human-hand motions using surface electromyography (sEMG) signals were developed. Using sEMG signal as the input to obtain the motor control command can be regarded as the control strategy for myoelectric prothesis. However, even until now, it is still a challenging task to propose a control strategy or system with high accuracy and natural performance. Many studies are considering applying deep learning for forearm motion classification or regression prediction. However, most of the studies were limited to offline analysis, few of them can be applied to real-time control systems. Since the sEMG signals quality and electrode positions will change on different days, even for the same person, using the trained model directly will decrease the model accuracy. Moreover, until now there is no research has proposed real-time control strategies or systems based on deep regression models. In this thesis, we proposed a deep regression model for multi-joint angles estimation with three degrees of freedom, and evaluated the offline experiment in different days. We updated the model parameters via transfer learning in another days. Furthermore, we applied the proposed regression model to build a novel real-time control system for prosthetic hand. We evaluated the system by analyzing real-time estimation accuracy and computational latency. As a demonstration of the successful control, we controlled a virtual hand in real-time via the proposed control strategy.

Key words: deep learning, multi-joint angles estimation, myoelectric prosthetic hand, real-time control system, regression model, surface electromyography (sEMG), transfer learning.

Acknowledgements

First of all, I would thank my parents for their continued support and encouragement for my doctoral studies. I am also very grateful to my fiancée, Ms Yangyuting Zhang, for her company, support and care. When I entered *Ph.D.* course, sometimes I doubted my ability to pursue a *Ph.D.* degree. Without my family's expectations and support, it would have been difficult for me to complete *Ph.D.* course.

Then, I would like to express my sincere gratitude to my academic supervisor, dear Professor Koike Yasuharu, for his careful guidance and helpful suggestions to my research and paper publication. I think he has played a vital role in my growth, he is a really good teacher and successful educator, and I will never forget his guidance, help and inspiration in my life. And I would like to thank Professor Yoshimura Natsue for teaching me in data analysis skills.

Finally, I would like to thank all friends who helped me a lot during my research. Especially Zhaolun Zou, Yuanhao Li, Zixun He, Woorim Cho, Sorawit Stapornchaisit, Shunki Kitsunai, Chihiro Sano and so on. Without your help and company, it would have been difficult for me alone to get the achievement today. Moreover, I should thank all the subjects who participated in my experiments.

I also thank to "Cross the border! Tokyo-Tech Pioneering Doctor Research Program" (JST SPRING, grant no. JPMJSP2106) for providing me with financial support (including living expenses and scientific research funds) during my *Ph.D.* research.

List of Publications

1. **Qin Z**, He Z, Li Y, Saetia S and Koike Y. (2022). *A CW-CNN regression model-based real-time system for virtual hand control.* Front. Neurorobot. 16:1072365. doi: 10.3389/fnbot.2022.1072365 [**Chapter 4**]

2. **Qin Z**, Stapornchaisit S, He Z, Yoshimura N and Koike Y. (2021). *Multi–Joint Angles Estimation of Forearm Motion Using a Regression Model.* Front. Neurorobot. 15:685961. doi: 10.3389/fnbot.2021.685961 [**Chapter 3**]

3. **Z Qin**, S Stapornchaisit, Z He, N Yoshimura and Y Koike. *A channel-wise CNN Model with Transfer Learning for Forearm Motion Estimation using sEMG signal.* in 2022 SICE-DAS symposium (online), 2C1-3. January 21st—22nd, 2022. Access: 34th SICE DAS [**Chapter 3**]

4. Y Koike, Y Kim, S Stapornchaisit, **Z Qin**, T Kawase, and N Yoshimura, *Development of Multi-sensor Array Electrodes for Measurement of Deeper Muscle Activation*, Sens. Mater., Vol. 32, No. 3, 2020, p. 959-966. doi: 10.18494/SAM.2020.2636 [**Chapter 3**]

5. He Z, **Qin Z**, Koike Y. *Continuous Estimation of Finger and Wrist Joint Angles Using a Muscle Synergy Based Musculoskeletal Model.* Applied Sciences. 2022; 12(8):3772. doi: 10.3390/app12083772

6. Shunki Kitsunai, Woorim Cho, Chihiro Sano, Supat Saetia, **Zixuan Qin**, Yasuharu Koike, Mattia Frasca, Natsue Yoshimura, and Ludovico Minati. *Generation of diverse insect-like gait patterns using networks of coupled Rössler systems.* Chaos 30, 123132 (2020) doi: 10.1063/5.0021694

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Human hands play very important role in activities of daily living (ADL), the upper limb amputations significantly cause inconvenience to human life and work, which impact functional independence. According to a survey in 2011, there are approximately 8000 people lost their upper limb in Japan; and in the United States, there are approximately 1 of every 200 people are living with their lost upper limb [1–3]. After the treatment, in order to work or return to normal ADL, the upper limb amputees prefer to use prosthetic hands, which can imitate human motions to complete their activities. However, according to a questionnaire about the frequency and satisfaction of wearing functional prostheses, 64% of the upper limb amputees rated their devices as "not acceptable" and "fair", 56% of them used their prostheses in daily life "once in a while" or "never" [4]. Therefore, improving the evaluation and dependence on functional prosthetic hands in daily life is particularly important for upper limb amputees to return to normal life, and this will be the goal that disciplines in related field need to work on.

Nowadays, researchers have considered to apply surface electromyography (sEMG) signals on prosthetic hand control, and the challenge is how to estimate the human forearm motion based on sEMG signals, and control the corresponding degrees of freedom (DOFs) of the robot in real-time. Many strategies were proposed to solve the problem of motion estimation, however,

many of them were not discussed why the motion information from sEMG signal can be captured by their methods, and how to solve the problem that the signal quality will change in another days. Moreover, most of the strategies were limited to offline analysis, without discussion of the possibility of being able to build a complete real-time control system.

1.1 Myoelectric Prosthetic Hand

There are mainly three types of prosthetic hand available [5] to upper limb amputees:

- Cosmetic: shown as Figure 1.1(a). Passive prosthetic devices, because this kind of prosthetic hand does not have active motion, remain non-functional, unlike both body-powered and myoelectric devices. Amputees can use them for appearance purposes, or perform some movements such as pushing open a door or steadying a piece of paper during writing [5–7].

- Body-powered: shown as Figure 1.1(b). Functional prostheses, can switch between opens and closes by operating cable, which is powered using human muscle movements. This kind prostheses have advantages in durability, feedback, adjustment and maintenance [5, 8, 9].

- Electronic: shown as Figure 1.1(c). With motors and microprocessors, amputees can control the prosthetic hand through active muscle contraction.

Apparently, without control modules, cosmetic prosthetic hand only has the appearance of human hands. Body-powered prosthetic hand can only perform 1-DOF movement, and fails to achieve flexible motion control. With more motors and intelligent microprocessors, electronic prostheses have possibilities to complete flexible human hand motion control. The sEMG-based electronic

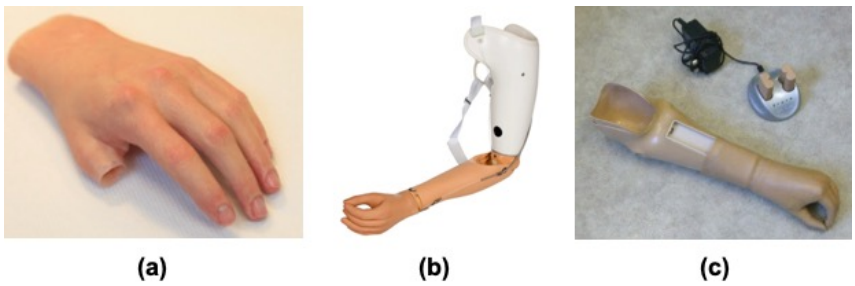


Figure 1.1: *Prosthetic hand. (a) Cosmetic prosthesis [7]; (b) Body-powered prosthesis [9]; (c) Electronic prosthesis [5].*

prostheses named myoelectric prostheses, they were introduced in about 1960, and widely used in rehabilitation for upper limb amputees [5, 10, 11].

There are several commercial prosthetic hands have been released based on sEMG control, such as Vincent hand, i-Limb, Bebionic hand, Michelongelo hand, SoftHand Pro, and even the open source 3D hand [12–18], as shown in Figure 1.2.

However, as we know, the control strategies of those myoelectric prosthesis are rudimentary system, which fail to achieve natural movement control performance and require complicated training procedures [19]. The control methods of the above commercial myoelectric prosthesis not only limit the ADL qual-

Figure 1.2: *Commercial myoelectric prosthetic hands. (a) Vincent hand [12]; (b) i-Limb [13]; (c) Bebionic hand [14]; (d) Michelongelo hand [15]; (e) SoftHand Pro [16]; (f) Open source 3D hand [17].*

ity, but also cause the low acceptance. Therefore, it is important to propose a novel real-time control system with high-accuracy motion estimation model based on sEMG signals which has ability to achieve natural performance, and it is exactly the aim this study hope to achieve in this thesis.

1.2 Deep Learning-based Motion Recognition

1.2.1 Motion Recognition

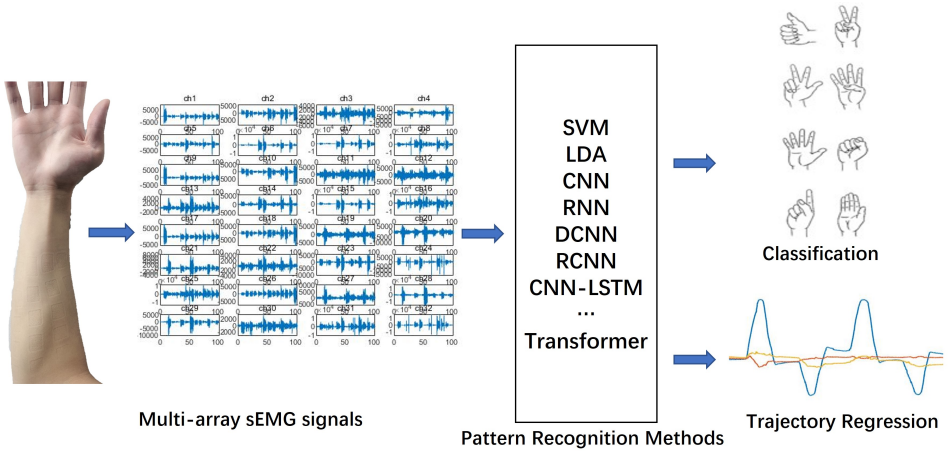


Figure 1.3: *Motion Recognition by sEMG signals.*

Motion pattern recognition methods, including motion classification and motion trajectory regression prediction from sEMG signals, are achieved by connecting sEMG signal patterns with specific motions [20]. Classification models are essentially the same as regression models, in that classification models discrete the output of models, and output of regression models is continuous. Figure 1.3 shows the motion recognition process by raw sEMG signals obtained from human forearm. Motion classification is to tag motion or gesture labels to the corresponding sEMG signals, while motion regression is to label the trajectory (continuous positions or angles) of the motions to sEMG signals. Nowadays, more and more researchers investigated machine learning methods to solve the tasks of using sEMG signals for motion recognition and prosthetic hand control, such as traditional methods-based classifier like support vector

machine (SVM) [21–24] or linear discriminant analysis (LDA) [23, 25, 26], and even deep learning methods. Deep learning is end-to-end learning, which is easy to use, and compared to traditional machine learning, it has more learning capability and can mine the potential features of data.

1.2.2 Related Deep Learning Approaches

As the mainstream deep learning method that is widely used in the field of image recognition, convolutional neural network (CNN) is also mainly used for hand and forearm motion classification or trajectory regression prediction [20, 27–30]. Recurrent neural networks (RNN) is also a kind of artificial neural networks where connections between nodes form a directed graph along a temporal sequence, so that applied to tasks such as speech recognition and bio-signal pattern recognition [31–33]. In addition, some research proposed models to combine CNN with another deep learning models, such as RCNN which consisted of CNN and RNN [34–36], or even CNN-LSTM model which consisted of CNN and long short-term memory (LSTM) [37, 38]. Moreover, new technologies such as transformer [39] were also applied for motion recognition.

Compared to motion classification, real-time regression prediction is more challenging. In this study, I used 32 channels of sEMG signals to approach joint angles regression, the sEMG signals can be regarded as two-dimensional (2D) image within a sliding window (time×channel). Thus, I considered to designed a CNN-based regression model (Section 3.2.4) to predict 3-DOFs angles. To prove the proposed CNN model is superior, it was compared with conventional regression models, and was evaluated in both offline and real-time environment. Moreover, sEMG quality of human changes everyday, in order to keep the robustness of model performance in different days, transfer learning strategies like fine-tuning, layer transfer, *etc.* [30, 40–43] need to be used for updating the model parameters.

1.3 Myoelectric Prosthesis Real-Time Control

Until now, several researchers have presented real-time control systems for myoelectric prosthesis, *e.g.*, Furui *et al.* [44] proposed synergy-based control system in 2019, and de Oliveira de Souza *et al.* [45] proposed multilayer perception classifier-based control system in 2021. Even though many papers attempted to recognize motions *via* deep learning methods (Section 1.2.2), they were limited to offline analysis, only a few recent papers applied the deep learning approaches on real-time control system. Jafarzadeh *et al.* [46] built a CNN classifier-based control system in 2019, the training and validation accuracy were high (99.98% and 91.26% respectively), however, testing accuracy was only 48.40%. In 2020, Tam *et al.* [47] presented a CNN-based real-time gesture recognition system, the classification accuracy was 98.15% with 100—200 ms latency. However, neither of them completed the demonstration videos. In addition, we can find that almost all of the related researches were CNN-based classification system, and hard to find the papers applied deep learning regression model in real-time control system.

For the reasons above, this thesis constructed the real-time control system for myoelectric prosthesis with our regression model. This thesis should be the few studies that has successfully applied CNN regression models for real-time control. For a complete real-time control system, only the proposed model is not enough, modules such as data acquisition, signal preprocessing (real-time) and output filter are also required. Moreover, the control system carrier also needs to be determined, *e.g.*, the model and modules are to be carried on a fully embedded chip [47], or controlled via a PC with graphical user interface (GUI) as described in Section 4.2. To evaluate a real-time control system, both recognition accuracy and latency are necessary.

Chapter 2

Technical Background and Aims

2.1 Surface Electromyography (sEMG) Signal

2.1.1 Overview

The electromyography (EMG) signal is a temporal and spatial superposition of motor unit action potentials (MUAP) in numerous muscle fibers. Figure 2.1 demonstrates the process of EMG generation and how muscle generates contraction and relaxation. After our brain sends motor commands, the central nervous system (CNS) generates electrical impulses through synaptic stimulation and travels along the axon of the neuron to the point of contact between the peripheral nerve and the muscle. When the motor nerve contacts the muscle, the axon will branch to the muscle fiber (which called terminal axonal branches), and the released acetylcholine causes the synapse to generate a potential, which in turn generates a motor potential in the muscle fiber and transmits along the muscle fiber to both directions [48–50].

The surface electromyography (sEMG) signal is the hybrid effect of the electrical activity on the nerve and superficial muscle EMG on skin surface, which can reflect the neuromuscular characteristics of the whole muscle. The sEMG signal is a non-invasive, commonly used and safe method of data acquisition. The amplitude of sEMG generally in the range of 0.01—10 mV, the useful sig-

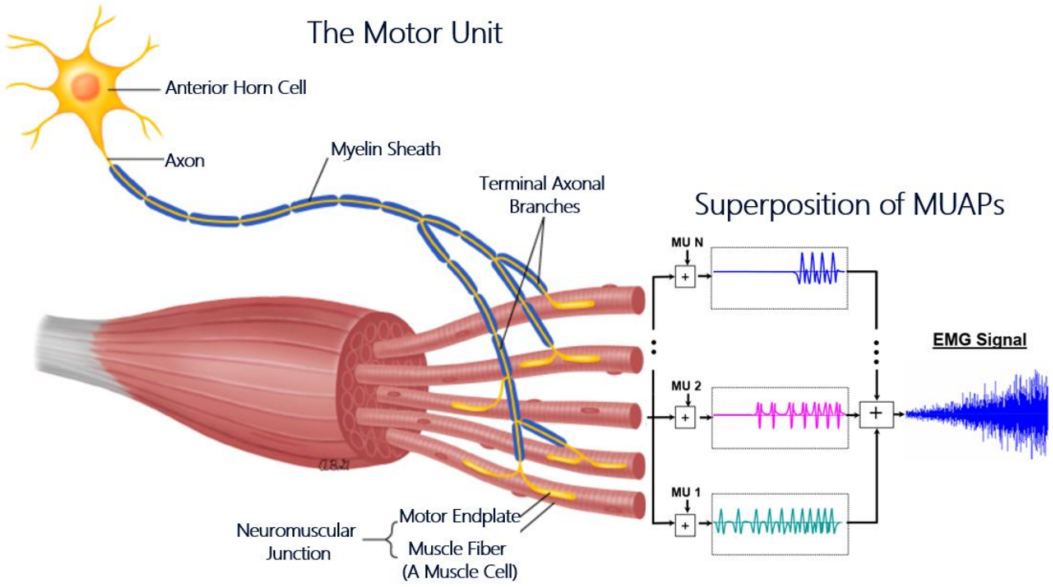


Figure 2.1: *How EMG signals are generated [49].*

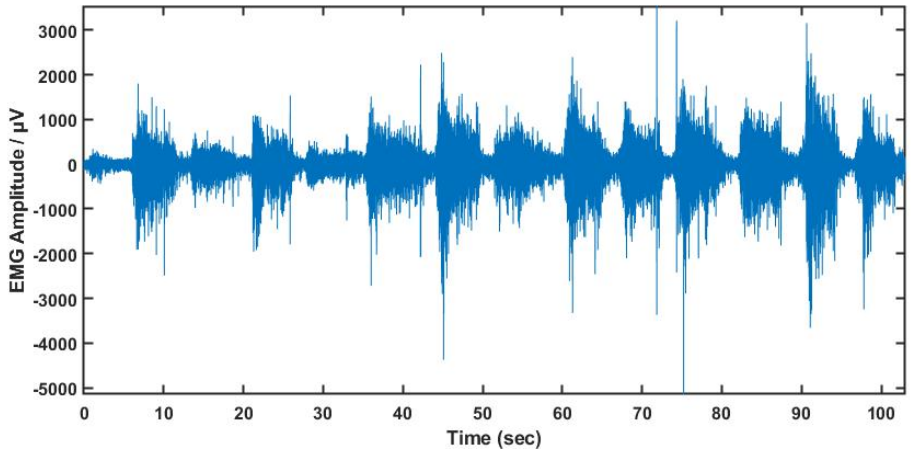


Figure 2.2: *Example of sEMG signal acquired by multi-array electrode (SMK Corp., SEIREN Co., Ltd.).*

nal frequency is located in 0—500 Hz, and the main energy is concentrated in 20—150 Hz [48, 51]. Figure 2.2 shows an example of sEMG signal acquired in the study of this thesis. Researchers have investigated biosignal-based methods to control prosthetic hands, such as sEMG and eletroencephalography (EEG). Compared to EEG signals, sEMG signals which can directly utilize neural pathways in humans to recovery ADL function, have higher reliability and accuracy [36, 52, 53].

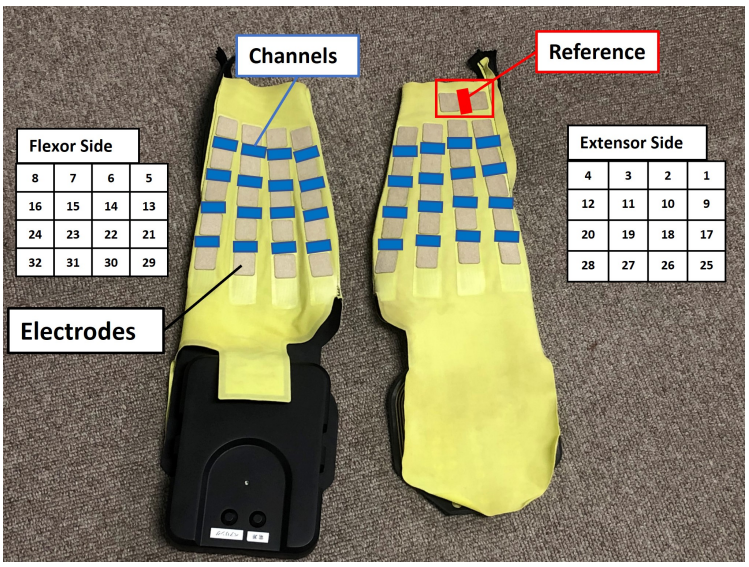


Figure 2.3: *Bipolar multi-array electrodes (32 channels, right hand) for sEMG signal acquisition. The red part indicates the reference point which shared by two electrodes, the blue part is the channel that shared by upper and lower electrodes. The channel matrixes on both flexor and extensor side are given, and the number shows the channel number. Data transmission method is from the black box (signal transceiver) to the PC via Bluetooth.*

2.1.2 Bipolar Multi-array Electrode

Bipolar multi-array electrode with 32 channels (SMK Corp., SEIREN Co., Ltd.) was used to acquire sEMG signals. The electrode sleeve and the signal transceiver are shown in Figure 2.3. The sleeve includes flexor side and extensor side.

There are 40 electrodes (5×8) in total, two electrodes share one channel (the blue part), and the top two electrodes share one reference (red part). Therefore, this multi-array electrode has 32 channels in total (4×4 in each side). Extensor side includes ch1—ch4, ch9—ch12, ch17—ch20 and ch25—ch28. Flexor side includes ch5—ch8, ch13—ch16, ch21—ch24 and ch29—ch32. The analog-to-digital converter (ADC) resolution of the electrode is 16 bit. The signal can be sent to PC by a Bluetooth module inside the signal transceiver. The sampling frequency of the bipolar multi-array electrode is 500 Hz [54] due to the limitation of Bluetooth signal transmission speed, and I do not have to adjust the electrode location for each participant with this device.

2.1.3 Signal Pre-processing

The sEMG signal analysis can be widely applied on functional evaluation of rehabilitation status, fatigue identification, motion pattern research and myoelectric prosthesis control. In this thesis, I processed the measured sEMG signal to integrated EMG (IEMG) in both offline and real-time environment. This section will introduce the pre-processing steps.

The procedure of processing can be roughly divided into three steps: rectification, low-pass filtering and normalization.

Rectification

Rectification is to simply take the absolute value of the raw signal, also called *full wave rectification*, which is essential for getting the shape or envelope of sEMG signal. Full wave rectification of the sEMG signal provides the temporal pattern of grouped motor unit (MU) firing, and we do not need to focus on the shape of action potential (AP) [55].

Generally, the practice to simply low-pass filtering the un-rectified raw signal to capture the envelope fails, because sEMG signal oscillates rapidly around

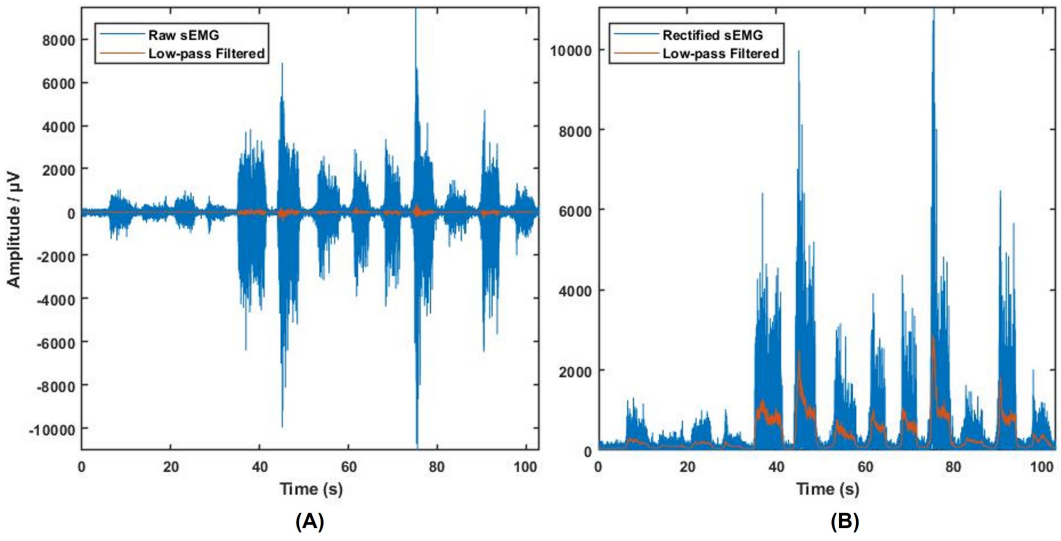


Figure 2.4: *Importance of Rectification in obtaining envelope of sEMG signal.* ***(A)*** *Low-pass filtering raw sEMG signal without rectification;* ***(B)*** *Low-pass filtering rectified sEMG signal.*

zero, if we smooth such signal by low-pass filter, we can only obtain zero, which useless (Figure 2.4(A)). On the contrary, if the rectified signal is low-pass filtered, the result looks like envolope of the raw signal (Figure 2.4(B)).

Low-pass filtering

It was found that a 2^{nd}-order low-pass filter was suitable to estimate the muscle force [56]. The function of the low-pass filter is to allow signals in the lower frequency band below the cutoff frequency f_c to pass through, and remove the signals above f_c. According to implementation methods, digital filters can be divided into finite impulse response (FIR) filter and infinite impulse response (IIR) filter. [57].

- **FIR filter** linear phase delay (definite value); faster operation speed; smaller error; output values without feedback and non-recursive according to difference equation:

$$y(n) = \sum_{k=0}^{N} a(k)x(n-k) \tag{2.1}$$

- **IIR filter** non-linear phase delay (phase delay varies with the input waveform); slower speed; output values with feedback (recursive) according to difference equation:

$$y(n) = \sum_{k=0}^{N} a(k)x(n-k) + \sum_{j=0}^{P} b(j)y(n-j) \tag{2.2}$$

In 1995, Koike *et al.* [58] defined an FIR filter to represent the relationship between EMG and quasi-tension (filtered-EMG):

$$\hat{T}(t) = \sum_{j=1}^{n} h_j \cdot EMG(t-j+1) \tag{2.3}$$

where j is discrete time, $\hat{T}$ is quasi-tension, h_j is the proposed filter. The

2^{nd}-order frequency response of the filter is:

$$H(s) = \frac{\omega_n^2}{s^2 + \zeta\omega_n s + \omega_n^2} \tag{2.4}$$

where ω_n and ζ are natural frequency and damping coefficient, respectively. Impulse response is the filter appears in time domain, filters typically have broad frequency response, which correspond to short duration pulse in the time domain. The impulse response of the Equation 2.4 is:

$$h(t) = a \times (e^{-bt} - e^{-ct}) \tag{2.5}$$

According to [58], the experimental results shown that the coefficients of resulting impulse response are $a = 6.44$, $b = 10.80$ and $c = 16.52$, respectively.

In this thesis, I filtered the raw sEMG signals obtained in all experiments *via* the FIR low-pass filter (with the coefficients a, b and c) proposed by Koike *et al.*.

Normalization

Although we can obtain IEMG signals by rectification and FIR low-pass filtering, the same person has different muscle activity and force in different days, even for different person. Therefore, we need to normalize the obtained IEMG signals. Normalization is to map the data to the specified range, *e.g.*, map the data to be processed within the range of 0 to 1 or -1 to 1. By this approach, the expressions with dimensions are turned into without dimensions, so that metrics of different units or magnitudes can be compared or weighted. After normalization, IEMG can be turned into a pure quantity, which can also simplify the calculation.

There are mainly two normalization methods are commonly used:

- Min-Max Normalization

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2.6}$$

- Z-Score Normalization

$$x_{norm} = \frac{x - \mu}{\sigma} \tag{2.7}$$

  where μ is mean value, σ is standard deviation (std).

In this thesis, Min-Max Normalization method was used to normalize the IEMG signal in the range of $[0, 1]$. In the offline experiments, when I built the dataset for model training, the IEMG signals was normalized based on minimum and maximum value of itself. In the real-time experiments, I acquired the maximum voluntary contraction (MVC) of each participant before the experiment, thus, the normalization of IEMG signal was based on the minimum and maximum value of MVC signals.

In summary, in this thesis, progress of raw sEMG signal pre-processing can be described as the following equations:

$$IEMG_i = filter(abs(EMG_i)) \tag{2.8}$$

$$IMVC_i = filter(abs(MVC_i)) \tag{2.9}$$

$$IEMG^i_{norm} = \frac{IEMG_i - min(IMVC_i)}{max(IMVC_i) - min(IMVC_i)} \tag{2.10}$$

where i is channel. $IMVC$ is integrated MVC, means to process MVC signals in the same way as IEMG signal, $IEMG_{norm}$ is the normalized IEMG signal. In offline expeirments, $IMVC_i$ should be replaced by $IEMG_i$ in Equation 2.10. The $abs()$ function means to calculate the absolute value of input series, so that obtain rectified sEMG signals; the $filter()$ is the FIR low-pass filter which proposed in [58], shown as Equation 2.3—2.5.

2.2 Upper Extremity Motions

Human upper extremity includes the forearm, wrist and hand. For wrist and hand motions, the activated muscles generated from forearm. This section will introduce the DOFs of the motions and muscle anatomy knowledge in terms of wrist and hand, respectively.

2.2.1 Wrist Motion

Wrist is an important entity which connects the human hand to forearm, and we called it wrist joint. The wrist joint has 3-DOFs, including flexion/extension, ulnar/radial deviation and pronation/supination (Figure 2.5).

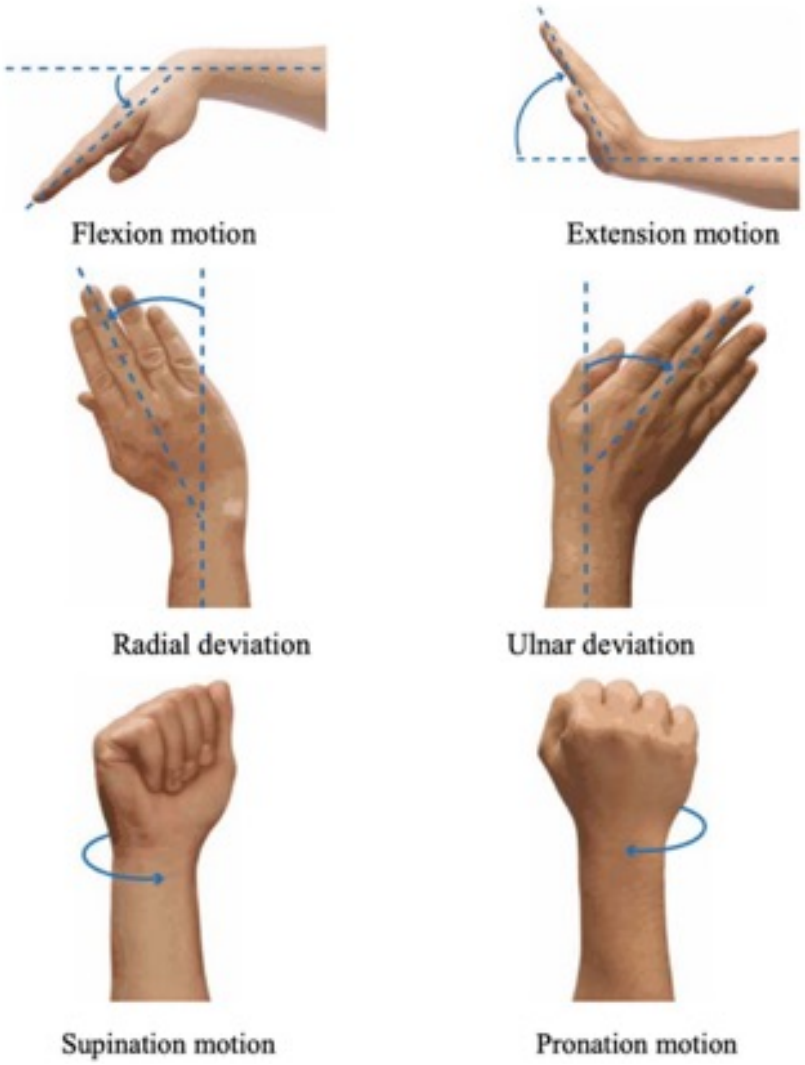


Figure 2.5: *Description of wrist motions [59]*

Wrist flexion is rotating the palm to the direction of anterior surface of the forearm, while wrist extension is the opposite direction. Radial deviation (or wrist abduction) is to bend up the wrist (towards thumb side), while ulnar deviation (or wrist adduction) is the opposite direction. Pronation is to rotate the whole forearm to a palm-down orientation, while supination is the opposite direction. The activated muscles corresponding to each motion are shown in Table 2.1.

Table 2.1: Activated muscles for wrist motions [59].

Motion	Muscles
Wrist flexion	Flexor carpi radialis Flexor carpi ulnaris Palmaris longus
Wrist extension	Extensor carpi radialis longus Extensor digitorum Extensor carpi ulnaris Anconeus
Wrist ulnar deviation	Flexor carpi ulnaris Extensor carpi ulnaris
Wrist radial deviation	Extensor carpi radialis longus Extensor carpi radialis brevis Flexor carpi radialis
Wrist pronation	Pronator quadratus Pronator teres
Wrist supination	Supinator Biceps brachii

2.2.2 Hand Motion

There are a total of 21 DOFs in human fingers: 4 for each finger (3 for flexion/extension, 1 for abduction/adduction), and 5 for thumb [60]. With so many DOFs, human can perform many gestures, such as the examples in Figure 2.6. For finger motions, the muscle activity is complex, because different fingers have different related activated muscles, and generally they are activated together to complete a gesture. In this thesis, I only focus on hand

grip/open motion, namely, all joints of the 5 fingers (thumb, index, middle, ring and pinky) perform simultaneously flexion during hand grip, and extension during hand open (f and g in Figure 2.6).

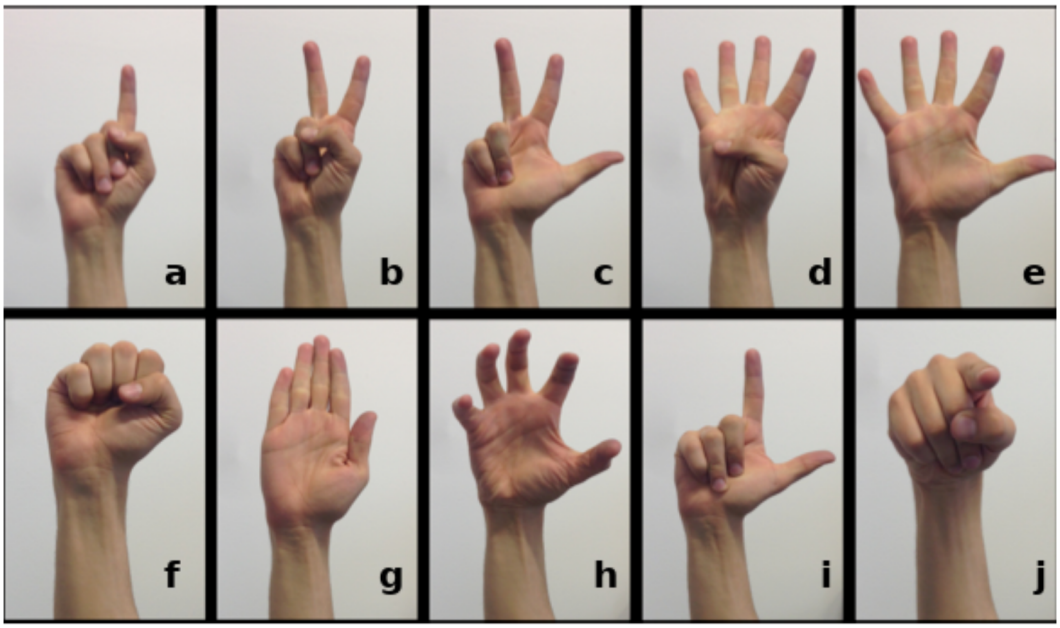


Figure 2.6: *Examples of hand motions [61]*

Even though there are many muscles located in human hand are related to hand motions, in this research I acquired sEMG signals from forearm muscle, thus, I only focus on the activated muscles in forearm. Some of the important activated muscles corresponding to hand grip and open are shown in Table 2.2. Among them, FPL, APL, EPL are related to thumb motion, ECU is related wrist flexion/extension.

Table 2.2: Activated muscles (forearm) for hand motions.

Motion	Muscles
Hand grip/open	Flexor digitorum superficialis (FDS) Flexor pollicis longus muscle (FPL) Extensor pollicis longus muscle (EPL) Abductor pollicis longus muscle (APL) Extensor carpi ulnaris (ECU) Flexor digitorum profundus (FDP) Extensor digitorum (ED)

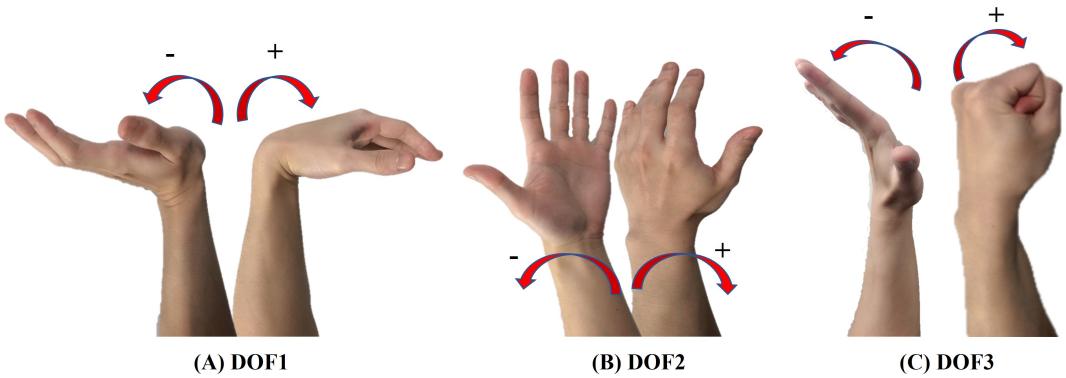


Figure 2.7: *Description of the three degrees of freedom (3-DOFs) in this thesis.* ***(A)*** *DOF1 is the joint to perform wrist flexion and wrist extension, wrist flexion angle is positive number, and wrist extension angle is negative number.* ***(B)*** *DOF2 is the joint to perform wrist pronation and wrist supination, wrist pronation angle is positive number, and wrist supination angle is negative angle.* ***(C)*** *DOF3 is the joint to perform wrist grip or return to rest motion, the angle hand grip shows positive number.*

2.2.3 Research Highlights

In this thesis, I focused on researching 3-DOFs of human upper extremity, including two wrist DOFs and one hand DOF. The two wrist DOFs are wrist flexion/extension (WF/WE) and wrist pronation/supination (WP/WS), respectively. The hand DOF is hand grip/open (HG/HO), which means all of the fingers bend to grip or open simultaneously. They were called DOF1, DOF2 and DOF3, respectively, as shown in Figure 2.7. Moreover, when I acquired joint angles by motion capture, I used positive angles to indicate WF, WP and HG motion, and negative angles to indicate WE, WS and HO motion. The daily motions evaluated in this research were related to the 3-DOFs, and the corresponding activated muscle areas were used to analyze motion patterns *via* geometry plot.

2.2.4 Perception Neuron Motion Capture System

I used the Perception Neuron Motion Capture System (Noitom Ltd., China) to acquire joint angles data during the experiment. This motion capture system can be used for movie makers, game developers, sports analysis, and biomechanics research [62]. Participants needed to wear the glove with mark-

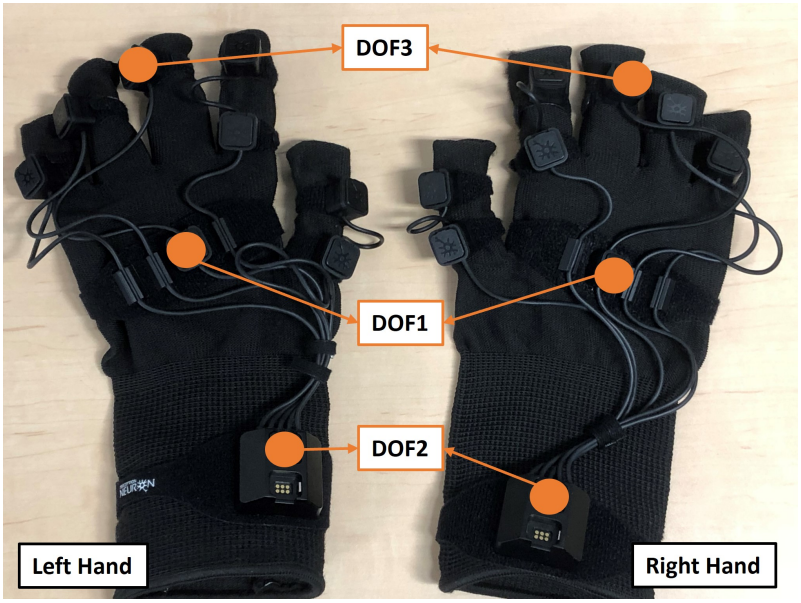


Figure 2.8: *Perception Neuron Motion Capture System glove (both hands) for joint angles collection. The orange markers can obtain the joint angles from different DOFs.*

ers (Figure 2.8) and completed the calibration to build a skeleton model on the Axis Neuron software. After a successful calibration, I could check the corresponding joint angles for 3-DOFs according to the specific marker (orange circle) [63] and axis of rotation (X, Y or Z axis).

I chose Bounding Volume Hierarchies (BVH) data type as the joint angle data format while using Axis Neuron. This data type includes joint hierarchy and motion data. The marker data of each joint are stored in the local coordination system, the body joints are arranged as tree structures, children joints are connected to their parent joint [64]. BVH data of each joint is calculated from children joint system to parent system by internally multiplying the transformation matrix, then the data can be shown in global coordinates.

The orange circles in Figure 2.8 indicate the three required markers to obtain corresponding 3-DOFs joint angles. According to Figure 2.9, for right hand glove, the name of the marker for collecting WF/WE (DOF1) joint is "RightHand", WP/WS (DOF2) joint is "RightForeArm", and HG/HO (DOF3) joint is "RightHandMiddle1". Similarly for the left hand, the name of the three markers are "LeftHand", "LeftForeArm" and "LeftHandMiddle1", respectively.

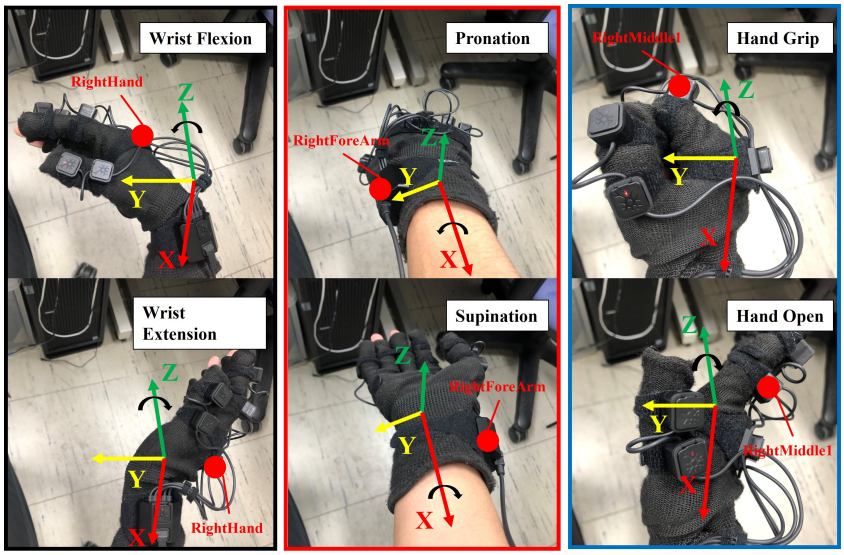


Figure 2.9: *Illustration of Data Collection via Perception Neuron Motion Capture (right hand). The red points shows the focus markers corresponding to the 3-DOFs respectively, the coordinate here shows the BVH global coordinates, red axis is X axis, yellow axis is Y axis, and green axis is Z axis. Counterclockwise direction shows positive angle, clockwise direction indicates negative angel.*

To independently check whether the angle data of a certain joint is as our expectation, we need to know the name of the marker and the rotation axis corresponding to that degree of freedom. According to coordinate document which provided by Aiuto Co. Ltd., Japan and Noitom Ltd. [65], the BVH global coordinates are shown in Figure 2.9, the red, yellow and green axis are X-axis, Y-axis and Z-axis, respectively. Therefore, the joint angle of WF/WE motion can be described as the "RightHand" (or "LeftHand") marker rotating around Z-axis, WP/WS can be described as "RightForeArm" (or "LeftForeArm") marker rotating around X-axis, and HG/HO is "RightHandMiddle1" (or "LeftHandMiddle1") marker rotating around Z-axis. For right hand, the counterclockwise direction denotes the positive rotation, i.e. WF, WP and HG are positive angles; the clockwise direction denotes the negative rotation, i.e. WE, WS and HO are negative angles. For the left hand glove, it is the opposite situation.

The sampling frequency of the Perception Neuron Motion Capture System is 120 Hz. The data was obtained with raw sEMG signal simultaneously *via* lab streaming layer (LSL), and synchronized as dataset (Section 3.2.3).

2.3 Convolutional Neural Network (CNN)

Convolutional neural network (CNN), is a main architecture of deep learning for multi-array of data, such as images, signals and languages. CNN works on local receptive field, shared weights and pooling. The first layer of CNN is always convolutional layer, and filters (neuron or a kernel) within the layer swipe across the whole areas of the input image. The region covered by the filter is called the receptive field. The depth of the filter has to be the same as the input depth. During the convolutional calculation, multiplications of the parameters in filter and the pixel values in the receptive field are summed up as one number (Figure 2.10). The computation in the convolutional layer can be described as:

$$\hat{Y} = a(\sum_{i=1}^{m}(W_i X_i) + b) \tag{2.11}$$

where $\hat{Y}$ is the output to next layer, W and b are weight and bias, respectively. The function $a()$ is activation function. In the intermediate layer of the model, we can choose whether to use the pooling layer or not to downsampling, which can greatly reduce the dimension of the data (Figure 2.11). The pooling layer can effectively avoid overfitting. Generally, the final layer of CNN is fully connected (FC) layer (Figure 2.12). After downsampling from previous layers,

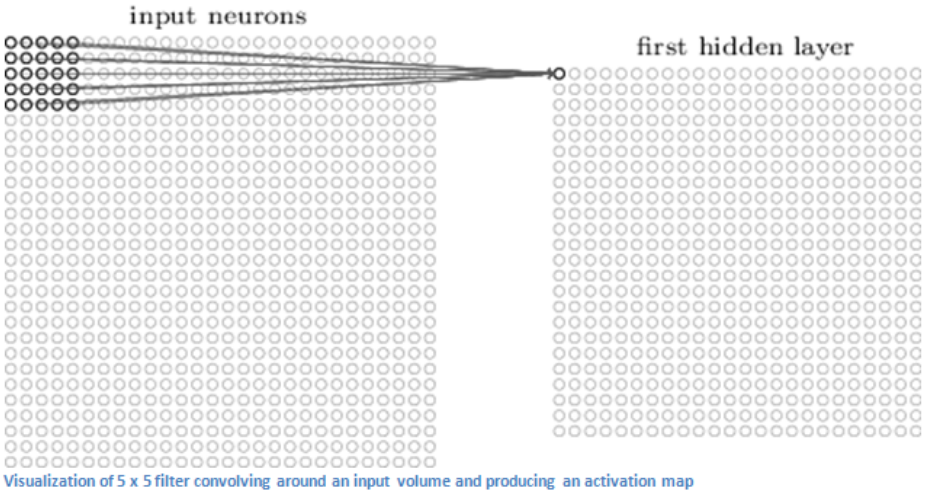


Figure 2.10: *Diagram of convolutional layer in CNN [66].*

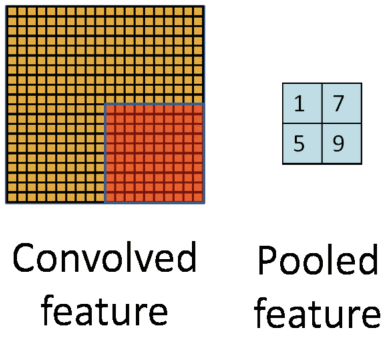


Figure 2.11: *Diagram of pooling layer in CNN.*

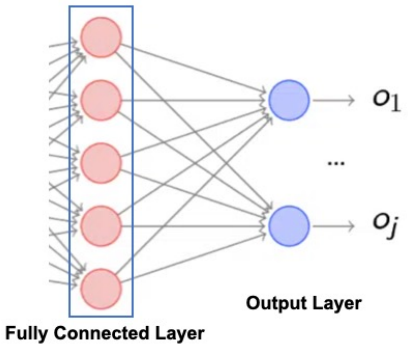


Figure 2.12: *Diagram of fully connected layer in CNN.*

the obtained neuron features are connected end to end and then convolved with FC layer parameters to get the corresponding output values. In addition to the model structure, the selection of the activation function for each layer is also crucial. These are some of the commonly used activation functions:

(1) Sigmoid

$$f(x) = sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{2.12}$$

(2) Tanh

$$f(x) = tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{2.13}$$

(3) Rectified Linear Unit (ReLU)

$$\sigma(x) = \begin{cases} max(0, x) & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{2.14}$$

(4) Softmax

$$f(x) = softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \tag{2.15}$$

where z_i is the output value of the i^{th} node, K is the number of categories.

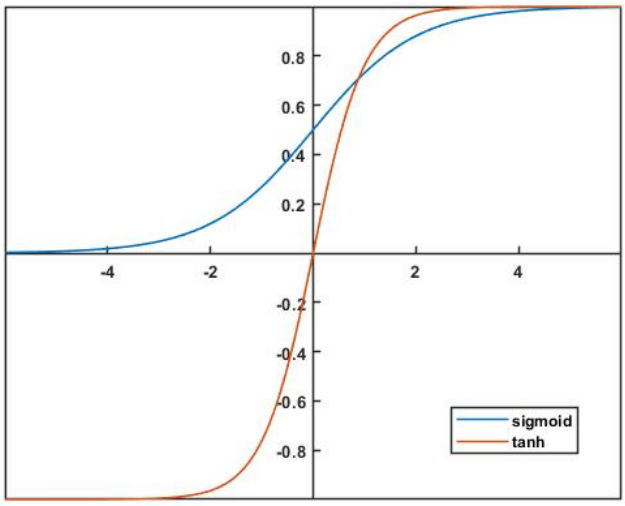


Figure 2.13: *Comparison between sigmoid() and tanh() function.*

In this thesis, the expected outputs are joint angles, and for each DOF, I used positive and negative values to indicate the direction of rotation from the central posture. The *tanh*() is centered on zero, negative inputs will be strongly mapped to negative, and zero inputs are mapped near zero, which is considered better than *sigmoid*() according to our targets (Figure 2.13). Therefore, I designed the activation function of convolutional layer to *tanh*(), which will be mentioned in Section 3.2.4.

2.4 Transfer Learning

Transfer learning is to transfer parameters of an existing trained model to a new model, so that help train the new model. Considering that most of the data or tasks are correlated, *e.g.*, we can ask amputees to perform the same motions for model update in different days. The learned model parameters (can be interpreted as learned knowledge by model) can be shared to the new model to accelerate and optimize the learning efficiency of the model. Therefore, we do not need to start training the model all over again everyday. Due to the degradation of the sEMG signal quality on different days, we could not directly

use the trained model for joint angle estimation even for the same participant. That is why we always need to update model parameters *via* transfer learning before each day's real-time control. There are many approaches to transfer learning, such as conservative training, layer transfer, multi-task learning, and so on [40, 67, 68].

In this thesis, I applied layer transfer, which is widely used for speech recognition or image tasks [40]. Description of layer transfer can be found from Figure 2.14, the process is to fix (copy) parameters of some layers directly into a new model, then update the rest of the layer parameters with only a small amount of new dataset. In this way, only a few parameters need to be trained during model update, so that can prevent over-fitting. To acquire sEMG signals, I used 32 channels multi-array electrode (Figure 2.3). The electrode sleeve completely cover the forearm, and for the same participant, we do not need to adjust the electrode location. Therefore, as the layer in direct interacts with the input signal, there is no need to retrain the first convolutional layer on different days. Because the electrode locations are not significantly shifted, only the quality and amplitude of the sEMG signal change. Thus, when updating the model, I considered fixing the parameters of the first convolutional layer. The layer transfer strategy was applied to all experiments in this thesis.

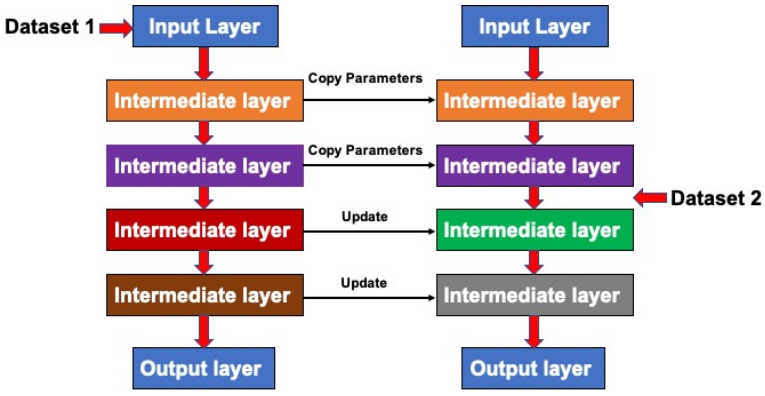


Figure 2.14: *Example of layer transfer. Left part is an existing model trained by Dataset 1, colors in different layer denote the parameters. During transfer learning, the parameters of the first two layers are fixed (copied to new model), and from right part, the new Dataset 2 is used to update the parameters of the last two layers.*

2.5 Aims

Even until now, it is still a challenging task to propose a control system with high accuracy and natural performance. In this thesis, I aimed at proposing a real-time control system for myoelectric prosthesis control. To achieve so, I need to (1) propose a model with good performance and high robustness, (2) build a real-time system based on this model, and (3) successfully control a virtual hand using the proposed system and complete some tasks, to prove the approach can be applied to prosthetic hand in the future.

2.5.1 Model Proposal for Motion Estimation

The first objective is to propose a model for ADL of human forearm. To achieve our goal, I need to solve the following problems:

- How to design the architecture of a desired CNN-based regression model?
- How to evaluate the regression accuracy of the model?
- How to calibrate the model parameters on different days?
- How to prove the model is better than conventional regression models?
- Explain how the model learn the motion patterns from sEMG signals?

Please refer to **Chapter 3**.

2.5.2 Real-Time Control System

The next objective is to build a real-time control system based on the proposed model. In order to reduce the data transmission delay, and smooth the oscillation of joint angles output (to protect the motor in prosthetic hand), I believe the control system should include the following modules:

1. A sliding window (120 ms) for raw sEMG acquisition;
2. Data Processing Module in real-time (Section 2.1.3);
3. Proposed regression model (**Chapter 3**);
4. Output Smoothing Module (Adaptive Kalman Filter).

After I proposed a model for motion estimation in **Chapter 3**, even though it obtained high accuracy in offline experiments, it might exist some significant real-time difference. Thus, I need the following evaluation for the system:

- Performance of both single- and multi-DOF motions simultaneously;
- Regression accuracy in real-time;
- Computational latency of the real-time control system;
  (how much; whether stable or not)
- Emphasize the regression motions in 3-DOFs simultaneously (TAC test);
- Successfully control a virtual hand using proposed method.

I have completed the above tests and achieved the desired objectives. Please refer to **Chapter 4**.

Chapter 3

Multi-Joint Angles Estimation using Regression Model

3.1 Overview

For healthy people, they can perform complex forearm movements generated by forearm muscle contraction and relaxation which controlled by brain. Upper limb amputees lose their forearm, the use of a myoelectric prosthetic hand should be a good solution to restore their activities of daily living. The control command can come from their remaining muscle or EEG signal. Compared to EEG signals, sEMG signals have higher accuracy, signal-to-noise ratio and reliability. However, using bio-signals as motion recognition method is still a challenging task.

With the development of technology, researchers started to consider using machine learning methods to recognize hand or forearm movements. Moreover, many studies proposed deep learning methods to train regression predictor such as CNN-based and RNN-based model, or the combination structure of RNNs and CNNs. However, almost all of the research deep learning method to do classification or regression without explaining how the model can learn the motion from sEMG signal. In sEMG signal pattern recognition, due to skin impedance or electrode shift, the model accuracy decreases in different days

even though the trained model in high accuracy. Therefore, the model update in different days is significant to this research topic.

For the above reasons, to propose a novel method for forearm motion recognition, we have to solve the following problems:

- Specific method to characterize motion pattern recognition (hand motion classification or joint angle estimation)
- Specific model base (CNN-based or RNN-based or hybrid structure)
- Explain how the model can learn the muscle activation from sEMG
- Propose a method to update the model in different days
- Model performance indicator

In this study, we proposed a channel-wise CNN (CW-CNN)-based high-precision regression model to predict forearm joint angles in 3-DOFs, and discussed the learnablity and robustness of the proposed model for amputees to update daily parameters. We used geometry plot to analyze muscle activation by backtracking model layer parameters. We designed the experiment in two different days. On the first day, for each participant, models were initially trained, which we named the Initial Experiment. On different day, experiment was conducted for a smaller amount of dataset to update the trained model parameters by transfer learning, which we named the Second Experiment. We compared our model with another four conventional regression models with and without transfer learning, they are linear regression (LR), support vector regression (SVR), K-nearest neighbor (KNN) and decision tree (DT) regression. The model comparison results proved that the proposed model significantly outperformed the conventional methods. The work in this chapter shows the stability and superiority of the proposed model, and we will applied it to future real-time control.

Table 3.1: Information of each participants. All of them participated in Initial Exp., and part of them were invited to join the Second Exp.. M indicates male, F indicates female; ◯ denotes participated to the corresponding experiment.

Participant ID	Age	Handedness	Gender	Initial Exp.	Second Exp.
S1	24	Right	M	◯	◯
S2	23	Right	M	◯	-
S3	21	Right	M	◯	-
S4	23	Right	F	◯	-
S5	23	Right	M	◯	-
S6	26	Right	M	◯	◯
S7	43	Right	M	◯	◯
S8	25	Right	M	◯	◯
S9	24	Right	M	◯	-
S10	25	Right	M	◯	◯

3.2 Methodology

3.2.1 Participants

We invited ten right-handed healthy participants joined our experiment, including nine males and one female, aged 21—43. The dataset from the participants were acquired at the Tokyo Institute of Technology, Japan. Experiment protocol was approved by the ethics committee of the Tokyo Institute of Technology, and was conducted in accordance with the Declaration of Helsinki. All ten participants were asked to read the participant information sheet and provide written informed consent to participate in the study. The detailed information of each participant can be checked in Table 3.1.

All participants joined the Initial Experiment, and five of them were invited into Second Experiment. The detailed information about Initial Experiment and Second Experiment, please refer to Section 3.2.2.

3.2.2 Experiment Protocol

The experimental paradigm for data acquisition was created using MATLAB (The MathWorks, Inc., USA). Each participant was instructed to sit in front of a screen. Lab streaming layer (LSL, refer to [69]) is a system for the uniform collection of measurement time series data in research experiments, in all experiments in this thesis, LSL was used to synchronize EMG signal and joint angles as dataset. Before starting the experiment, we need to perform the following steps:

- Moisten the electrode sponge with water and wear the bipolar multi-array electrode sleeve.
- Check signal quality.
- Instruct participants to wear the Perception Neuron glove.
- Perform calibration on Axis Neuron to build skeleton model.
- Check the 3-DOFs joint angles by choosing the corresponding marker name (Figure 2.9) and perform the corresponding motions (Figure 3.1).

After finishing the above preparations, participant was taught the motions they should perform during the experiment, and so that they can familiarize

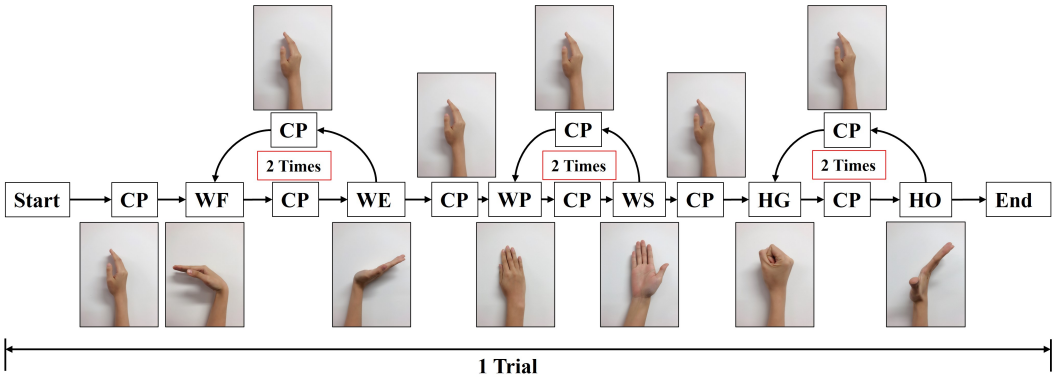


Figure 3.1: *Experiment paradigm for each trial. Participant will start from the central position (CP), and perform wrist flexion/extension (WF/WE) two times, wrist pronation/supination (WP/WS) two times, and hand grip/open (HG/HO) two times. After each motion, they should move their forearm back to CP and perform the next motion. The screen shows what motion they should perform in next stage.*

with the experimental paradigm (Figure 3.1) of each trial. Between each trial, participants were asked to relax their forearm muscle for approximately two minutes to prevent muscle fatigue. The order of each trial can be explained as follows: WF/WE twice, P/S twice, and HG/HO twice. At the beginning of each trial, participants performed their whole forearm as central position (CP). Then, screen displayed the next movement name to be performed, and participants had to rotate the corresponding joint from CP according to the displayed motions. For each motion, participants rotated their hand to the maximum angle and returned back to the CP. For example, if the screen shown the next motion is WF, the order of motion should be CP—WF—CP.

As mentioned before, even for the same participant, sEMG signal quality will be different in another days, and lead to a decrease in the prediction accuracy of the model. Therefore, although we can train a high accuracy model, we still cannot use it directly on different days. In this thesis, we applied transfer learning to solve this problem. To test the effectiveness of transfer learning, and apply to motion estimation in different day, we have to design the experiment in different days. Thus, we designed the following two experiment.

Initial Experiment

We invited the ten participants (Table 3.1) to conduct the Initial Experiment, each participant was required to obtain 10 trials data. After the experiment, sEMG signal data and joint angles data were processed and synchronized as dataset using LSL time stamps. The 10 trials dataset were used for the initial model training. The trained model can be used for the same participant and model parameters can be updated on different days by the experiment content of the Second Experiment, so that the model can keep the regression robustness for joint angles estimation.

Second Experiment

We invited the participant S1, S6, S7, S8 and S10 (Table 3.1) to join the Second Experiment. The Second Experiment was conducted two months after Initial Experiment. Participants need to complete 5 trials data acquisition, so that the dataset amount was reduced. The same experiment protocol (Figure 3.1) was also used to the Second Experiment, and the model parameters were updated based on the trained regression model and smaller amount of new dataset *via* transfer learning.

3.2.3 Dataset

After data acquisition experiment, we processed the raw sEMG signal to IEMG signal by the following steps (Section 2.1.3):

- Rectification
- FIR Low-pass Filtering [58]

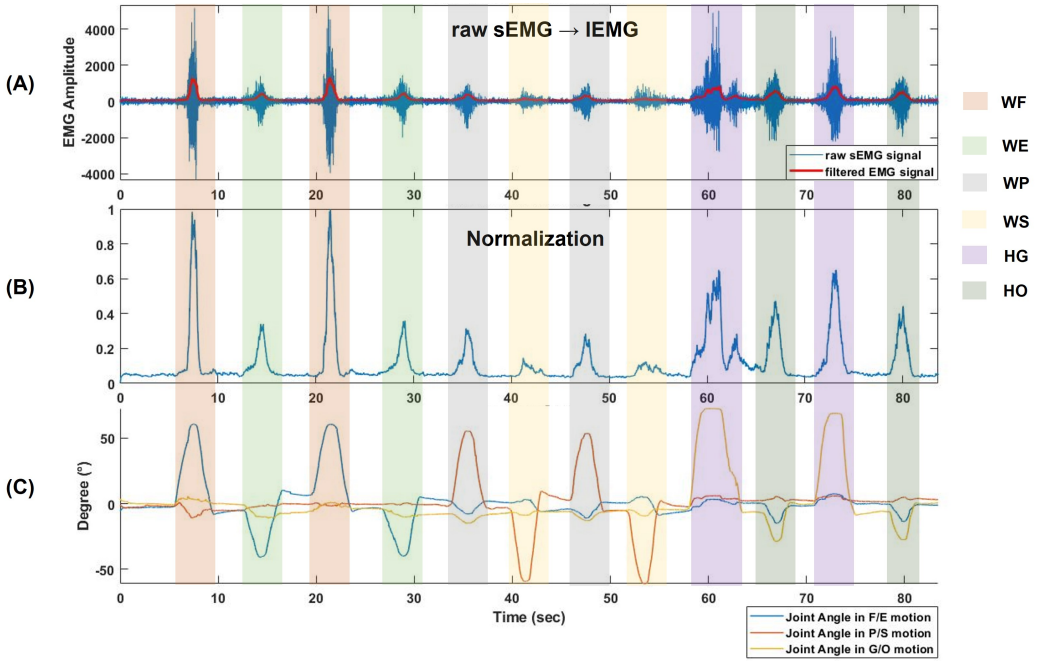


Figure 3.2: *Data Preprocessing and Synchronization (S1). (**A**) The processing from raw sEMG signal to IEMG signal, the data is obtained from ch10. Blue line is raw sEMG signal, and red line indicates the IEMG signal. (**B**) Normalize IEMG signal from 0 to 1 via Min-Max Normalization. (**C**) Acquired 3-DOFs joint angle data using Perception Neuron Motion Capture System.*

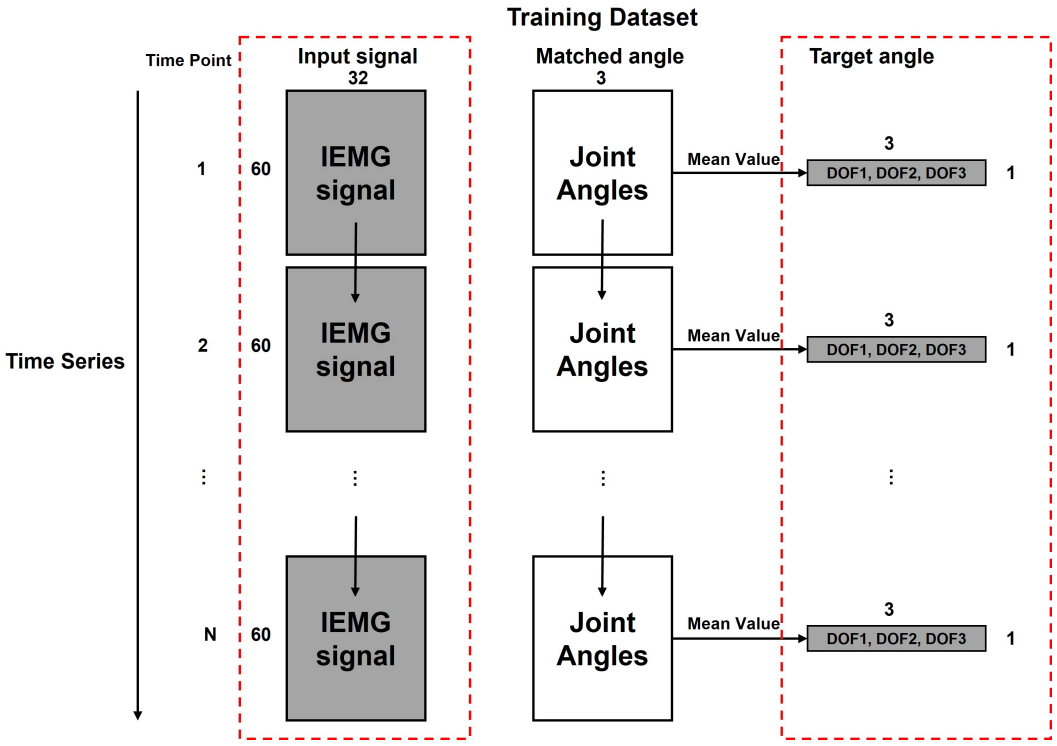


Figure 3.3: *Dataset generation process and structure. The 500 ms sliding window slides along the time series direction with a interval of 100 ms. Within the sliding window, calculate the mean value of [60×3] joint angles data in the DOF dimension and get [1×3] as target angle corresponding to the [60×32] IEMG input. The two data shown in the red dashed box (input signal and target angle) is the content of the dataset.*

- Min-Max Normalization (0—1)

Before dataset synchronization, we need to do sEMG signal preprocessing. Figure 3.2(A) shows an example of the processing result (S1, ch10) from raw sEMG signal to IEMG signal, the blue line is raw data, and the red line is IEMG data. The raw data was rectified, and filtered by the FIR filter proposed by Koike *et al.*. The Min-Max Normalization result was shown in Figure 3.2(B).

With LSL time stamps, we can check very clear data synchronization between raw sEMG, IEMG and joint angles data according to Figure 3.2. The Figure 3.2(C) shows the 3-DOFs joint angles data collected from Perception Neuron, the blue line is the DOF1 (joint for WF/WE motion), red line is the DOF2 (joint for WP/WS motion), and yellow line is the DOF3 (joint for HG/HO motion).

Due to the difference in sampling frequency between IEMG signal and joint angle data, to build a dataset for supervised deep learning, we have to match the two data. We re-sampled the EMG signal from 500 Hz to 120 Hz to match joint angles data. The dataset generation can be explained from Figure 3.3. After re-sampling, we used a sliding window of 500 ms to segment IEMG data and joint angles. For a sampling frequency of 120 Hz, 500 ms corresponds to a window segment length of 60 (=120Hz×0.5s). We hope the ideal dataset structure should be a series of [60×32] image corresponding to a series of [1×3] joint angles, thus, we calculated the mean value of the joint angles within the sliding window, as the target, corresponding to the IEMG image within that window, for supervised learning. The interval of the sliding window is 100 ms (400 ms overlapping).

After completing the data segmentation, the dataset was divided into five groups for five-fold cross validation (CV): For Initial Experiment, there are 10 trials data, so we randomly combined two trials as one group; For Second Experiment, there are 5 trials, one trial can be considered as a group with a total of five groups for five-fold CV.

3.2.4 Channel-Wise CNN Regression Model

In deep learning, CNN is a very important and commonly used architecture for multi-array data, such as images, languages and signals. CNN works on the local receptive field, pooling and shared wights [70].

In this Chapter, we presented a regression model to estimate joint angles in 3-DOFs based on CNN, which we named it channel-wise CNN (CW-CNN). The structure of proposed CW-CNN model is shown in Figure 3.4. This model includes a convolutional layer and an FC layer. The input as a [60×32] 2D image, indicates the 32 channels sEMG signal data in time dimension, where the 60 denotes 500 ms sliding window length (the sampling frequency of re-sampled IEMG and collected joint angles are 120 Hz).

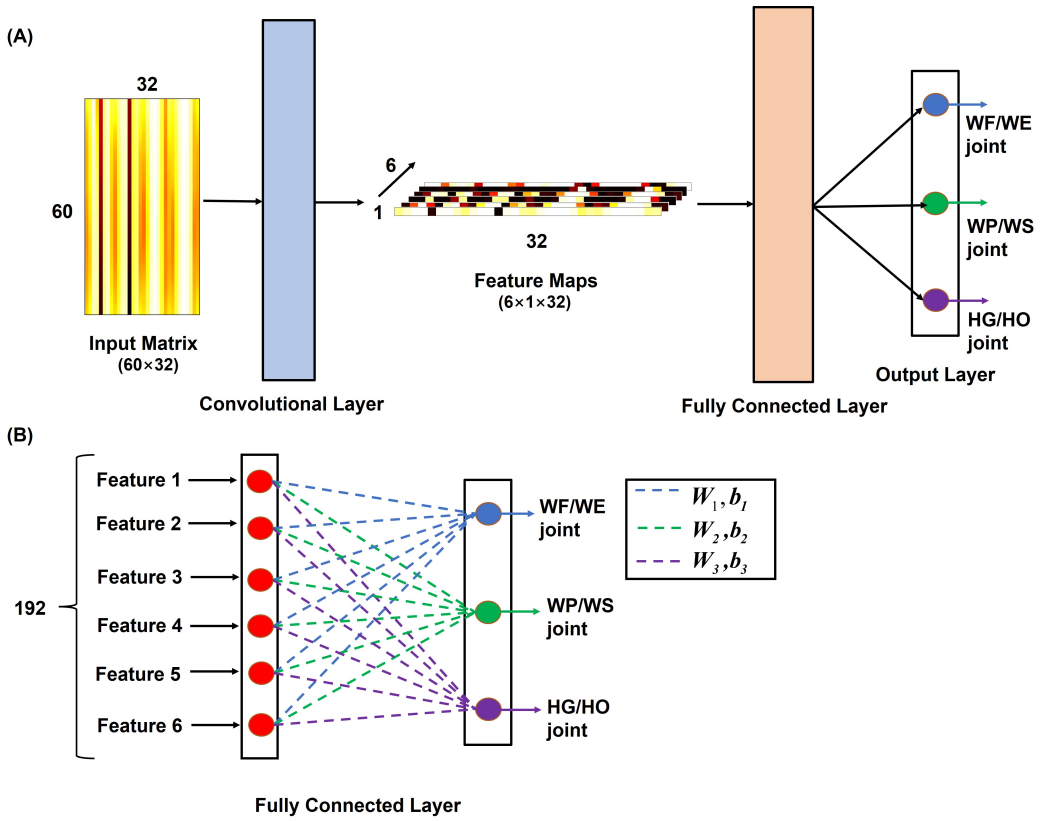


Figure 3.4: *Proposed CW-CNN regression model. (**A**) Model overall structure, including a channel-wise convolutional layer and a fully connected (FC) layer. In the convolutional layer, there are six [60×1] filters, stride=1, padding=0, activation function is tanh(). The output of convolutional layer is six [1×32] feature maps, then go through FC layer directly and calculated as three joint angles correspond to 3-DOFs. (**B**) Details description of FC layer. The six feature maps connected end to end to become a [192×1] sequence, each red point shows one transposed feature map represents a neuron. The sequence performs dot multiplication on three weight sum the corresponding bias to obtain joint angles correspond to different DOF.*

We designed the convolutional layer architecture as channel-wise, the concept was proposed by Sakhavi *et al.* in the paper [71] and applied to pattern recognition of EEG. We will discuss the reason of choice in Section 3.4. As shown in Figure 3.4(A), there are six channel-wise filters in convolutional layer, sized of [60×1] to compress the time dimension. Therefore, the outputs of this layer are six [1×32] feature maps in channel-wise. In this way, we can backtrack the parameters to analyze each channel. The activation function is *tanh*(), without padding and max pooling layer, with a stride size of 1.

The convolutional layer is followed by a FC layer (Figure 3.4(B)). The input is the six [1×32] feature maps. They are transposed and connected end-to-

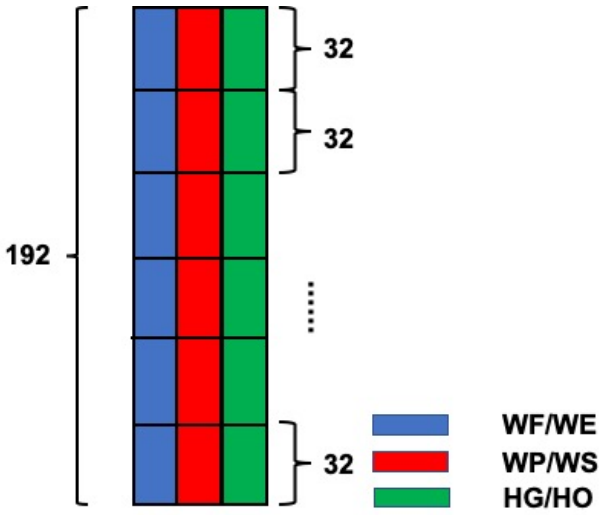


Figure 3.5: *Weight matrix in FC layer. The size of this matrix is [192×3], due to the [192×1] input consisted of the six feature maps, each of the 32 numbers corresponds to a feature map, and each of these 32 numbers corresponds to a channel. Each column represents a DOF, thus, each DOF joint angle can be calculated by multiplying the corresponding numbers of [192×1] input and corresponding weight vector one by one and sum them as output.*

end as [192×1] vector, and could be used to calculate the 3-DOFs joint angles output *via* the following equation.

$$Angle_i = W_i^T \cdot fm + b_i \tag{3.1}$$

Where i is the joint number, namely, i=1,2,3; $Angle_i$ is the joint angle we need to obtain: $Angle_1$ is the angle for DOF1 (WF/WE), $Angle_2$ is the angle for DOF2 (WP/WS), and $Angle_3$ is the angle for DOF3 (HG/HO); fm is the fully connected [192×1] feature map; W_i is the weight parameter matrix within FC layer, W_1 is related to DOF1, W_2 is related DOF2, and W_3 is related to DOF3; The detailed information of weight matrix can be checked according to Figure 3.5; b_i is bias parameter within FC layer. The output of the proposed CW-CNN regression model is a [3×1] vector, indicates the 3-DOFs angles respectively. In the future control of prosthetic hand or virtual hand model, the 3-DOFs joint angles will be regarded as control command and sent to the three activation joints (or motors) respectively.

3.2.5 Geometry Plot

It is difficult to discuss the reasons for getting good training results *via* deep learning method, because actually deep learning method is a black box model. In 2019, Spornchaisit *et al.* [72] applied a topology graph to plot the weight of the IC as heat map to study the relationship between channels. In 2020, Tam *et al.* [47] also used heat map to show the EMG activation maps. Therefore, we can also consider using heat maps to represent the muscle activation on both forearm flexor side and extensor side.

In this thesis, I backtracked the weight parameters in the FC layer, and plot the parameters as heat maps, which I named it geometry plot. The weights have a significant contribution to the different motions in FC layer, so that I can explain why the proposed CW-CNN regression model obtained good training and testing results. Here, the weights were separated corresponding to channels (Figure 2.3), and represented the wrist flexor and extensor side respectively. As mentioned in Formula 3.1, W_i (i=1,2,3) is a [192×1] vector, and the [192×1] input consisted of the six feature maps (Figure 3.5), each of the 32 sections corresponds to a feature map, each weight numbers corresponding to each channel. Therefore, the values of the weights indicate the importance of the channels for different forearm motions. All the six feature maps contribute to the computation of the 3-DOFs joint angles with the corresponding weight vector, thus, to check the motion pattern heat map using weight, I separated the FC layer weight to six [32×1] corresponding to the six feature maps, and I performed the superposition calculation as [32×1]. According to the channel array in both side of forearm (Figure 2.3), I arranged the 32 weight values and plotted the two matrix as heat maps (geometry plot). To analyze the motion of DOF1, I plotted the geometry plot based on W_1; for DOF2, I plotted using W_2; and for DOF3, I used W_3. Please check the results from Section 3.3.4.

3.3 Experimental Results

I used Pytorch1.3.1 as framework to finish the deep learning program, and GeForce RTX 2080 graphics processing unit (GPU) and CUDA 10.1 to accelerated the training process. For model training, I used Adam optimizer to minimize loss function for model parameters update, the loss function I used is mean-square error (MSELoss). In this Chapter, the learning rate was $1e-3$. There are 15 training epochs for Initial Experiment, and 5 training epochs for Second Experiment.

For the model training of proposed CW-CNN regression model, I used k-fold cross validation (CV) [73], [74]. Generally, CV is applied to evaluate the performance of a machine learning model, especially the performance of trained model on new data, and reduce over-fitting. Further, more effective information can be obtained from limited dataset. The k-fold CV can reduce the variance by averaging the results of k different group training, thus, the performance of the model can be less sensitive to data. In this Chapter, k=5, *i.e.*, I applied 5-fold CV for model training and testing.

3.3.1 Evaluation Metrics

In order to evaluate the training effect of the model *via* 5-fold CV, I used Pearson correlation coefficient (CC) as evaluation metrics, which can be used to statistically measure the relationship strength between two variables [75]. In this thesis, I measured the relationship strength between the estimated joint angles and the collected joint angles *via* CC, to evaluate the model accuracy:

$$CC = \gamma(X, Y) = \frac{1}{N-1} \sum_{i=1}^{N} \Big(\frac{X_i - \mu_X}{\sigma_X} \Big) \Big(\frac{Y_i - \mu_Y}{\sigma_Y} \Big) \tag{3.2}$$

Where X and Y are the two variables; N is the length of series, $N_X = N_Y$; μ_X and μ_Y are the mean value of the variable X and Y respectively; σ_X and σ_Y are the standard deviation of variable X and Y. The calculated CC ranges

from -1 to 1:

- $CC = 0$: No relation ship between the variable X and Y.
- $CC > 0$: Positive correlation (the higher the better).
- $CC < 0$: Negative correlation (not expected).

Thus, if we hope to train a high accuracy regression model for 3-DOFs joint angles estimation, we need to try to increase CC value and make it as close as possible to 1.

In this Chapter, the dataset was divided into five groups, and we use 5-fold CV to test the training effectiveness: there are five iterations, for each iteration, the proposed model was trained using four groups dataset, the remaining dataset was used for testing. Therefore, we can obtain five CC values, and the average of the five values can be calculated for the final testing result [71] [76]:

$$CC_5 = \frac{1}{5}\sum_{k=1}^{5} CC_k \tag{3.3}$$

3.3.2 Offline Joint Angles Estimation

In this subsection, I will show the joint angles offline estimation result for Initial Experiment and Second Experiment, and explain the effectiveness of transfer learning.

Results of Initial Experiment

In the Initial Experiment, the mean CC values of all participants (S1—S10) are listed in the Table 3.2. An example of the joint angles estimation results are plotted in Figure 3.6, where the red solid line indicates the measured angles acquired from Perception Neuron Motion Capture System, green dashed line is the estimated joint angles *via* proposed CW-CNN regression model. In this figure, CC results of S1 have been shown, $CC = 0.95$ for DOF1, $CC = 0.97$ for DOF2, and $CC = 0.91$ for DOF3.

Table 3.2: CC Results of 5-fold CV of Initial Experiment (mean CC ± std).

	DOF1	DOF2	DOF3
S1	0.93 ± 0.01	0.95 ± 0.03	0.88 ± 0.04
S2	0.89 ± 0.03	0.93 ± 0.04	0.94 ± 0.05
S3	0.87 ± 0.03	0.86 ± 0.02	0.82 ± 0.05
S4	0.85 ± 0.05	0.86 ± 0.03	0.85 ± 0.02
S5	0.88 ± 0.03	0.72 ± 0.05	0.78 ± 0.03
S6	0.90 ± 0.03	0.86 ± 0.04	0.89 ± 0.05
S7	0.91 ± 0.04	0.93 ± 0.02	0.76 ± 0.01
S8	0.86 ± 0.02	0.94 ± 0.03	0.81 ± 0.02
S9	0.86 ± 0.02	0.82 ± 0.02	0.85 ± 0.02
S10	0.88 ± 0.02	0.84 ± 0.02	0.85 ± 0.02
Mean	0.88 ± 0.03	0.87 ± 0.03	0.84 ± 0.03

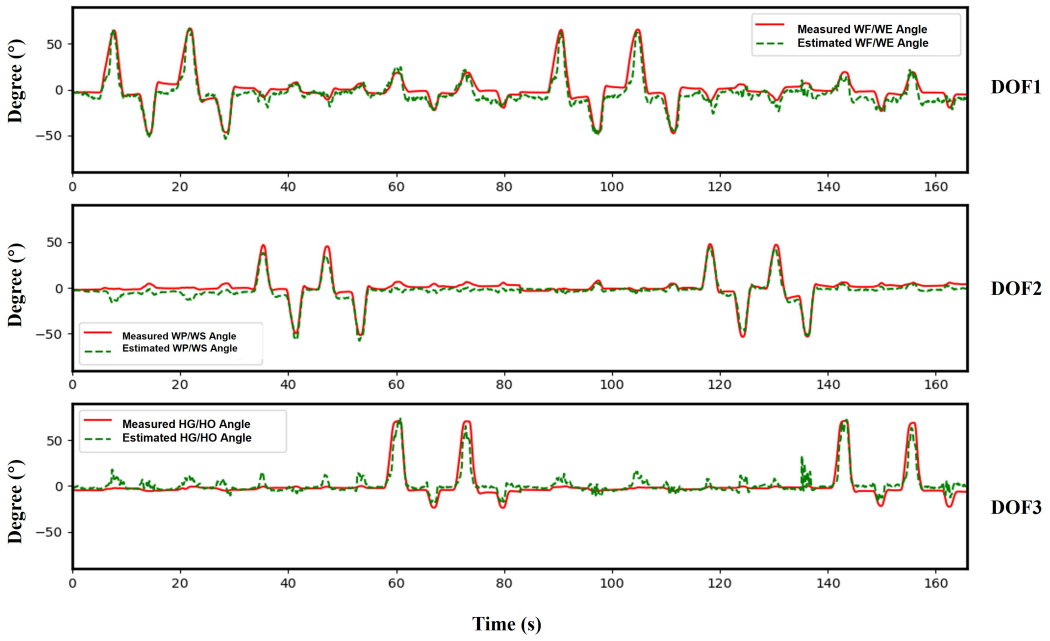


Figure 3.6: *Offline Testing Result of Initial Experiment (S1). Top: DOF1 (WF/WE joint); Middle: DOF2 (WP/WS joint); Bottom: DOF3 (HG/HO joint). Red solid line is the measured joint angles obtained by Perception Neuron Motion Capture System, green dashed line is the estimated joint angles using proposed CW-CNN regression model. This result shows CC=0.95 for DOF1, CC=0.97 for DOF2, CC=0.91 for DOF3.*

Table 3.3: CC Results of 5-fold CV of Second Experiment (mean CC ± std).

	DOF1	DOF2	DOF3
S1	0.97 ± 0.01	0.92 ± 0.03	0.92 ± 0.03
S6	0.92 ± 0.02	0.84 ± 0.07	0.93 ± 0.04
S7	0.92 ± 0.01	0.96 ± 0.01	0.89 ± 0.03
S8	0.91 ± 0.04	0.95 ± 0.03	0.87 ± 0.04
S10	0.91 ± 0.03	0.95 ± 0.03	0.87 ± 0.04
Mean	0.93 ± 0.02	0.92 ± 0.03	0.89 ± 0.04

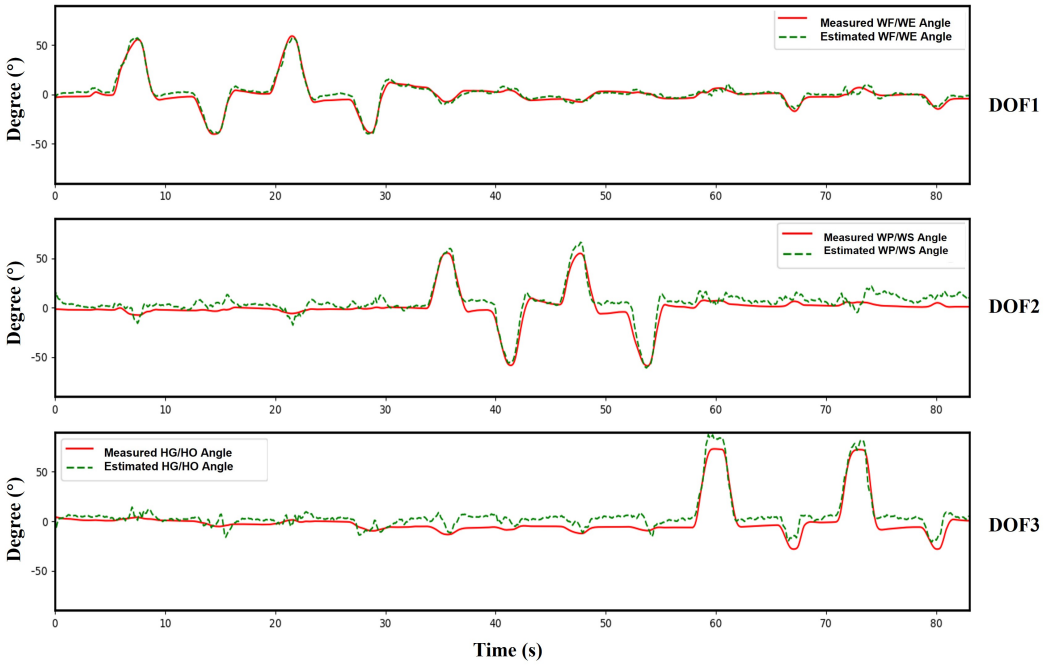


Figure 3.7: *Offline Testing Result of Second Experiment (S1). Top: DOF1 (WF/WE joint); Middle: DOF2 (WP/WS joint); Bottom: DOF3 (HG/HO joint). Red solid line is the measured joint angles obtained; Green dashed line is the estimated joint angles using the proposed model. This result shows CC=0.98 for DOF1, CC=0.96 for DOF2, CC=0.96 for DOF3.*

Results of Second Experiment

In different days, I invited the participant S1, S6, S7, S8 and S10 to do our Second Experiment. I obtained smaller amount of dataset, and used transfer learning to update model parameters. The mean CC results are listed in Table 3.3. Example of the estimated joint angles are compared with measured data and plotted in Figure 3.7. This figure shows the prediction result of S1, CC = 0.98 for DOF1, CC = 0.96 for DOF2, and CC = 0.96 for DOF3.

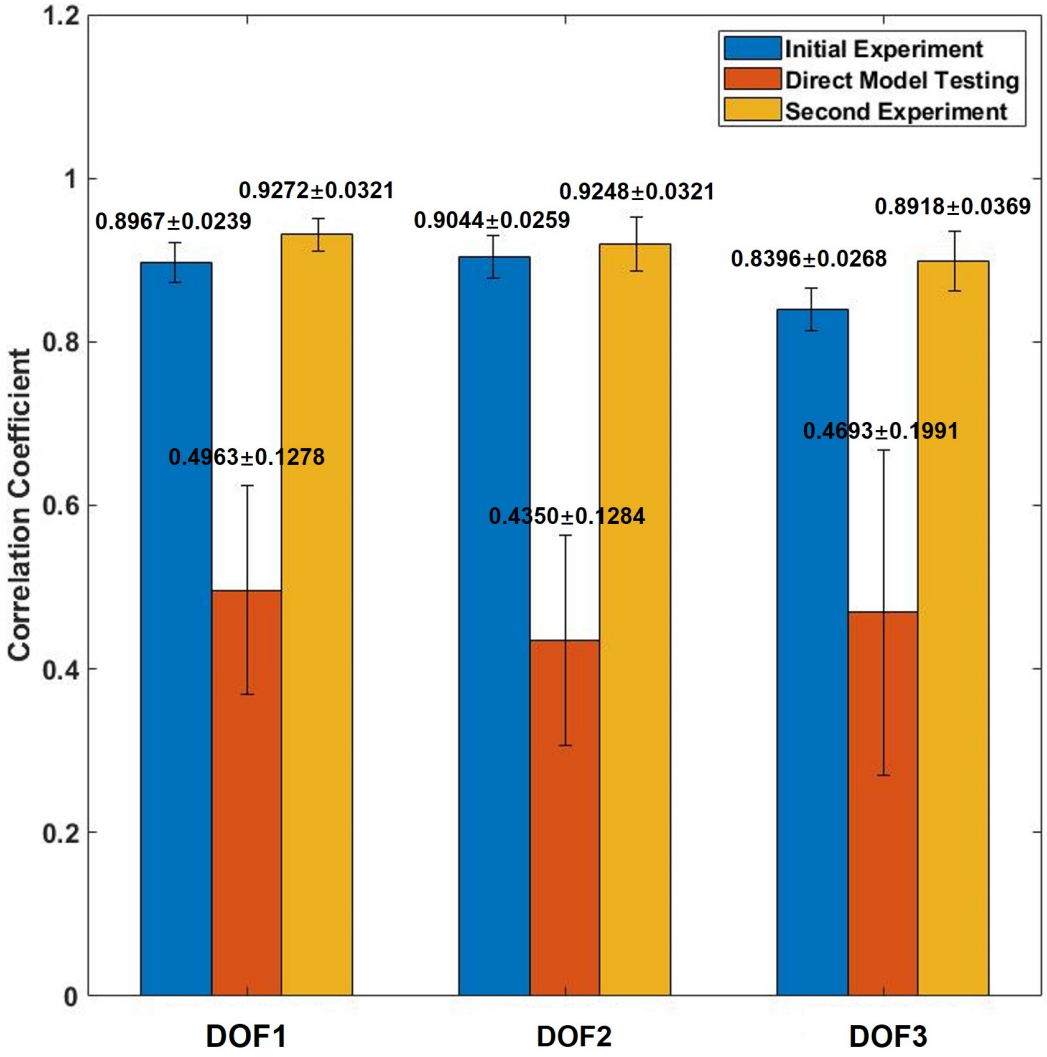


Figure 3.8: *Effectiveness of Transfer Learning. The bar plot shows the mean CC result of participant S1, S6, S7, S8 and S10. Blue bar indicates the Initial Experiment result; Red bar shows the Direct Model Testing result; Yellow bar shows the Second Experiment result.*

Transfer Learning Effectiveness

In this thesis, I applied layer transfer as our transfer learning method, the reason is discussed in Section 3.4.1. After the Initial Experiment, I trained model for each participant; In different day, due to the difference of sEMG signal, model must be updated to keep robustness. To prove the effectiveness of transfer learning, I need to check the model performance with trained model and new dataset, and compare the result with the updated model. Thus, before updating the model, I used the previous trained model to test on the new dataset directly for each participant (I call it *Direct Model Testing*); then, after

data acquisition in Second Experiment, I fixed the convolutional layer parameters, only trained the FC layer parameters using the new dataset. Figure 3.8 shows the corresponding result, the blue bar is the mean CC of S1, S6, S7, S8 and S10 in Initial Experiment (10 trials dataset, 5-fold CV), the red bar is the mean CC result of Direct Model Testing, and the yellow bar is the mean CC results of S1, S6, S7, S8 and S10 in Second Experiment (5 trials dataset, 5-fold CV).

As our expectation, when compared to the blue bars (Initial Experiment), the red bars (Direct Model Testing) show that the mean CC decreased: for DOF1, reduced from 0.8967 to 0.4963; for DOF2, reduced from 0.9044 to 0.4350; for DOF3, reduced from 0.8396 to 0.4693. With small number new dataset and only 5 training epochs, the mean CC results were improved. The yellow bars (Second Experiment) show that, compared to the red bars, with transfer learning, the robustness of the model restored to usable levels; compared to the blue bars, transfer learning can even improve the performance of the model(for DOF1, improved from 0.8967 to 0.9272; for DOF2, improved from 0.9044 to 0.9248; for DOF3, improved from 0.8396 to 0.8918).

A similar conclusion can be drawn by comparing Table 3.2 and Table 3.3. By comparing the results of the Initial Experiment and Second Experiment for the five participants (S1, S6, S7, S8 and S10), I can find that transfer learning can indeed maintain the robustness of the model on different days while improving the performance of the joint angles estimation.

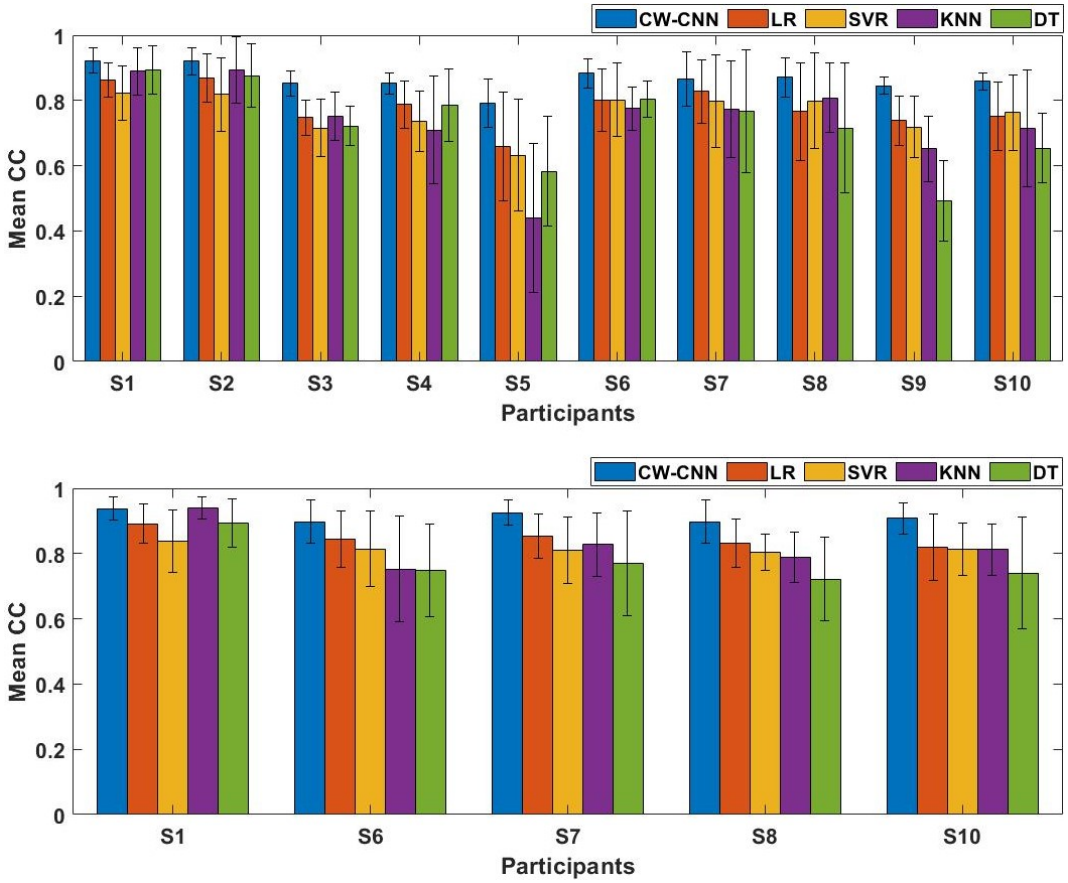


Figure 3.9: *Performance comparison results of each participant using five regression models (proposed CW-CNN, LR, SVR, KNN and DT). Top: Initial Experiment; Bottom: Second Experiment.*

3.3.3 Models Comparison

To highlight the superiority of the proposed CW-CNN regression model in multi-joint angles estimation, I compared it with four commonly used conventional regression model, they are linear regression (LR), support vector regression (SVR), K-nearest neighbor (KNN) and decision tree (DT) regression. I compared the five models with and without transfer learning.

Figure 3.9 is the average CC value of the five regression models of each participant. The top figure is the Initial Experiment result (before transfer learning), and the bottom figure is the Second Experiment (with transfer learning) result. This result indicates the proposed model outperformed conventional regression models for all participants with or without transfer learning.

In order to compare the performance between the five regression models, statistical analysis was applied. I calculated and compared the p-values be-

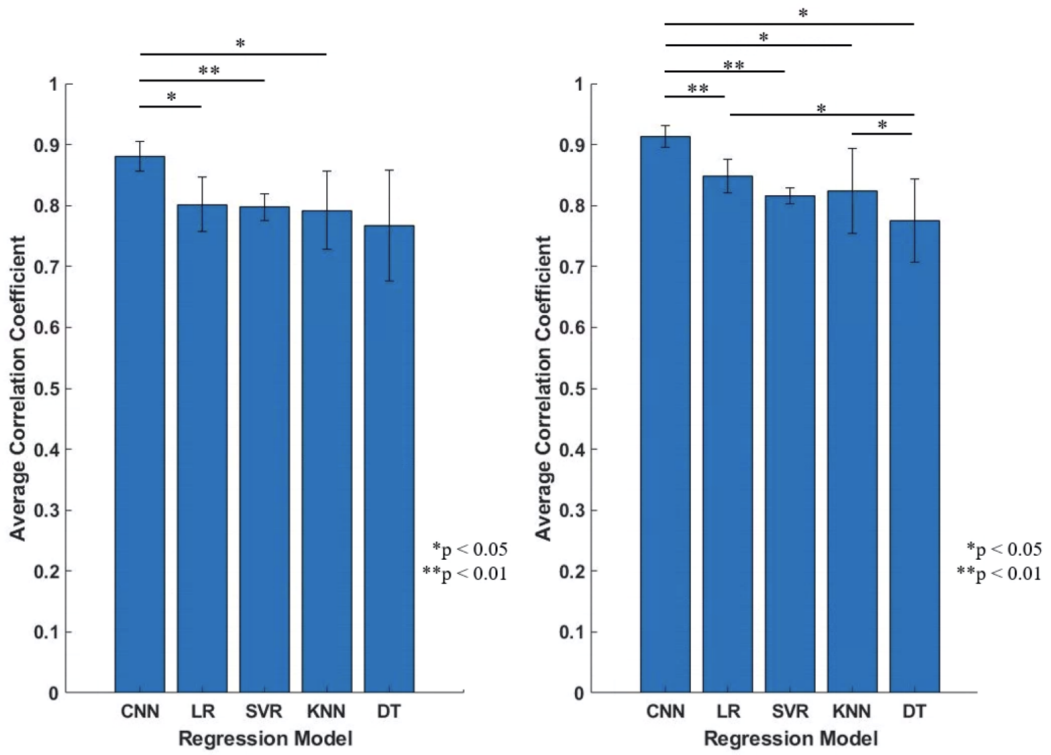


Figure 3.10: *Performance comparison results of the five regression models. Left: Initial Experiment; Right: Second Experiment. Statistical differences were calculated using T-test with Benjamini and Hochberg false dicovery rate (BHFDR) correction for multiple comparison [77].*

tween each of the two models, and tried to investigate the best model. The Figure 3.10 shows the average CC value of the five regression models of all participants. The left figure is the Initial Experiment result (before transfer learning), and the right figure is the Second Experiment result (with transfer learning). The statistical analysis was applied for evaluation. We can find that, according to the p-value, the proposed CW-CNN regression model significantly outperformed the other four conventional models, and the error bar of the proposed model is smaller and more stable. The detail of the statistical analysis method and result will be explained in Section 3.4.3.

3.3.4 Geometry Plot Results

Table 3.2 and 3.3 indicate that the proposed CW-CNN model has ability to estimate the joint angles in 3-DOFs of forearm motion with high accuracy. To explore how the model can be trained to such high precision, I used geometry

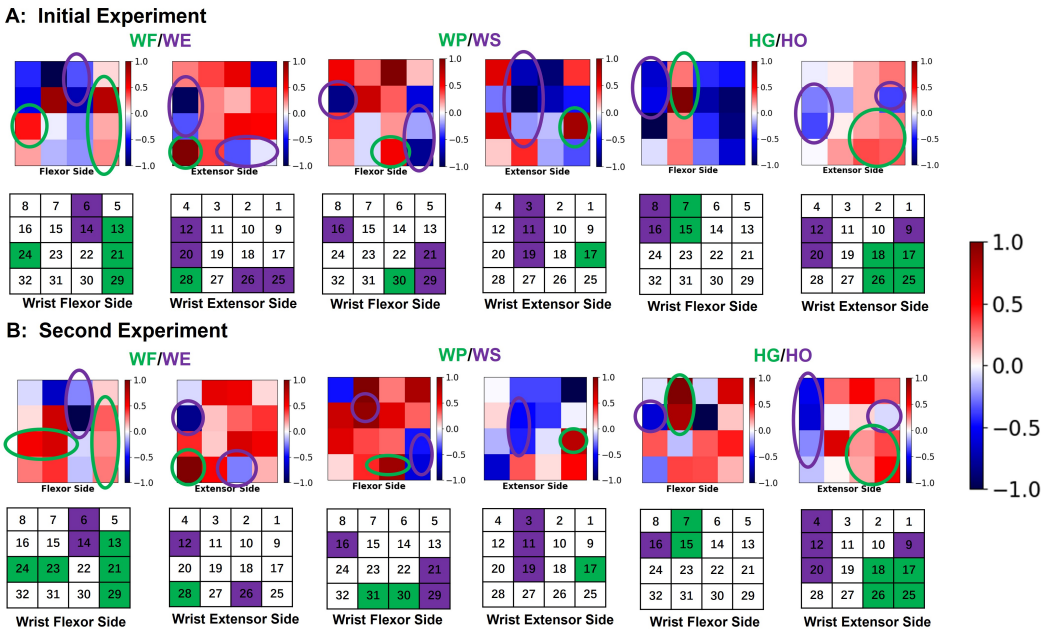


Figure 3.11: *Geometry Plot Result (S1). Each geometry plot has flexor side and extensor side. The value of the weight are normalized from -1—1 and shown as color bar. The white color (0 value) indicates the channel does not contribute to the corresponding motions; red color (0—1) shows the corresponding area contribute to positive joint angles (WF, WP, HG); blue color (-1—0) shows the area contributes to negative joint angles (WE, WS, HO). The circles show the motion pattern locations: the green circle denotes positive joint angles; purple circle denotes negative joint angles. The bottom matrix shows the motion patterns with channels. (A) Initial Experiment result; (B) Second Experiment result.*

plot mentioned in Section 3.2.5 to analyze the forearm muscle activity. I backtracked from the weight matrix of FC layer to create geometry plot. We can see the result of participant S1 from Figure 3.11, geometry plot includes wrist flexor side and wrist extensor side (Figure 2.3). The color bar ranges from -1 to 1, the blue part (-1—0) indicates the areas contribute to negative joint angles, *i.e.*, WE, WS and HO; the white color (0) indicates the area has no contribution to corresponding motion; the red part (0—1) indicates the area contributes to positive joint angles, *i.e.*, WF, WP and HG. The matrices under the geometry plots are corresponding motion patterns with channel number, green area (green circle of geometry plot) shows the muscle activation when the forearm performs motions for positive joint angles, purple area (purple circle of geometry plot) denotes the muscle activation for negative joint angles. Figure 3.11(A) is the result of Initial Experiment, and Figure 3.11(B) is Second

Experiment result.

The results shown that, muscle patterns can be shown by backtracking the weight parameters of FC layer, and the motion pattern locations correspond to the actual muscles in human forearm. I will discuss the relationship between FC layer weights and channel-wise filters in Section 3.4.4. And according to the comparison between Figure 3.11(A) and (B), even though the sEMG signal quality of the same participant changes everyday, the muscle pattern of forearm area can be kept *via* transfer learning, I consider they are the reasons why the proposed model performs well whether with or without transfer learning.

3.4 Discussions

In this Chapter, I proposed a CW-CNN regression model to estimate 3-DOFs joint angles based on sEMG signals. I designed the experiment in two different days for each participant, and used Pearson CC to evaluate the model performance by comparing estimated joint angles with measured data. To prove the superiority of the model, I compared our CW-CNN regression model with four conventional regression models (LR, SVR, KNN and DT), and concluded that the proposed model significantly outperformed. Moreover, I backtracked the FC layer parameters to investigated the muscle activation by creating geometry plot. The experimental results show that this model can estimate multi-joint angles in high accuracy, and the proposed model can be applied in different days by updating the model parameters with small amount of new dataset *via* transfer learning.

In this section, I explain the reason to design the CNN as channel-wise convolutional layer structure in **3.4.1**, and analyze the estimation results in Section 3.4.2. To explain the model comparison results, I introduce the statistical analysis method in Section 3.4.3. Finally, I discuss the relationship between geometry plot (muscle activation pattern) and forearm motion anatomy in Section 3.4.4.

3.4.1 Channel-Wise Structure Design

In this Chapter, the proposed CW-CNN regression model includes a convolutional layer and one FC layer. As we all know, FC layer can just flatten the input (here, the six feature maps), thus, the most important point is to design the convolutional layer structure. In 2018, Sakhavi *et al.* [71] introduced three convolutional kernel for linear mixture of EEG signals input: channel-wise CNN (CW-CNN), channel mixing CNN (CM-CNN), and channel-wise convolution with channel mixing (C2CM). According to the study of [71], the authors mentioned the difference is that CW-CNN does not demonstrate channel mixing, which may lead to widened networks. Thus, without channel mixing, the network receptive field can be emphasized.

The skeletal muscle contraction and relaxation process can be described as Figure 3.12. When we want to activate our muscles, our brain generates the electrical signals to nerve endings, so that transmit the motion potentials to nerve endings cell membranes. This procedure lead a series of chemical changes to change the tropomyosin conformation. The binding site of the protein and myosin, as well as the head of myosin, is activated, generating the drive power to swing the head and slide the thin filaments. During this process, time delay

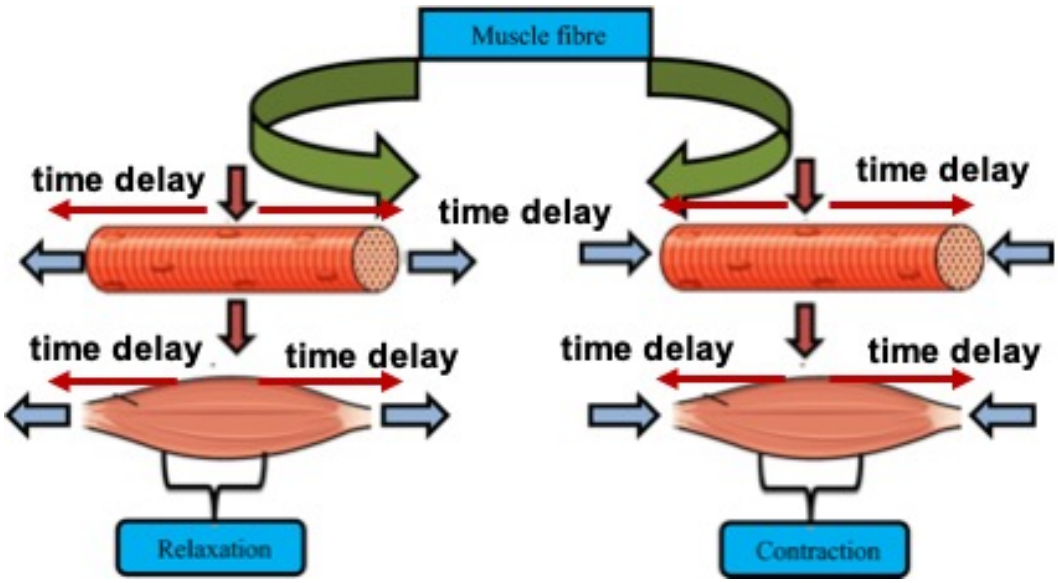


Figure 3.12: *Muscle Contraction and Relaxation Process [78].*

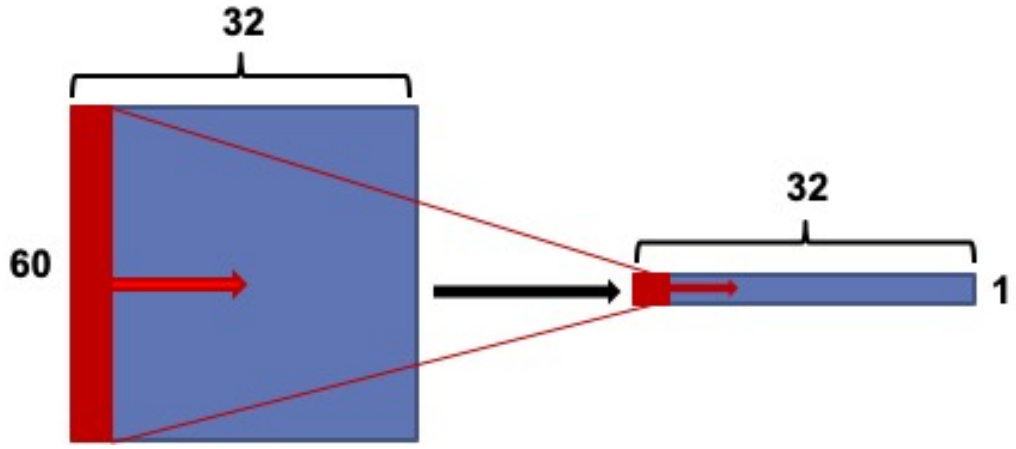


Figure 3.13: *Time Compression Process of Channel-wise Filter. The channel-wise filter reduces the temporal dimension to one, so that the output consists of 32 channels' information.*

will be generated from the arrival of the eletrical signals from brain to nerve endings, or to neurons to contract or relax our muscles. In this thesis, our input for CW-CNN is a mixture of 32 channels sEMG signals with temporal dimension ([60×32]), and I aim at providing sufficient information of sEMG signals and muscle force pattern to CW-CNN model, even during the time delay of electrical signals transmition.

The six channel-wise filters designed in the convolutional layer can reduce the 60-dimension of temporal to 1-dimension (Figure 3.13), thus, the feature maps output includes the 32 channels information. The 32 numbers in feature map correspond to each channel, and they are independent to each other.Due to the reduction of temporal dimension, we can obtain the muscle force pattern (FP) from each channel *via* such channel-wise convolution kernel. Unlike [71], which used the kernel to process EEG signal, I am the first time to use such kind of convolution kernel to process obtain muscle FP based on sEMG signal. The channel-wise filter can also be called as FP filter.

And precisely because FP filter can extract the FP information of muscles for each forearm motions, after the model initial training (Initial Experiment), the muscle pattern of each participant is preserved in another form by the model—the weight parameters in FC layer. The reason is that, the weight

value corresponds to each number in feature map (so that corresponds to each channel). For the same participant, the FP is the same. Therefore, in another days (Second Experiment), we can use transfer learning to update the model, so that the trained model can adapt to the sEMG signal quality of that day: fix the convolutional layer (we do not need to change FP for each participant), only train FC layer (the calculation from feature maps to joint angles can be improved to adapt different signal quality). This process method belongs to *layer transfer* in transfer learning.

The related discussions about the relationship between muscle activation pattern and anatomy refer to Section 3.4.4.

3.4.2 Joint Angles Estimation

In this thesis, I used Pearson CC (Formula 3.2) to evaluate the correlation between estimated joint angles and measured joint angles. The model was trained by 5-fold CV, thus, Formula 3.3 was applied to obtain the average CC of different participants for 3-DOFs joint angles. Table 3.2 shows the 5-fold CV results of all participants (mean $CC \pm std.$), and the intuitive result is represented as Figure 3.14. I checked the quality of raw sEMG signal from each participant's dataset, S1, S2 and S8 are the best, S3, S4, S6, S7, S9 and S10 are lightly noisy, and S5 is the worst (almost half of the channels are noisy). From the box plot in Figure 3.14, S5 performed the worst, especially for WP/WS motion, and S1, S2 and S8 performed the best. Therefore, I inferred that the noisy channels confuse the estimation of WP/WS and HG/HO motions, which contain more muscle crosstalk, and hence lead to worse results than WF/WE degree. I can further conclude that with less noisy, the proposed regression model can perform well, and this view is proved in next reasearch. In Chapter 4, all raw sEMG signal are acquired in high quality without big noise, and the results are good.

Figure 3.6 shows one of the testing results of S1, the top figure shows the

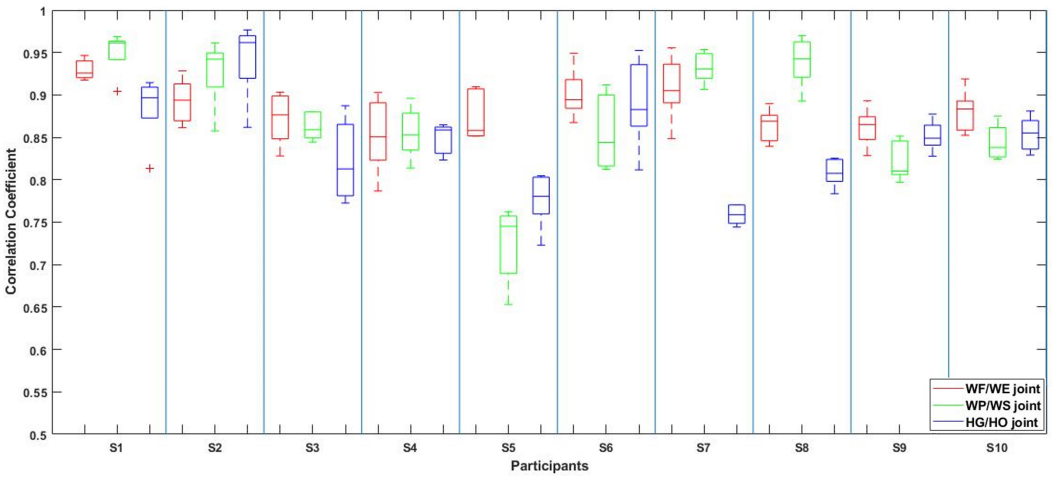


Figure 3.14: *Box plot of the f-fold CV results of all participants in Initial Experiment according to Table 3.2. The red box is the CC value of WF/WE degree (DOF1), green box is the CC value of WP/WS degree (DOF2), and blue box is the CC value of HG/HO degree (DOF3).*

DOF1, middle figure shows the DOF2, and bottom figure shows the DOF3. We can find that the estimated 3-DOFs joint angles met our expectations. The result indicates that WF/WE motions and HG/HO motions have common muscle areas, thus, when the participant performed HG or HO, the WF/WE joint also showed small angles, vice versa. I invited the participants with relatively high-quality raw sEMG signals (S1:Figure 3.2, S6, S7, S8 and S10) participated in the Second Experiment.

During the Second Experiment, I used the same experimental paradigm (Figure 3.1), and obtained 5 trials dataset for 5-fold CV. If we used the trained model for control in different days, the estimation accuracy will be decreased (Figure 3.8) due to skin impedance or electrode shift. I updated the trained model parameters based on small amount of new dataset *via* layer transfer learning: fix the convolutional layer parameters, update FC layer parameters. The testing CC results of 5-fold CV of the five participants are shown in Table 3.3, the mean CC values of the five participants are 0.93 ± 0.02, 0.92 ± 0.03, and 0.89 ± 0.04 for DOF1, DOF2, DOF3, respectively.

Then, I will explain why using transfer learning can keep the model robustness in different days, and even improve (Figure 3.8). As mentioned in Section 3.4.1, in the Initial Experiment, the parameters of FP filters and FC layer were

trained from the 10 trials datasets, the parameters include all of the information from the dataset of the participants. Refer to [71], the proposed FP filters in convolutional layer can obtain the muscle FP from the 32 channels, namely, feature maps extracted from sEMG signal are force patterns. Different people has their own FP, thus, I trained the model for each participant. In the Second Experiment, I used layer transfer, the model still contains the information from the previous 10 trials data, and the motion FP is the same for the same participant. The FC layer parameters can be updated for new dataset at that day. From the perspective of the entire training process, actually what we do is to add a new dataset to the original dataset. Therefore, this approach ensures that, in the daily update, the model can be calibrated based on only less training set, the overall training set is constantly superimposed.

3.4.3 Model Significance Analysis

I compared the proposed CW-CNN model with another four conventional regression models, they are LR, SVR, KNN and DT. For different models, I used the same dataset and input format to obtain the 3-DOFs joint angles, and evaluated the correlation between estimated joint angles and measured angles and compared with another models' results.

Figure 3.9 is the comparison result of all participants with error bars. The top figure is the result of Initial Experiment, the bottom figure is the reuslt of the Second Experiment (with model update *via* transfer learning). Figure 3.10 shows the comparison result between each regression models based on all participants' data, left figure is Initial Experiment result, right figure is Second Experiment result (with model update *via* transfer learning). I can conclude that both Figure 3.9 and Figure 3.10 indicate that our proposal outperforms the conventional regression models with or without transfer learning.

I conducted the statistical analysis to prove our proposed CW-CNN regression model significantly outperformed the conventional methods. The statis-

tical differences were obtained *via* T-test with Benjamini and Hochberg false discovery rate (BHFDR) [77] correction during multiple comparisons. According to Benjamini and Hochberg's work, FDR is a guideline for controlling relative errors, and defined as:

$$FDR = E[FDP] \tag{3.4}$$

Namely, FDR is the expectations (E) of false discovery proportion (FDP), and FDP indicates the proportion of false rejections out of all rejections. BHFDR is a method to control FDR. For the m hypotheses, the corresponding p-values are obtained. Then process the following steps to control FDR $\leq \alpha$ (generally α =0.05 or 0.01):

- Sorting the p-values:

$$p_1 \leq ... \leq p_m$$

- Defining K

$$K = max\{j : p_j \leq \alpha \frac{j}{m}\}$$

- If $p_m \leq p_K$: Reject all H_i $(i = 1, 2, ..., K)$

In this study, $m = 5$. As shown in Figure 3.10: from the left figure, in Initial Experiment, mean CC value of CW-CNN regression model is 0.8803 $\pm$ 0.0247, which is significantly higher than LR ($p = 0.021 < 0.05$), SVR ($p = 0.0016 < 0.01$), and KNN ($p = 0.0337 < 0.05$); from the right figure, in Second Experiment, mean CC value of CW-CNN model is 0.9133 $\pm$ 0.0175, significantly outperforms LR ($p = 0.0044 < 0.01$), SVR ($p = 0.0004 < 0.01$), KNN ($p = 0.031 < 0.05$) and DT ($p = 0.015 < 0.05$).

I designed only one convolutional layer to obtain higher estimation accuracy, and proved that it outperformed the conventional methods. Without using multiple layers or more complex deep learning structures, which makes the estimation of 3-DOFs joint angles more efficient.

3.4.4 Motion Pattern

I assumed the proposed CW-CNN regression model can learn the motion patterns information based on sEMG signal. Figure 3.11 shows the geometry plot of S1, which includes both sides of the forearm. According to the actual bipolar multi-array electrode sleeve (Figure 2.3), the left border of flexor side should be connected to the right border of extensor side, while the right border of flexor side should be connected to the left border of extensor side.

In this section, I will combine the geometry plot (use S1 as example) with human forearm anatomy for discussion to explain why I think the proposed hat the proposed model can obtain information on motion patterns from FP extracted through channel-wise filters. Figure 3.15—3.17 show the comparison result between geometry plot (motion pattern) with human forearm anatomy of the corresponding DOF motions, respectively.

Discussion of WF/WE Motion

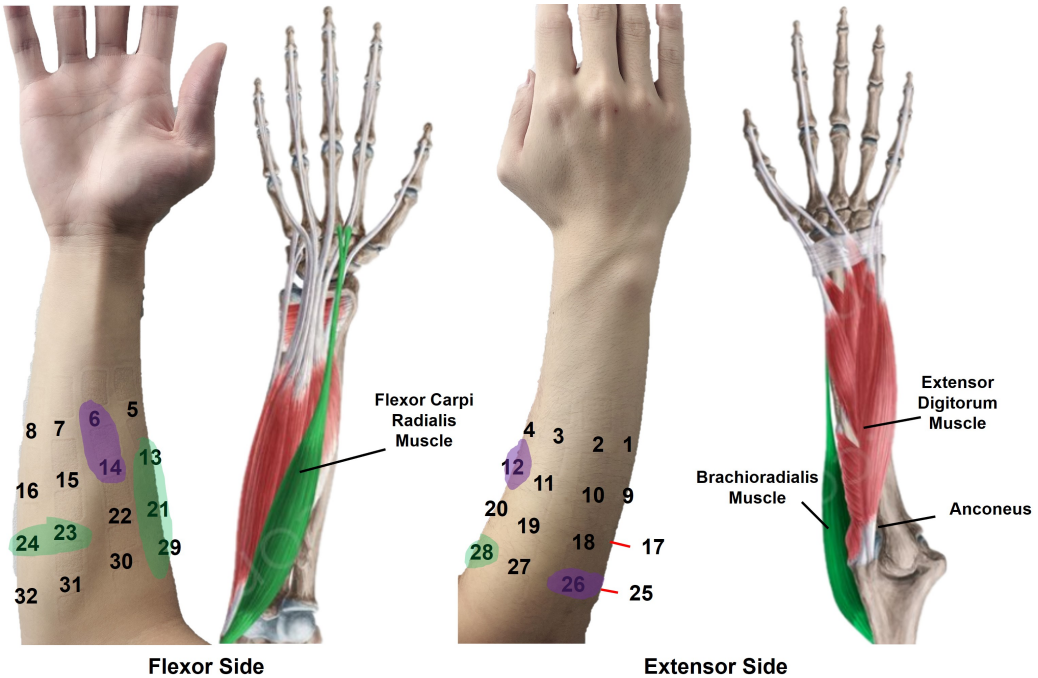


Figure 3.15: *Motion Pattern Discussion on WF/WE. The motion pattern based on the geometry plot result of participant S1 in Second Experiment (Figure 3.11B). Left part of each side shows geometry plot result; Right part of each side shows the human anatomy figures, which refer to Kenhub [79]. From geometry plot: green area denotes WF motion pattern, purple area denotes WE motion pattern.*

When WF and WE motions are performed, from geometry plot (Figure 3.11B, WF/WE part), both sides generated WF/WE motion pattern. WF motion occurred on two areas: channel 13, 21, 29 and 28, and channel 23, 24, while WE motion occurred on channel 6, 14, 12, 25, 26. From the human forearm anatomy, we know that WF motion mainly produced by the flexor carpi radialis muscle and brachioradialis of the forearm, and WE motion mainly produced by the anconeus, extensor digitorum muscle and so on. With the geometry plot result and the knowledge of human forearm anatomy, I only need to compare them to check if the parameters in FC layer reflect the correct motion pattern.

Figure 3.15 shows the motion pattern with human forearm muscle anatomy. The motion patterns based on the geometry plot of participant S1 in Second Experiment, the green area indicates WF motion pattern, while purple area indicates WE motion pattern. From this figure, we can find that: for WF motion, area of channel 13, 21, 29 and 28 correspond to the brachioradialis and flexor carpi radialis muscle of human forearm, and channel 23, 24 actually near the flexor carpi radialis muscle; for WE motion, channel 25, 26 correspond to anconeus, channel 12 near extensor digitorum muscle.

Thus, I can conclude that the motion pattern of WF/WE is correct, and actually when I perform the WF and WE motion, the mentioned areas in forearm show the corresponding motion activatity.

Discussion of WP/WS Motion

When WP and WS motions are performed, the forearm muscle contraction and relaxation happened, and pronator teres and supinator muscle will cross each other. From the geometry plot in WP/WS part in Figure 3.11, both sides show wrist pronation and supination. Compared to WF/WE motion, WP/WS motion generated from deep layer muscle, thus, it is challenging to recognize these two motions *via* machine learning.

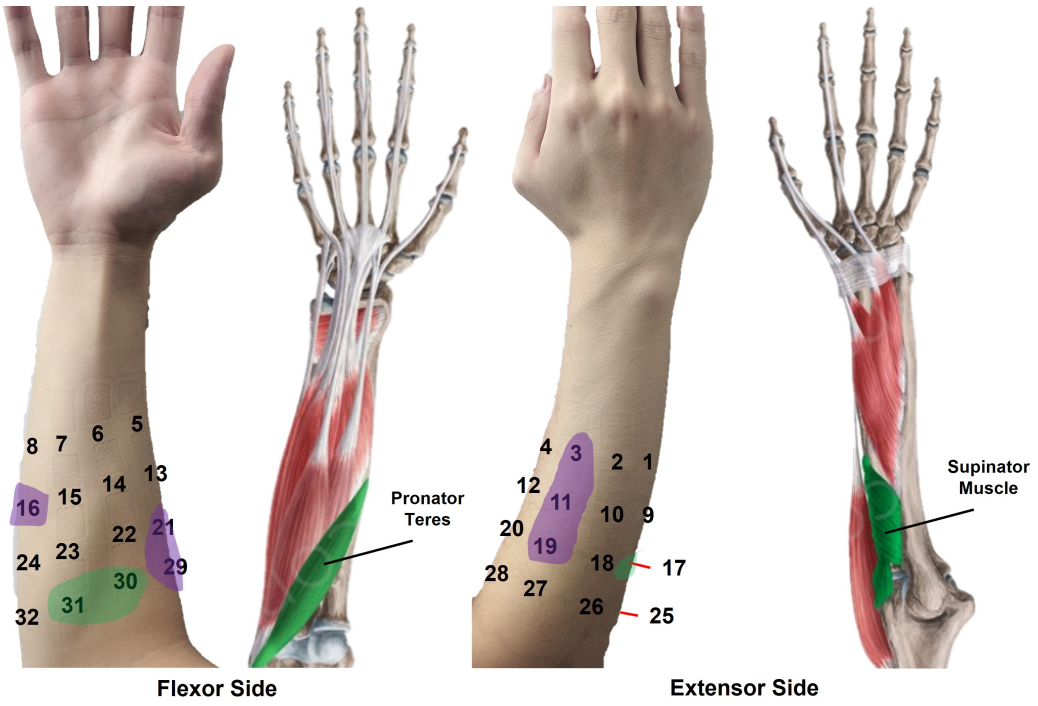


Figure 3.16: *Motion Pattern Discussion on WP/WS. The motion pattern based on the geometry plot result of participant S1 in Second Experiment (Figure 3.11B). Left part of each side shows geometry plot result; Right part of each side shows the human anatomy figures, which refer to Kenhub [79]. From geometry plot: green area denotes WP motion pattern, purple area denotes WS motion pattern.*

From Figure 3.16, we can see the comparison between geometry plot of S1 and corresponding forearm muscle anatomy. The left figures of each side show motion pattern area, the green area indicates the muscle activation area of WP motion, and purple area indicates the WS motion. WP occurred mainly on channel 30 and 31, which corresponds to the position of pronator teres muscle; WS occurred mainly on channel 21, 29 (correspond to supinator muscle) and channel 3, 11, 19 (muscle area involved at the upper end of the supinator).

For the reason that WP/WS motion contains the interactive motion between two bones,therefore, the muscle activity generated also on the opposite side. From this result, channel 17 shows the opposite position of pronator teres (channel 30 and 31), also contributes to WP motion; channel 16 shows the opposite position of supinator muscle (channel 21, 29, 3, 11, 19), contributes to WS motion. Therefore, I conclude the motion patterns of WP/WS to be appropriate.

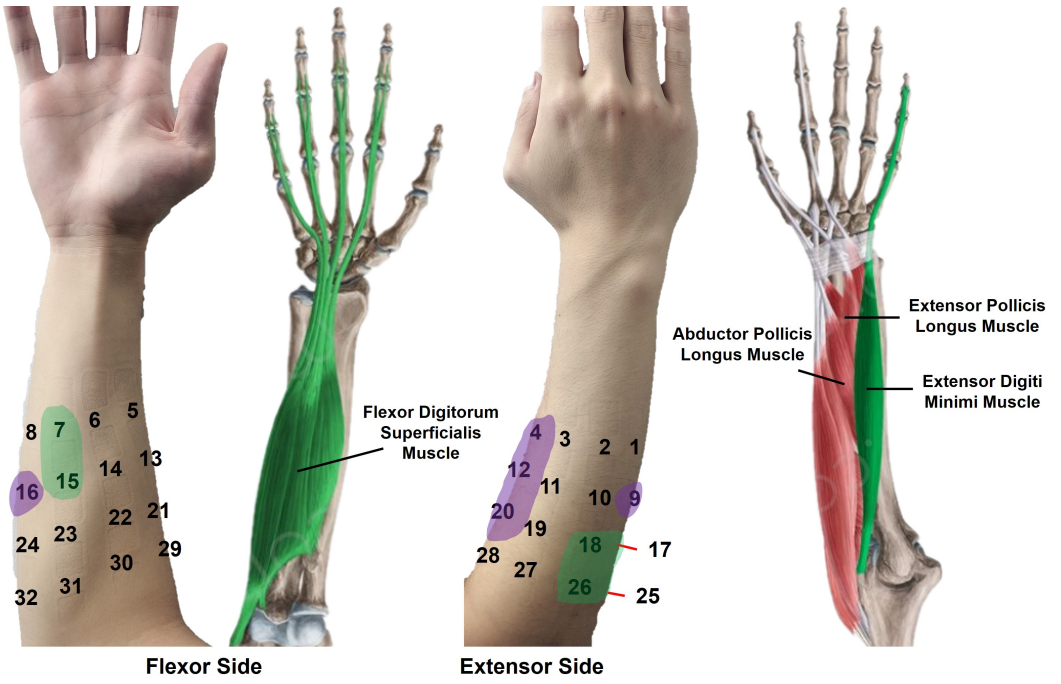


Figure 3.17: *Motion Pattern Discussion on HG/HO. The motion pattern based on the geometry plot result of participant S1 in Second Experiment (Figure 3.11B). Left part of each side shows geometry plot result; Right part of each side shows the human anatomy figures, which refer to Kenhub [79]. From geometry plot: green area denotes HG motion pattern, purple area denotes HO motion pattern.*

Discussion of HG/HO Motion

The difference between HG/HO and previous motions (WF, WE, WP, WS) is that, hand motion are generated from both forearm superficial and deeper muscles which may lead to finger motions (we can consider HG/HO are finger motions).

Figure 3.17 shows the motion pattern with human forearm muscle anatomy. According to forearm anatomy, HG motion is produced mainly from the flexor digitorum superficialis muscle and its opposite side. For HO, the muscle area is more complex, it is a hybrid motion produced by anconeus and some mucles in extensor side (such as extensor pollicis longus muscle, extensor digiti minimi muscle, abductor pollicis longus muscle, *etc.*).

From the geometry plot result (HG/HO part in Figure 3.11) or left part of each side of Figure 3.17, channel 7 and 15 correspond to motion HG, which belong to flexor digitorum superficialis muscle area. And due to muscle con-

traction, the opposite side (area of channel 17, 18, 25, 26) also contributes to HG motion; channel 4, 12, 20 correspond to abductor pollicis longus muscle, channel 9 and 16 belong to extensor digiti minimi muscle, all of them contribute to HO motion. Although it is difficult to evaluate the motion pattern of HO motion, I can still conclude that motion pattern of HO appears in forearm extensor side. When HG/HO motion was perform, muscle area of WF/WE and HG/HO motions activated, this is the effect of multiple layers of muscles, this explain why HG/HO motion appeared while doing WF/WE motion (or doing HG/HO motion will also appear WF/WE motion) (Figure 3.6 or Figure 3.7).

3.5 Limitations and Future Work

The research in this Chapter is mainly focus on offline analysis, I set 500 ms as sliding window length to segment dataset as CW-CNN input. However, the size of sliding window was not set to be suitable for real-time estimation. In real-time control, the estimation window lengths should be set from 50—400 ms [80], the 500 ms window might generate delay, thus, I revised the window length to 120 ms for real-time estimation, and I will introduce this study in the future (next Chapter).

Moreover, in this Chapter, I only discuss single motions (participants performed each specified motion without doing another motions). This research would be more interesting if the model could learn mixture movements (*e.g.*, WF+HG or WF+WP+HG, *etc.*) and predict multiple mixture movements with high *CC* results.

In the future (next Chapter), I used the 120 ms sliding to obtain real-time sEMG signal, and applied the CW-CNN regression model to our proposed control system, estimated joint angles of 13 daily motions (including single motions and mixture motions) in real-time. The estimated joint angles were sent to control a virtual hand model.

3.6 Conclusions

In the study of this Chapter, we aimed at providing a strategy for joint angles estimation of 3-DOFs based on forearm sEMG signal. To build dataset, Bipolar Multi-Array Electrode (32 channels) was used for sEMG data acquisition, and Perception Neuron Motion Capture System for angle data acquisition.

We completed the following works:

- Proposed a CW-CNN regression model for 3-DOFs joint angles estimation based on sEMG signal, and used correlation coefficient (CC) to evaluate the regression model accuracy
- Design the Initial Experiment and Second Experiment in different days
- Applied transfer learning with small amount of new dataset to keep model robustness in different days
- Designed channel-wise filter for convolutional layer to obtain force pattern as feature maps
- Explained the muscle activation and how the model achieve high accuracy *via* geometry plot.

Moreover, the model comparison results show that, the proposed CW-CNN regression model significantly outperformed traditional regression model (LR, SVR, KNN and DT), with or withou transfer learning.

In Chapter 4, we will apply the CW-CNN regression model to a real-time control system and complete target achievement control test and daily motions (both single motions and mixture motions) in real-time, and prove that it can be also applied for achieving prosthetic hand real-time control.

Chapter 4

Real-Time Control System for Virtual Hand

4.1 Overview

In the research presented in Chapter 3, CW-CNN regression model was proposed and discussed for 3-DOFs joint angles estimation. However, even though the offline performance looks nice, some significant real-time difference will be generated [81]. Therefore, in this chapter, I aimed at applying the CW-CNN regression model to build a real-time control system, evaluating the real-time performance including regression accuracy and computational delay, and demonstrating the possibility for application on virtual hand or prosthetic hand in the future.

The real-time control system environment was built in a laptop (MouseComputer CO.,LTD, Japan), the operation system was Microsoft Windows 10 Home 64 bit, with NVIDIA GeForce GTX 1660 Ti, the processor was Interl(R) Core(TM) i7-10750H CPU @ 2.60GHz, RAM 32.0 GB. In addition, the virtual hand model was made by Unity Engine (version 2021.3.2f1). The PC device for model initial training and transfer learning in this chapter is the same as the equipment presented in Chapter 3 (Section 3.3).

To build and propose a new real-time control system for myoelectric hand control, the following problems need to be solved:

- If the system has ability to estimate multi-DOFs motions simultaneously during real-time regression prediction?
- How to evaluate whether the computational latency is stable and meets the requirements for real-time use?
- How to smooth the joint angles to prevent the oscillations to improve performance?
- How to prove that it is real-time regression instead of real-time classifier?
- If the user can control the 3-DOFs of the virtual hand simultaneously to achieve any movement (joint angles within angular limitation)?

According to the previous chapter, the CW-CNN was only used to estimate single-DOF movement, without testing on simultaneous multi-DOFs motions. Thus, in this study more complex motions should be designed in experiments. Moreover, for practical applications in real-time environment, the analysis of latency is unavoidable. Simply averaging the latency data is an uncritical evaluation method, because the latency data can be erratic due to some reasons such as hardware problem or environmental disturbances. Before getting the average computational latency, the stability of the delay should be proved by some approaches, then we can discuss whether the latency meets the requirements of real-time control.

In addition, to clearly distinguish the proposed regression-based method from real-time motion classification, whether the system has ability to achieve some locations of specific target postures needs to be demonstrated. Thus, I considered to find a method that can track trajectory and emphasize the 3-DOFs motions simultaneously. Target achievement control (TAC) test has demonstrated to be a challenging method to test the real-time performance of motion classifier [81, 82]. However, as we know, the real-time regression is

more challenging than motion classifier during TAC test. In this chapter, TAC test was the first time to be used for evaluating the real-time control system to emphasize the regression motions.

In this research, a CW-CNN regression model-based real-time control system was proposed for virtual hand control, and it was used to estimate thirteen daily motions with 3-DOFs joint angles in real-time environment. The objective is to achieve precise regression control not only for single-DOF but also for multi-DOFs. A sliding window (the length was reduced from 500 ms to 120 ms for real-time processing) was used to pre-process the sEMG signals received through the LSL to normalized IEMG signals. The estimated joint angles were filtered *via* Adaptive Kalman Filter to remove oscillations to improve performance and protect motors for future control of actual prosthetic hand. The filtered joint angles were sent to a virtual hand as control commands. The proposed real-time control system has ability to estimate complex daily motions in real-time, any prosthetic hand or virtual hand can apply it for demonstration. During the real-time experiments, the estimated joint angles, measured joint angles and computational latency of the system were acquired by LSL, CC was used to evaluate the accuracy, and one-way analysis of variance (ANOVA) was used to analyze the stability of computational latency. Eight healthy participants were invited to conduct three tasks experiments on two different days, and an additional TAC test was designed to investigate the real-time regression effect. The experimental results proved the real-time performance meet the our expectation, in the future, the proposed real-time control system can be applied to an actual prosthetic hand.

4.2 Real-Time Control System

Figure 4.1 is the illustration of the proposed real-time control system with the experiment information. There are three experiments designed for each participant in different two days, namely, ***Exp.1*** in Day 1, ***Exp.2*** and ***Exp.3*** in Day 2, respectively. The ***Exp.1*** and ***Exp.2*** are the Initial Experiment and Second Experiment respectively in Chapter 3, this chapter will focus on the analysis of real-time control experiment ***Exp.3*** (blue dotted box). The details of experiments will be introduced in Section 4.3.2.

The real-time control system is a GUI tool, which written in Python3 and PyQt5 (PyPI, Python Sorfware Foundation. https://pypi.org/project/PyQt5/). It includes three modules: Data Processing Module, CW-CNN re-

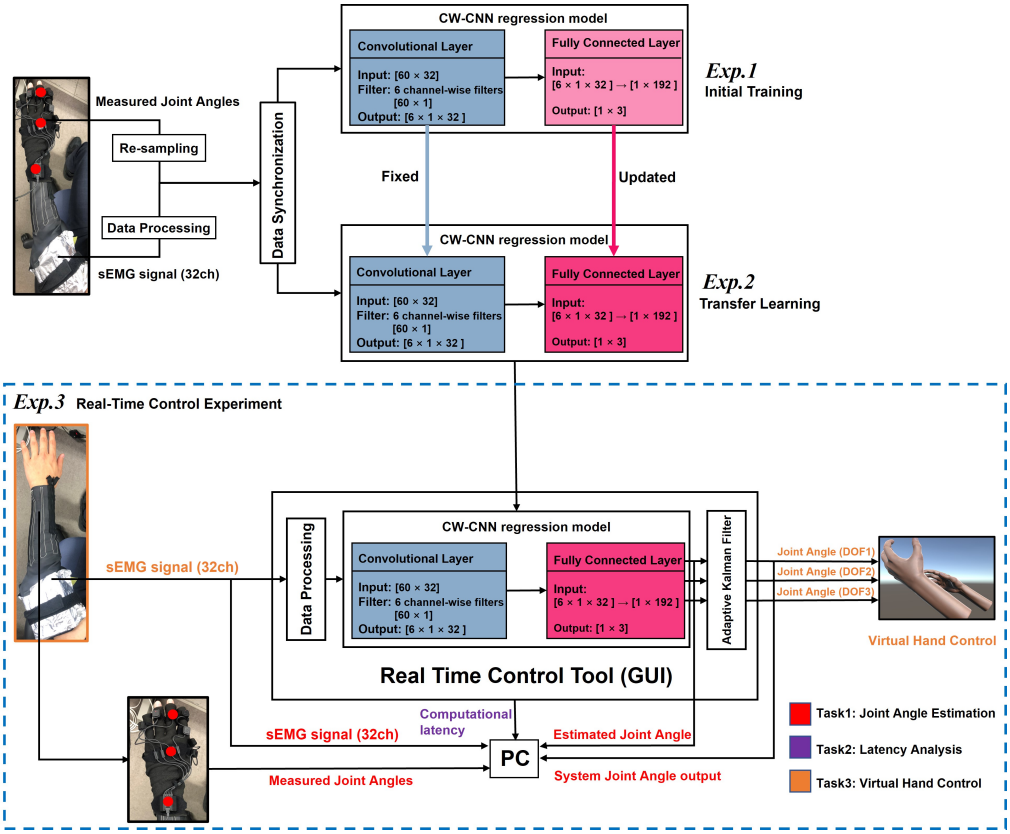


Figure 4.1: *Illustration of the proposed real-time control system. The real-time control consists of three sessions, where* ***Exp.1*** *and* ***Exp.2*** *show the model training procedures before real-time control (****Exp.3****). The* ***Exp.3*** *(part in the blue dashed box) shows the structure of the real-time control system, consists of four parts: Data Processing Module, trained CW-CNN regression model, Adaptive Kalman Filter, and a Virtual Hand model program.*

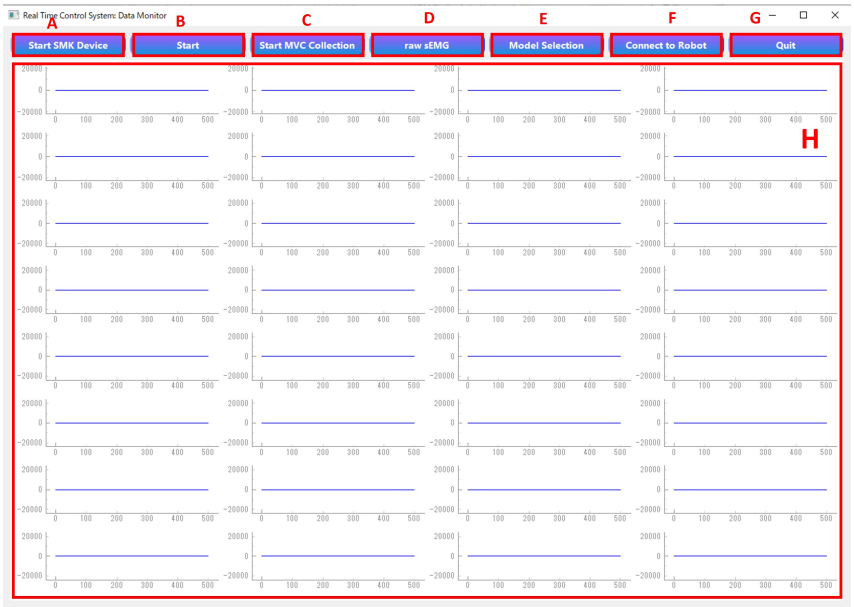


Figure 4.2: *Description of Real-Time Control GUI Tool. A to G are buttons that need to be clicked in order, H is the signal monitoring window. Please refer to Appendix B to obtain more detail information about the usage of the tool.*

gression model and Adaptive Kalman Filter. The Figure 4.2 shows the user interface of the real-time control system, and the following contents show the descriptions of each component A—H:

- Button A: **Start SMK Device**. Connect PC to multi-array electrode
- Button B: **Start**. Start to monitor the EMG signal after connecting successfully
- Button C: **Start MVC Collection**. Collect MVC from participants
- Button D: **raw sEMG**. Switch between raw sEMG and IEMG signals
- Button E: **Model Selection**. Select updated model and turn to "**Joint Angle Estimation**"
- Button F: **Connect to Robot**. Connect the output to prosthetic hand or Unity Program, turn to "**Data Collection**"
- Button G: **Quit**. Quit the GUI tool
- Interface H: EMG signal monitoring window (32 channels)

Figure 4.3 is a schematic diagram of the proposed real-time control GUI tool interacting with multi-array electrode, Perception Neuron Motion Capture System, LSL and virtual hand. The button "**Start SMK Device**" allows the PC which connected with multi-array electrode to send sEMG signals to LSL pipeline by sEMG outlet, while "**Start**" button allows the GUI tool to received sEMG signals from LSL pipeline by sEMG inlet, so that we can monitor the signals from the interface. Figure 4.3 also shows the operation steps of the Real-Time Control GUI tool. Finally, after connecting to virtual hand, we can click "**Data Collection**" button to obtain the real-time data for analysis, which is related to the three tasks shown in Figure 4.1, the GUI tool will push the corresponding data to LSL pipeline by outlets, and another PC will acquire the data from LSL pipeline by inlets. Moreover, the virtual hand will receive the trigger and angles from LSL pipeline via related inlets, so that users can control the virtual hand in real-time using their forearm sEMG signals.

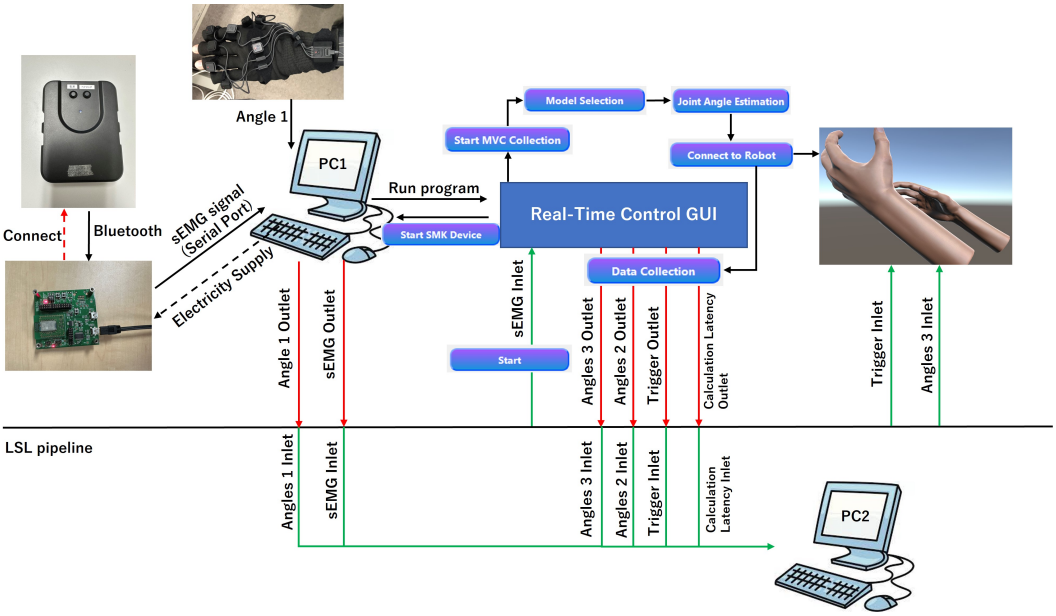


Figure 4.3: *Illustration of LSL interaction of the real-time GUI tool to other objects. Angle 1 indicates the measured joint angles using Perception Neuron Motion Capture System, Angle 2 indicates the estimated output of CW-CNN model, Angle 3 indicates the system output after processing by Adaptive Kalman Filter.*

To know the detail usage of the GUI tool for real-time control of prosthetic hand or virtual hand, please refer to Appendix B.

4.2.1 Data Processing Module

In real-time, sEMG signals were processed as normalized IEMG signals and used as input for 3-DOFs joint angles estimation. In this chapter, sampling frequency was not down-sampled from 500 Hz to 120 Hz, the window size was reduced to 120 ms, i.e., the raw sEMG signal in 32 channels can be expressed by [60×32] in each acquisition timing, where 32 is the channel numbers, $60 = 500Hz \times 120ms$. The real-time control system always collects the latest sEMG signals as input to Data Processing Module after the previous processing period, so that to prevent signal delay. Before the ***Exp.3***, MVC data from each participant were collected for signal normalization afterwards. The data processing steps can refer to Section 3.2.3 and Section 2.1.3. During real-time normalization, as distinct from the offline normalization in Chapter 3, IMVC did not be replaced with IEMG while using Equation 2.9—2.10, just used IMVC to normalize the signal directly. In real-time experiment, one of the sEMG data processing results is shown as Figure 4.4.

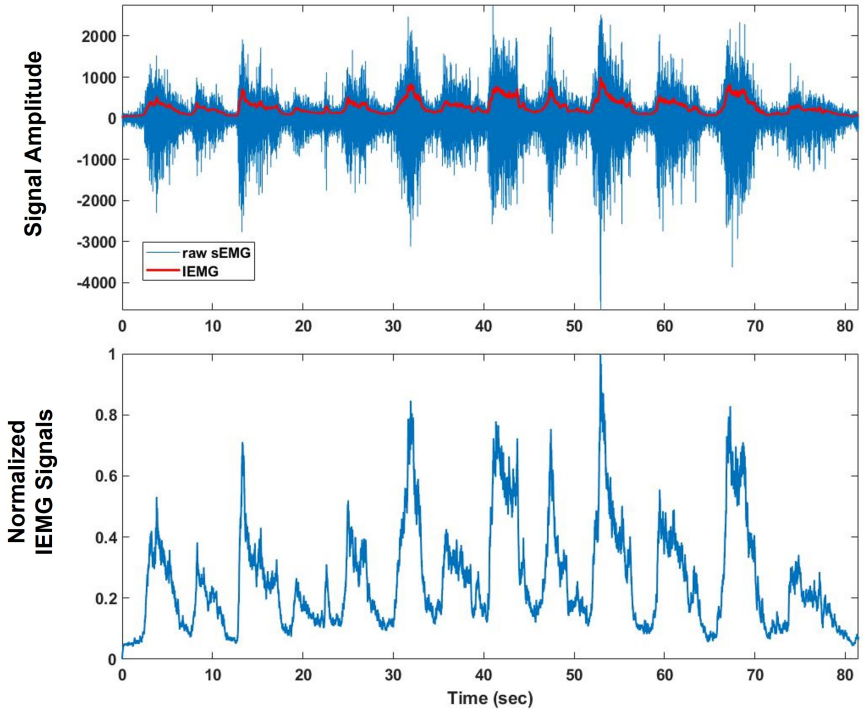


Figure 4.4: *Real-time processing results in the Data Processing Module (Sub.1, ch5). The top figure shows the process from the raw signal to the IEMG, the blue line represents the raw signal, and the bold red line is the IEMG signal. The bottom figure shows the results of the IEMG normalization process.*

Table 4.1: CW-CNN regression model training detail. The convolutional layer is denoted as *Conv layer* and the fully connected layer is *FC layer*. *LR* means learning rate, and *CV* is abbreviation of cross-validation.

Exp.	Training layer	LR	Validation	Epochs	Duration
Exp.1	*Conv.* layer FC layer	0.005	10-fold CV	10	≈30 min
Exp.2	FC layer	0.002	5-fold CV	5	≈2.5 min

4.2.2 CW-CNN Module

In Chapter 3, the proposed CW-CNN regression model (Figure 3.4) was used to estimate 3-DOFs joint angles in offline environment, and high performance CC was obtained with and without transfer learning. In this chapter, the CW-CNN was applied to the real-time control system to achieve virtual hand control. Because the window size was reduced to 120 ms, the input size [60×32] is the same as before, thus, there is no need to change the input size and structure of the model.

In ***Exp.1***, model of each participant was initially trained by 10-fold cross validation on 10-trial data. In ***Exp.2***, the model of each participant was updated *via* layer transfer with 5-fold cross validation on different days, 5-trial new data were needed. After the transfer learning, the updated model was used as the CW-CNN module to the proposed real-time control system for joint angles estimation directly in ***Exp.3***. From Figure 4.1, the color change of FC layer indicates the parameters update. The training detail information can be checked from Table 4.1.

The training was conducted using Pytorch 1.3.1 in another PC (MouseComputer CO.,LTD, Japan; GeForce RTX 2080 GPU, CUDA10.1 refer to Chapter 3). The convolutional layer is denoted as *Conv.* layer, learning rate is LR, and cross validation is CV. We can find that in different day, the transfer learning only cost about 2.5 min to update the model, the real-time experiment can start without waiting too long time.

4.2.3 Adaptive Kalman Filter

Kalman filter is commonly used to estimate the unknown variables more precisely with a series of data in engineering [83, 84]. In each iterative process, a new observation is obtained and the system state (x) and error covariance (P) are updated. Each iteration is only based on the previous iteration results and new measured input values, thus, the filter occupies very few computational resources.

The following formulas show the filtering process of conventional Kalman filter:

$$K(k) = \frac{P(k-1)}{P(k-1)+R} \tag{4.1}$$

$$x(k) = x(k-1) + K(k) \cdot (z(k) - x(k-1)) \tag{4.2}$$

$$P(k) = (1 - K(k)) \cdot (P(k-1)) + Q \tag{4.3}$$

where, at k time, the $K(k)$, $x(k)$, $P(k)$ and $z(k)$ are Kalman gain, expected filtered value, filter deviation matrix and observation value, respectively. Q and R are transition covariance matrix and observation covariance matrix, respectively. Therefore, to use conventional Kalman filter, we have to determine the preset Q and R values manually.

However, it is hard to evaluate the preset values are the optimal solutions as expectation, thus the adaptive adjustment of Kalman coefficients is needed. To achieve an Adaptive Kalman Filter, Kalman coefficient Q and R should be updated after each iteration before going through to next iteration.

Refer to [85], I designed the adjustment procedure as the following formulas:

$$Q(k+1) = Q(k) + \frac{1}{k+1}(\hat{Q}(k+1) - Q(k)) \tag{4.4}$$

$$R(k+1) = R(k) + \frac{1}{k+1}(\hat{R}(k+1) - R(k))) \tag{4.5}$$

where the $k+1$ indicates the next iteration timing, $Q(k+1)$ and $R(k+1)$

indicate the updated values that will be used in next iteration, $Q(k)$ and $R(k)$ are the coefficients used in k-th iteration, $\hat{Q}(k+1)$ and $\hat{R}(k+1)$ are the variances estimates using the previous Q and R, respectively.

Initial values of Q and R were set to 0.05 and 0.4, respectively, and according to Equation 4.4 and Equation 4.5, we were able to observe the asymptomatic convergence of the two coefficients (Figure 4.5).

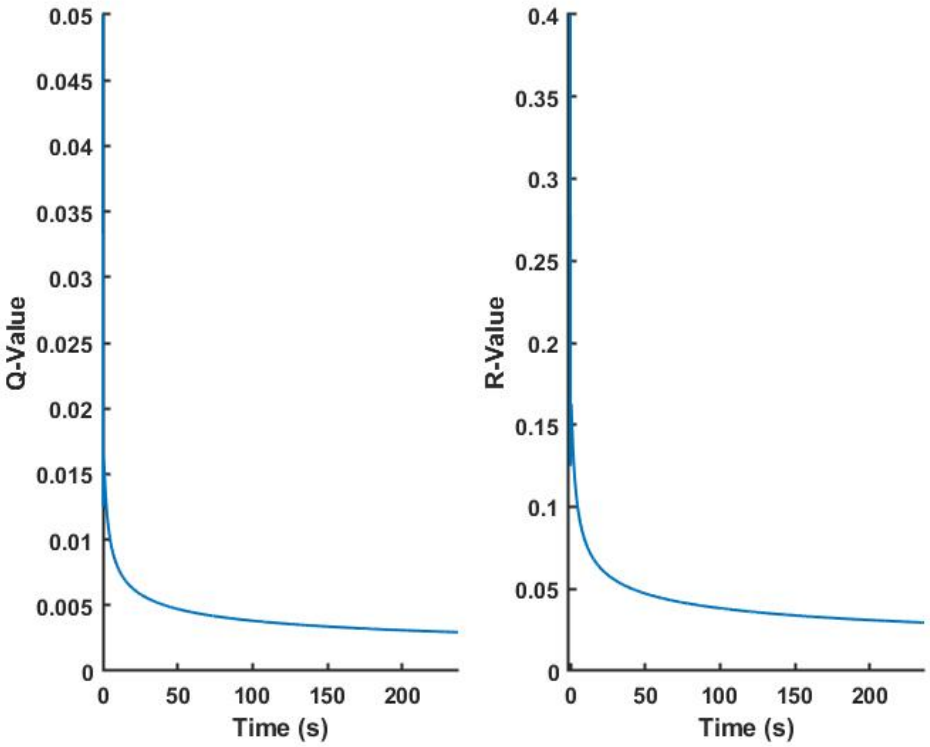


Figure 4.5: *Adaptive results of Kalman coefficients Q and R in real-time experiments.*

4.3 Methodology

4.3.1 Participants

Eight healthy participants (Sub.1—Sub.8) were invited to the experiments on different days in this study. They were guided to read the participant information sheet and provide the written informed consent to join the study before experiments. Due to the size limitation of the multi-array electrode sleeves, the choice of right or left hand depended on whether the forearm size of the participant fits better or not. The dataset from the participants were collected at the Tokyo Institute of Technology. Study protocol was approved by the ethics committee of the Tokyo Institute of Technology and was conducted in accordance with the Declaration of Helsinki. The participants' information can be checked in Table 4.2. In this study, all participants joined the real-time experiment ***Exp.1***—***Exp.3***. To prove the real-time regression motions, four of them were invited to join the target achievement control (TAC) test. About TAC test, please refer to Section 4.3.3.

Table 4.2: Information of each participants in this chapter. All of them participated in ***Exp.1***—***Exp.3***, and part of them were invited to join the TAC Test. M indicates male, F indicates female; ◯ denotes participated to the corresponding experiment.

Participant ID	Age	Handedness	Gender	***Exp.1***—***Exp.3***	TAC Test
Sub.1	22	Left	M	◯	-
Sub.2	25	Right	M	◯	-
Sub.3	25	Left	F	◯	◯
Sub.4	28	Left	M	◯	◯
Sub.5	26	Left	M	◯	◯
Sub.6	21	Left	M	◯	-
Sub.7	23	Right	M	◯	-
Sub.8	24	Right	M	◯	◯

4.3.2 Basic Experiment Design

From Figure 4.1, there are three steps for basic real-time control experiment in different two days for each participant, include two offline sessions (***Exp.1*** and ***Exp.2***) and one real-time session (***Exp.3***).

Exp.1 was designed to collect 10-trail dataset to train a new model initially for each participant, which is the same as the Initial Experiment in Chapter 3. ***Exp.2*** and ***Exp.3*** were designed for each participant on other days to control a virtual hand in real-time. In ***Exp.2***, small number of new datasets were acquired and model was updated *via* layer transfer. Then the updated model was applied to the proposed real-time control system, and participants were able to control a 3D virtual hand (***Exp.3***). In all sessions of the experiment, LSL was used to not only synchronize the IEMG signals and joint angles data as dataset for model training, but also transmit control command to virtual hand program.

After wearing the devices, participants sat in front of a screen during the experiment, and they were instructed to perform 13 daily motions which displayed on screen. The motions were shown as Figure 4.6, and the description of the motions were summarized as Table 4.3. The daily motions include five single-DOF motions (M1—M5), four double-DOF motions (M6—M9) and four triple-DOF motions (M10—M13). M0 indicates the central motion (rest motion).

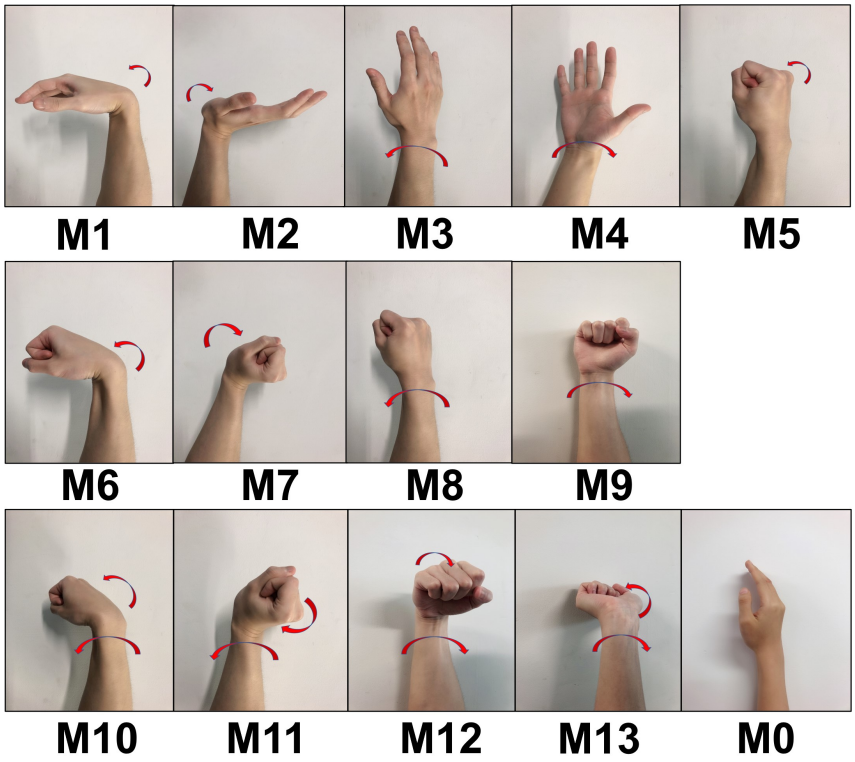


Figure 4.6: *Daily movements pictures (right hand). The details of M0 M13 are shown in Table 4.3. M1—M5 are single motions, M6—M9 are double-mixture motions, and M10—M13 are triple-mixture motions, M0 denotes the central position.*

Table 4.3: Specified daily motions that should be performed in the experiment.

Motion ID		Contents
M0		Rest
Single Motions	M1	Wrist Flexion (WF)
	M2	Wrist Extension (WE)
	M3	Wrist Pronation (WP)
	M4	Wrist Supination (WS)
	M5	Hand Grip (HG)
Double-Mixture	M6	WF+HG
	M7	WE+HG
	M8	WP+HG
	M9	WS+HG
Triple-Mixture	M10	WF+WP+HG
	M11	WE+WP+HG
	M12	WF+WS+HG
	M13	WE+WS+HG

Offline Experiments: *Exp.1* & *Exp.2*

In ***Exp.1***, there were totally 10 trials. Participants were instructed to perform M1—M13 as screen shown in each trial, and after each trial, approximately 2 minutes rest was needed to prevent muscle fatigue. The 10-trial datasets were used to initial train the model by 10-fold cross validation, which is different from Chapter 3.

In different day, ***Exp.2*** was conducted. There were 5 trials in total, participants repeated the same experiment procedure as ***Exp.1***, which from M1—M13 in each trial. After data acquisition, new datasets were used to calibrate the model *via* transfer learning by 5-fold cross validation. From Table 4.1, this process lasted about only 2.5 minutes, thus, the participants did not need to wait too long before proceeding to ***Exp.3***, ensured that the sEMG signal remained the same quality.

For offline experiments, after measuring the sEMG signals and joint angles data, sEMG signals were processed as normalized IEMG signals and synchronized with joint angles *via* the time stamps of LSL system. Due to the different sampling frequency of the two signals, and considering to avoid loss of sEMG signal information as much as possible, joint angles were up-sampled from 120 Hz to 500 Hz to match IEMG signals.

Real-Time Experiment: *Exp.3*

After the transfer learning, the updated regression model for the participant was applied to the real-time control system which proposed in this study. To start real-time control, we need to run the GUI tool, start receiving the sEMG signals, collect MVC for signal normalization, load the trained model into the system, then we can check the estimated 3-DOFs joint angles in real-time. If the data has no problem, then we can connect the system to Unity virtual hand or an actual prosthetic hand for control. In this experiment, I designed three tasks (Figure 4.1) of this session:

- Task 1: Joint Angles Estimation;
- Task 2: Computational Latency Analysis;
- Task 3: Virtual Hand Control Demonstration.

For Task 1, participants wore the Perception Neuron Motion Capture System glove for data collection. In this task, the measured angles (from motion capture), raw sEMG signals (from multi-array electrode), estimated joint angles (output of CW-CNN Module) and system output angles were acquired using LSL for evaluation. For the reason that the study in this chapter also regression prediction, thus Pearson CC [86–88] was used as evaluation metric again (Equation 3.2).

For Task 2, the latency data of each calculation process were measured. The following steps explain the order of computational latency: (1) receive raw sEMG signal using 120 ms sliding window → (2) process raw sEMG signals to normalized IEMG signals within the window → (3) CW-CNN regression model calculation → (4) Adaptive Kalman Filter calculation. Please refer to the following pseudo-code for the specific process. For each participant, the latency were saved as data streaming during this task, participants kept performing any motions lasted 5 minutes, the computational latency data were saved as 5-minute stream for statistical analysis.

Algorithm 1 Calculation routine for calculation latency

function $CalculationLatency(\ )$
  $time_{start} \leftarrow GetCurrentTime()$
  $EMG_{raw} \leftarrow GetEMG()$
  $IEMG \leftarrow Filter(abs(EMG_{raw}))$
  $IEMG_{norm} \leftarrow Normalization(IEMG)$
  $Angles \leftarrow CWCNN(IEMG_{norm})$
  $Output_{system} \leftarrow AdaptiveKalmanFilter(Angles)$
  $time_{end} \leftarrow GetCurrentTime()$
  return $time_{end} - time_{start}$
end function

For Task 3, participant controlled the 3D virtual hand in Unity to perform the daily motions M1—M13. Moreover, I designed TAC Test to emphasize the

real-time regression motions in multi-DOFs, which will be presented in next subsection.

4.3.3 Target Achievement Control (TAC) Test Design

In ***Exp.3***, M1—M13 were performed as control target by participants. However, the 13 daily motions started from central position to corresponding maximum angles, this mislead the readers that maybe the task is motion classification, and the CW-CNN model is a motion classifier. Therefore, it is important to prove the regression model in this study, thus to prove the difficulty of the real-time regression compared to real-time classification.

Simon et al. [81] presented the target achievement control (TAC) test to simulate the possible performance of myoelectric prosthetic hand in real life. During TAC test, participants had to control the virtual hand in multi-DOFs to touch the target postures and dwell for the specified waiting time (or dwell time). In this task, participants would experience several unexpected regression control feedback, more time is needed to adjust their postures, so that increased the completion times, even decreased the completion rates. According to [81], parameters such as acceptance tolerance, dwell time and trial time limitation should be focused on. In addition, in this thesis I used completion rates, completion duration and real-time trajectory as evaluation metrics to analyze the effect of the proposed real-time control system.

According to Table 4.2, four of the participants (Sub.3, Sub.4, Sub.5 and Sub.8) were invited to TAC test. Figure 4.7 is the illustration of the TAC tasks in this thesis, participants needed to control the virtual hand with only multi-array electrode sleeve to touch the target posture (grey hand). After the target is touched within the acceptance tolerance, the posture hand turns green color, then the controlled virtual hand needs to keep the posture and stays for the dwell time until the posture is completed so that target changes to next posture. To challenge the difficulty of real-time regression, the angles were

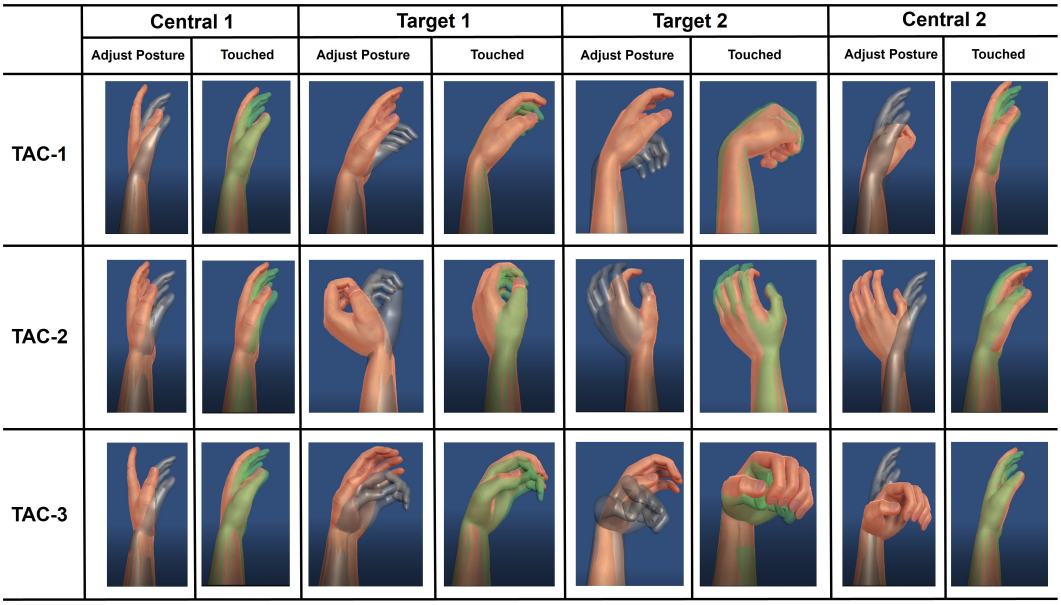


Figure 4.7: *Illustration of the target achievement control (TAC) test designed in this paper. The target posture is grey color, it will turn green after being touched. When the green color lasts for 0.5 s dwell time, the posture is completed and the target hand changes to the next posture. For detailed TAC test parameters and the 3-DOFs target joint angles for each group of TAC tests, please refer to the Table 4.4.*

Table 4.4: TAC test parameters and target motion details. The actual motion performance of each posture for each trial can be checked in Figure 4.7.

Parameters		**Setting**
DOF Number		3
Dwell Time		0.5 s
Target Tolerance		± 5°
Trial Time Limitation		30 s
TAC Test	**Posture**	**Joint Angles**
TAC-1	Central 1	DOF1: 0°; DOF2: 0°; DOF3: 0°
	Target 1	DOF1: 15°; DOF2: 15°; DOF3: 15°
	Target 2	DOF1: 30°; DOF2: 30°; DOF3: 30°
	Central 2	DOF1: 0°; DOF2: 0°; DOF3: 0°
TAC-2	Central 1	DOF1: 0°; DOF2: 0°; DOF3: 0°
	Target 1	DOF1: -15°; DOF2: 15°; DOF3: 15°
	Target 2	DOF1: -30°; DOF2: 30°; DOF3: 30°
	Central 2	DOF1: 0°; DOF2: 0°; DOF3: 0°
TAC-3	Central 1	DOF1: 0°; DOF2: 0°; DOF3: 0°
	Target 1	DOF1: 15°; DOF2: -15°; DOF3: 15°
	Target 2	DOF1: 30°; DOF2: -30°; DOF3: 30°
	Central 2	DOF1: 0°; DOF2: 0°; DOF3: 0°

changed in 3-DOFs simultaneously in each posture. The acceptance tolerance was set to ± 5° which the same as [81]. I reduced the dwell time to 0.5 s due to the difficulty of real-time regression, and shortened the limitation of trial time to 30 s accordingly, to keep the difficulty balance of the TAC task. I designed three trials, they are TAC-1, TAC-2 and TAC-3 respectively. Each participant was required to complete TAC-1, TAC-2 and TAC-3 test. From Figure 4.7, for each trial, the target posture started from the central position *Central 1* and ended with the central position *Central 2* after all postures were completed. The important TAC parameters and angular information of the 3-DOFs for each target were summarized as Table 4.4.

In this chapter, I used completion rate curve to evaluate the TAC performance. Within the 30 s trial limitation, the completion rate can be calculated at each time point by recording the number of completed postures, so that the curve can be plotted. The following formula shows the calculation of completion rate:

$$CR(t) = \frac{N_{completed}(t)}{N_{postures}} \tag{4.6}$$

where the CR indicates the recorded completion rate, t is time point. CR will be calculated from the beginning to time limitation in each trial. The $N_{completed}(t)$ indicates the completed postures number at time t. According to Figure 4.7 and Table 4.4, for all of the three trials, the posture number $N_{postures} = 4$.

4.3.4 One-Way ANOVA

One-way analysis of variance (ANOVA) is an appropriate method for comparison of more than two groups [89]. In the Task 2 of this chapter, computational latency was series data in 5 minutes for each participant, and they were divided into five groups, each group means a sequence of latency data per minute. In this thesis, in order to show the stability of the proposed sys-

tem, I considered to use one-way ANOVA to prove that there is no significant differences between each minute for each participant, and between different participants.

The aim of ANOVA is to evaluate whether the mean value of each group of data is the same. The null hypothesis of ANOVA is

$$H_0 : \mu_A = \mu_B = \mu_C = ... \tag{4.7}$$

where μ means expectation in Statistics. If there is no significant difference in the performance of these groups of participants and latency data per minute for each participant, then the data are multiple random samples of the same population. In another words, I want to analyze whether there is a group of data whose performance is so distinctive that it is not from the same group.

ANOVA has the following three main assumptions: (1) Homogeneity of variance; (2) The data groups follow a normal distribution; (3) Each sample is independent. There are two important concepts of ANOVA,

- Mean squared between (MSB), namely, variance of each group relative to the total group

$$MSB = \sigma^2 \tag{4.8}$$

- Mean squared error (MSE), namely, the variance of each distribution itself

$$MSE = \frac{\sum \sigma_i^2}{n} \tag{4.9}$$

now we can evaluate the MSB and MSE *via* F-statistics, *i.e.*,

$$F = \frac{MSB}{MSE} \tag{4.10}$$

As a result, the following three situations may occur:
(1) MSB much higher, F relatively larger: at least one of the distributions is distant compare to others. Thus, we cannot conclude that all the distributions

have the same mean value, so that reject H_0;

(2) MSE much higher, F relatively smaller: maybe the mean values of each group are relatively concentrated, or the variance of each group is large. Thus, we cannot reject H_0;

(3) $MSB \approx MSE$, F relatively smaller: it is difficult to find a particular group of distributions, thus we cannot reject H_0.

Therefore, when $MSB = MSE$, $F = 1$, we can say there is very little difference between the groups; when MSB becomes much larger than MSE, F becomes larger, and according to F-distribution, p-value turns to smaller. The above discussion is the principle of ANOVA [90].

In this thesis, I used $anova1()$ function in MATLAB (The MathWorks, Inc., USA) to calculate p-value of the latency groups for analysis. The significance level was set to 0.5 [91] to judge the stability of computational latency of the five groups (per minute in one group of each participant) and the eight participants.

- between minutes: if the system is unstable, the computational latency also becomes more pronounced as time goes, this step is to prove that the stability of the proposed real-time control system during usage of each participant.

- between participants: because participants were invited to the expeirments in different days, this step is to prove the stability of the system in different days.

If the p-value > 0.5, it can be proved that there is no significant difference between the compared groups, *i.e.*, the computational latency of the proposed real-time control system did not change with time, they were stable.

4.4 Experimental Results

4.4.1 Real-Time Joint Angle Estimation

In the Task 1 of ***Exp.3*** (Figure 4.1), there are four experimental data were collected *via* LSL:

- Raw sEMG signals by multi-array electrode
- Estimated 3-DOFs joint angles by CW-CNN model
- System output which processed by Adaptive Kalman Filter
- Measured 3-DOFs joint angles by motion capture

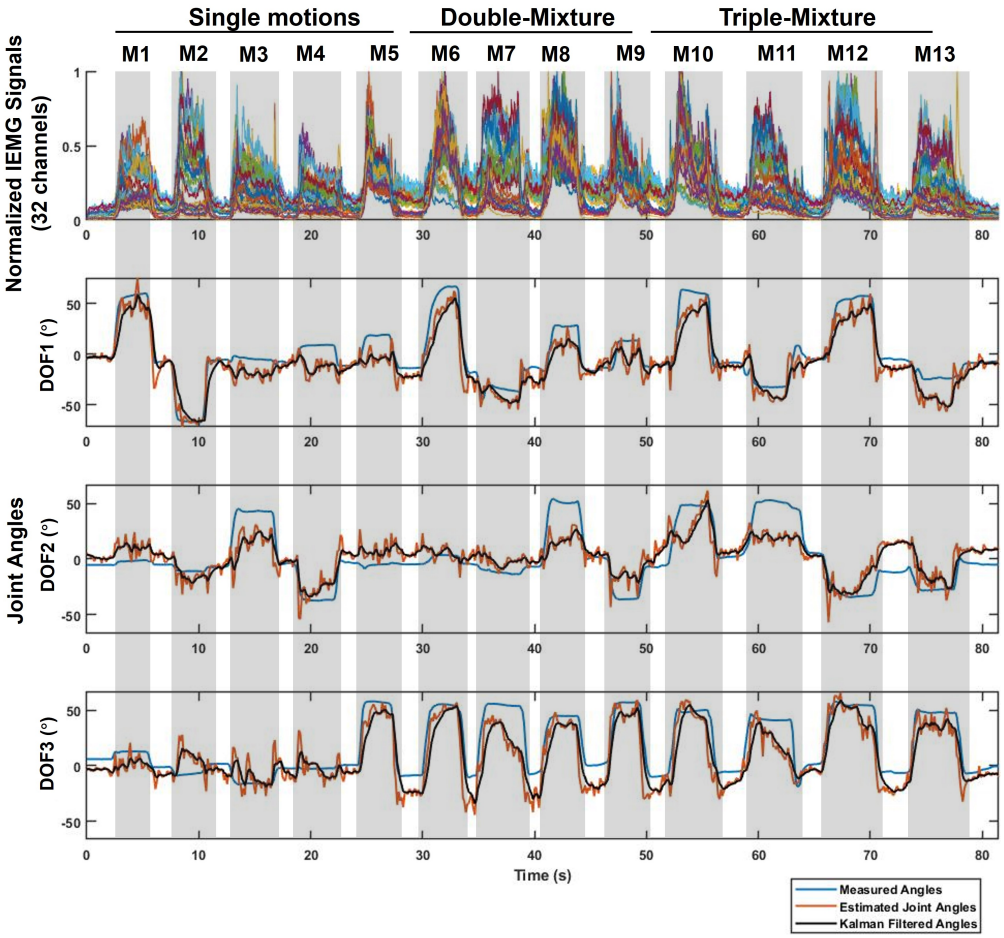
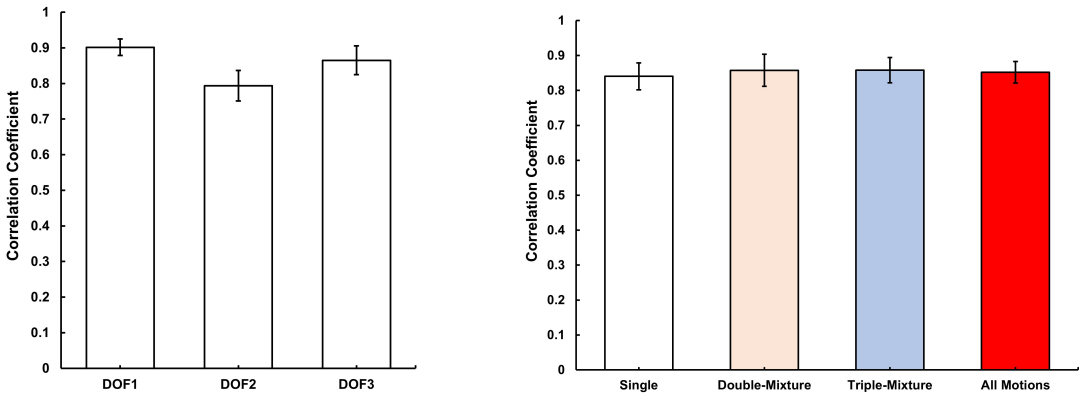


Figure 4.8: *Example real-time experimental results in Task 1 for 3-DOFs joint angles (Sub.1, left hand). Blue line: measured joint angles; Red line: the estimated joint angles from CW-CNN regression model; Black line: the smoothed output via Adaptive Kalman Filter. The CC between black and blue line are 0.94 (DOF1), 0.80 (DOF2) and 0.90 (DOF3) respectively.*

The CC between the system output and measured angles were calculated for each 3-DOFs respectively for evaluation. One of an example real-time estimation result of Sub.1 is shown as Figure 4.8. The top figure is the 32 channels normalized IEMG signals, the motion activation (M1—M13) can be checked and correspond to the joint angles. For joint angles, the blue curves indicate the measured angles, red curves indicate the estimated angles by CW-CNN regression model, and black curves indicate the joint angles processed by Adaptive Kalman Filter. It is obvious that there are some oscillations occurred in red curves, if this results are sent to prosthetic hand, there is a risk that the motors may be damaged. And the black curves shown more smooth results, and meet the requirements of expected joint angles estimation. This result proved the importance to use Adaptive Kalman Filter.

Since the objective is to propose a real-time control system in the study of this chapter, therefore, only the system output (black curve) should be compared with measured joint angles (blue curve) to obtain CC for evaluation. The CC values of the 3-DOFs joint angles were 0.94, 0.80 and 0.90 respectively.



(a) CC values over DOFs (b) CC values over different motion types

Figure 4.9: *Real-time experimental results of all participants for system evaluation. Error bar denotes the standard error.*

Figure 4.9(a) shows the average CC results over the 3-DOFs joint of the eight participants, and CC results were 0.90 ± 0.02 (DOF1), 0.79 ± 0.04 (DOF2) and 0.87 ± 0.04 (DOF3), respectively. The average CC for all participants for single-DOF, double-DOFs (double-mixture) and triple-DOFs (triple mixture) are

compared as shown in Figure 4.9(b). The CC results were 0.84 ± 0.04 (single), 0.86 ± 0.05 (double-mixture) and 0.86 ± 0.04 (triple-mixture), respectively. For all motions, average CC value for all participants is 0.85 ± 0.04. This results show that the proposed real-time control system is not only good for single motions (Chapter 3), but also meets the requirement of mixture motions (double-DOFs and triple-DOFs) in real-time.

4.4.2 System Computational Latency

The calculation process of "raw sEMG signals → normalized IEMG signals → CWCNN estimated joint angles → Kalman Filtered joint angles" consumes a certain amount of computing time, which may obtain some delay of real-time control. To analyze the computational latency, time consumptionn series data were collected during Task 2 in ***Exp.3***. One-way ANOVA was used to perform statistical analysis of computational latency, as described in Section 4.3.4. Figure 4.10 shows the analysis results, the p-value always higher than 0.5, not only for each participant, but also for the comparison between different days, proved the stable latency generated by the proposed system.

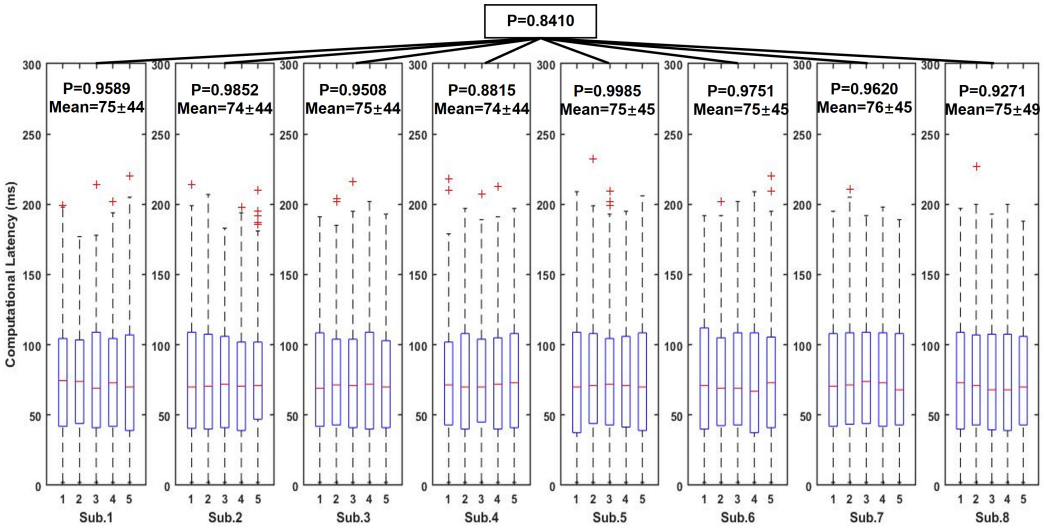


Figure 4.10: *Computational latency analysis results using one-way ANOVA. The red mark of the box plot indicates the outliers of the data*

4.4.3 Daily Motions Demonstration

In this study, a 3D virtual hand model was built in Unity engine, which can perform the required 3-DOFs movements for real-time control demonstration. The virtual hand can receive the joint angles by recognizing the system outputs from LSL pipeline in real-time (Figure 4.3). In order to record separate videos for left- and right-handed demonstration, both hands' program were completed as shown in Figure 4.11(A).

In Task 3, participants were instructed to complete the 13 specified motions, Figure 4.11(B) shown some video frames of the demonstration. More completed demonstration videos can be checked, please refer to Appendix C.1 to find the link.

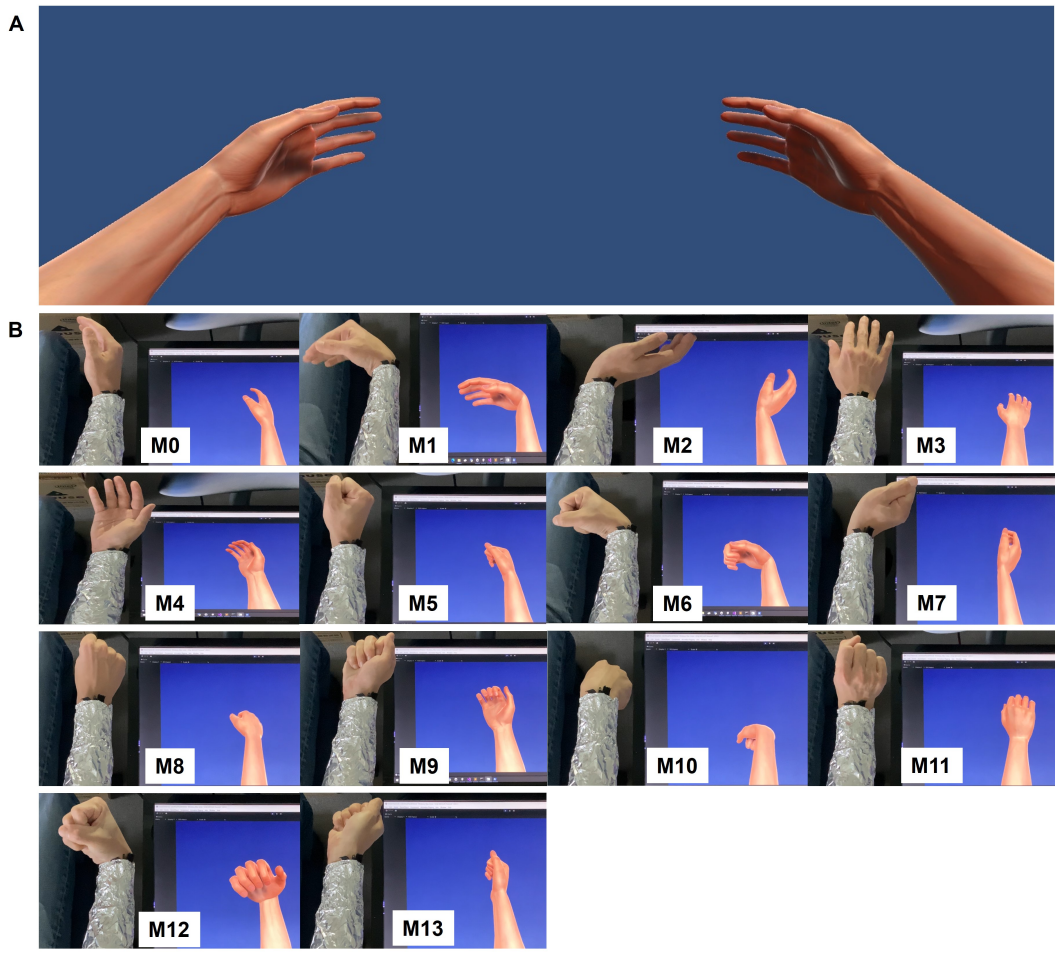


Figure 4.11: *Virtual hand real-time demonstration in Task 3. (A) Virtual hand 3D model built for left and right hand. (B) Real-time virtual hand control to perform the thirteen daily motions (Sub.8, right hand). Motion capture was removed, participants controlled the virtual hand only using their forearm sEMG signals.*

4.4.4 TAC Test

The demonstration videos of TAC test can be checked, please refer to Appendix C.2.

During TAC test, the four participants completed the TAC-1, TAC-2 and TAC-3 trial within the time limitation (30 s), Figure 4.12 is the average completion rate curves, where the solid line denotes the performance of TAC-1 trial, dashed line denotes the performance of TAC-2 trial, and dotted line denotes the performance during TAC-3 trial.

In addition, the trajectory plots of TAC-1 to TAC-3 of the 3-DOFs for one of the participants (Sub.4) were shown as Figure 4.13, the green area is the dwell duration, grey area indicates the participant was adjusting the posture, and red lines are the upper and lower boundaries of the acceptance tolerance ($\pm$ 5°) of the target posture. The regression effect will be discussed based on the results in Section 4.5.4.

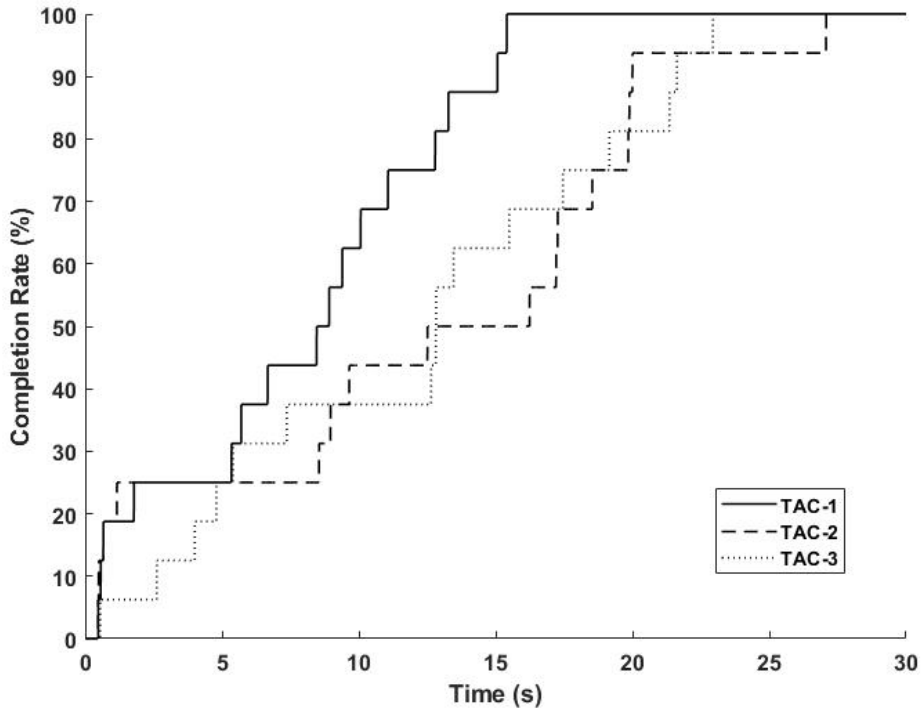
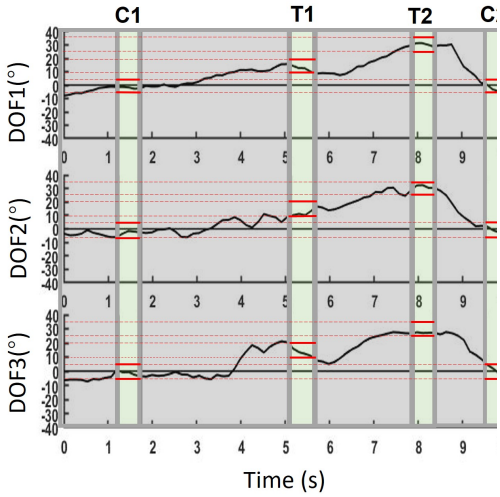
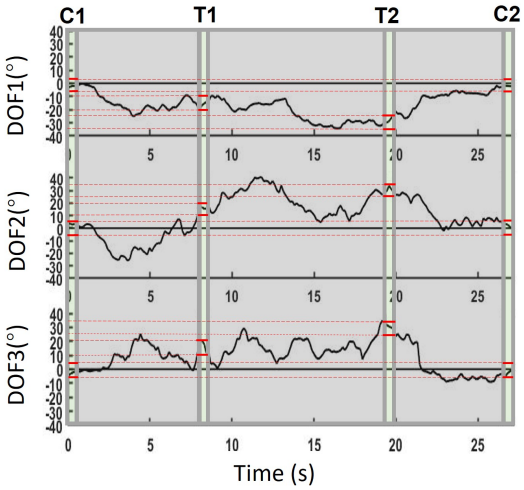


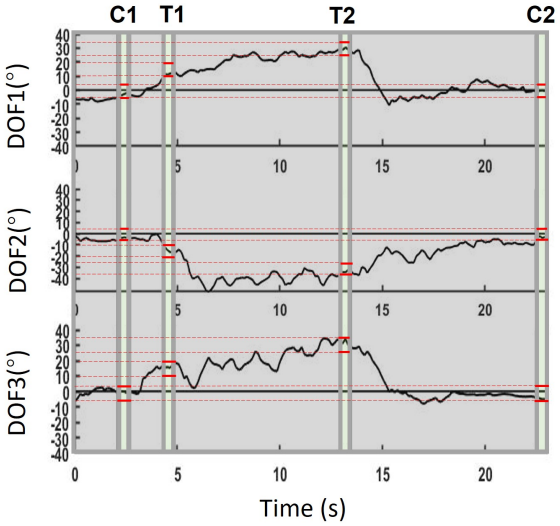
Figure 4.12: *Average completion rate curves for all TAC trials. Solid line indicates the TAC-1, dashed line indicates the TAC-2, and dotted line indicates the TAC-3.*



(a) TAC-1



(b) TAC-2



(c) TAC-3

Figure 4.13: *Example real-time regression trajectory curves in TAC test (Sub.4). C1 and C2 mean Central 1 and Central 2 posture, T1 and T2 mean Target 1 and Target 2, as shown in Tab 4.4 and Figure 4.7. The 0.5 s dwell time is represented as green, the grey areas denote the movement adjustment, the red lines are the upper and lower boundaries of the acceptance tolerance range. (a) Trajectory in TAC-1, the trial ended at 9.9 s; (b) Trajectory in TAC-2, the trial ended at 27.1 s; (c) Trajectory in TAC-3, the trial ended at 22.9 s.*

4.5 Discussions

In this chapter, a real-time control system was proposed to control a virtual hand. In Task 1, the regression accuracy was analyzed by Pearson CC between system joint angles output and the measured joint angles obtained from motion capture system. In Task 2, five minutes computational latency of the system obtained from each participant, and were analyzed by one-way ANOVA. In Task 3, participants controlled a virtual hand only with their sEMG signals, and completed the 13 daily motions (M1—M13) as real-time demonstration. Moreover, to prove the real-time multi-DOFs regression, three trials of TAC test were designed to emphasize the motion in 3-DOFs simultaneously. In this research, the high-accuracy and acceptable computational latency of the proposed real-time control system are the main contributions.

With a virtual hand as control object, we can focus on the evaluation of the real-time control system. If a real prosthetic hand was used for control, the real-time performance will also relate to the hardware performance (data transmission speed, delay, *etc.*), which should be another topic. Thus, in this chapter, only virtual hand was used to be controlled by the proposed system.

4.5.1 Real-Time Regression Accuracy

Figure 4.9(a) is the average CC results for different DOFs. Even though in Chapter 3 only single motions (M1—M5) was offline trained and estimated, in this chapter, we can still see the required performance accuracy and expected estimated curves (Figure 4.8) after adding the mixture motions (M6—M13) during model training.

Figure 4.9(b) is the average CC reuslts for different motion types: single motions (M1—M5), double-mixture motions (M6—M9), triple-mixture motions (M10—M13) and all motions. We can conclude that the estimation of mixture motions were accurate and nearly the same level compared to single motions. Thus, the proposed real-time control system has ability to not only regress

estimate the 3-DOFs joint angles simultaneously, but also achieve a required performance for mixture motions. This results gave me a lot of confidence to continue the regression check *via* 3-DOFs TAC test.

4.5.2 Effect of Adaptive Kalman Filter

From Figure 4.8, we can find the obvious oscillations from the red curves (estimated joint angles by CW-CNN regression model). The reason of the oscillations could be the low sampling rate of the estimated joint angles in real-time ($\approx$ 7.5 Hz), which may be caused by the computational latency and the process of sliding window. As shown in Section 4.4.2, the computational latency was stable, and unavoidable. However, the handling of sliding window always obtained the latest new sEMG signals from LSL pipeline, after they were processed to normalized IEMG signals, the sliding window still keep receiving the latest signals, which may generate a certain amount of signal loss during this processing procedure from the previous data processing to the next processing. I assumed that it should be main reason for the unexpected oscillations.

Although the oscillations occurred, we could recognize the motions from the estimated curves which meet our expectation. Therefore, only processing the output by CW-CNN to smoothed joint angles is enough, so that the outputs can be applied directly to control a prosthetic hand safely. In this study, virtual hand was control object, the smooth outputs also benefited for our real-time demonstration.

Adaptive Kalman Filter was considered to solve the problem of oscillations in this chapter. Before using the Adaptive one, the conventional Kalman Filter was tried to be used. In this situation, coefficient Q and R (Equation 4.1—4.3) needed to be adjusted manually. After many attempts, I found that $Q = 0.05$ and $R = 0.4$ should be suitable to obtain the filtered joint angles *via* Kalman Filter. However, it is hard to conclude that the preset coefficients manually

are the optimal solutions, thus, I improved the traditional Kalman Filter as Adaptive Kalman Filter, which can update all the coefficients automatically. To meet the expectation, the two coefficients Q and R should converge to optimal values as time goes (Figure 4.5), and the update method based on variance estimate [85] was considered within the iteration of Kalman Filter, as shown from Equation 4.4 to Equation 4.5. With the Adaptive Kalman Filter, we still need to assign initial values of Q and R, thus I set the initial values 0.05 and 0.4 to Q and R respectively, and for all participants, Q and R converged to near 0.002 and 0.02 respectively (according to Figure 4.5).

4.5.3 Computational Latency

In this study, I used PyQt5 to build the framework of the whole real-time control system, all components were updated within the *update()* function, and were looped regularly under *QtCore.QTimer()* module. I inserted *time.time()* in Python3 not only before each acquisition of the latest sEMG signals through sliding window to get starting point, but also after obtaining the system outputs *via* Adaptiva Kalman Filter to get ending point. The difference of time was calculated and converted to milliseconds (ms). Algorithm 1 shows the process of obtaining the computational latency.

From Figure 4.10, we can pay attention to the result for each participant first, *e.g.*, let us focus on Sub.1 as example. The p-value of Sub.1 for the five minutes groups is 0.9589 (> 0.5), which demonstrated that the computational latency was stable and average value was about 75 ± 45 ms during the Task 2 experiment. For Sub.2 to Sub.8, the p-values were higher than 0.5, thus I can conclude that the latency will not be pronounced as time goes, it can keep stable from start to end. The average computational latency was 74—76 ms according to the experimental results. Next, comparing the computational latency data of the eight different days, we can find the p-value $= 0.8410 > 0.5$, which demonstrated that the system was stable during the experiments even

on different days.

As we know, in the skeletal muscle contraction, a delay between electrical motion commands and force detection occurs, which ranges from 30—100 ms [92], and from the demonstration videos in Appendix C, in this study the computational latency of approximately 75 ms, within the electrical signals delay range as mentioned, which is acceptable, as we can visually feel from the videos. Moreover, the sliding window size was 120 ms, with the 75 ms computational latency still within the 300 ms real-time control constraint [47].

However, investigating from Figure 4.10, by focusing on the box plots of each participant, we can find the latency still floating within a certain range even though the delays were stable (standard deviation was about ±44 ms). In my opinion, this floating was generated by the uncontrollable latency caused by the process of using LSL to receive EMG signals and send experimental data to both the virtual hand and the computer which was used to receive the experimental data through LSL pipeline (Figure 4.3). Or maybe it was caused by the hardware of the PC being used for the experiment.

To investigate the delay ratio caused by each component, the process for computational latency was divided into five parts (according to Algorithm 1):

- Delay 1: sEMG signals acquisition
- Delay 2: pre-processed sEMG signals to IEMG signals
- Delay 3: normalize IEMG signals
- Delay 4: joint angles estimation using CW-CNN
- Delay 5: Adaptive Kalman Filter

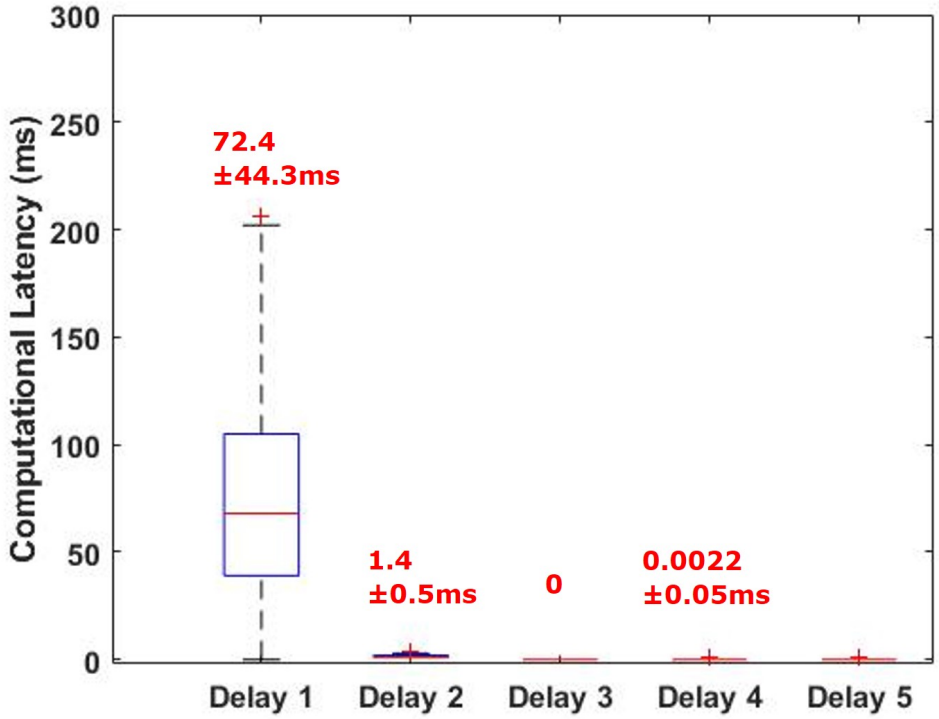


Figure 4.14: *Delay Ratio from computational latency.*

Figure 4.14 shows the experimental result. We can conclude that the process of sEMG acquisition (Delay 1) consumed the most computing time, so in the future, we can consider to speed up this period.

Moreover, from LSL system, we can check the sampling frequency of collected dataset, so we will have ability to know the time interval between two data. In fact, this time interval is the total latency of the computational period, which including the computation latency generated by proposed real-time control system, and the delay from LSL system or PC hardware. The sampling frequency of data acquisition using LSL system was about $7.4779Hz$, so the total latency was about 134 ms. Figure 4.15 shows the ratio from total latency, among the result, calculation latency accounted for about 55%, while LSL delay was about 45%, basically half of each. To this reason, in the future, if we use better way to transmit data from real-time control system to virtual hand or prosthetic hand instead of LSL system, we will get less total latency of each calculation period.

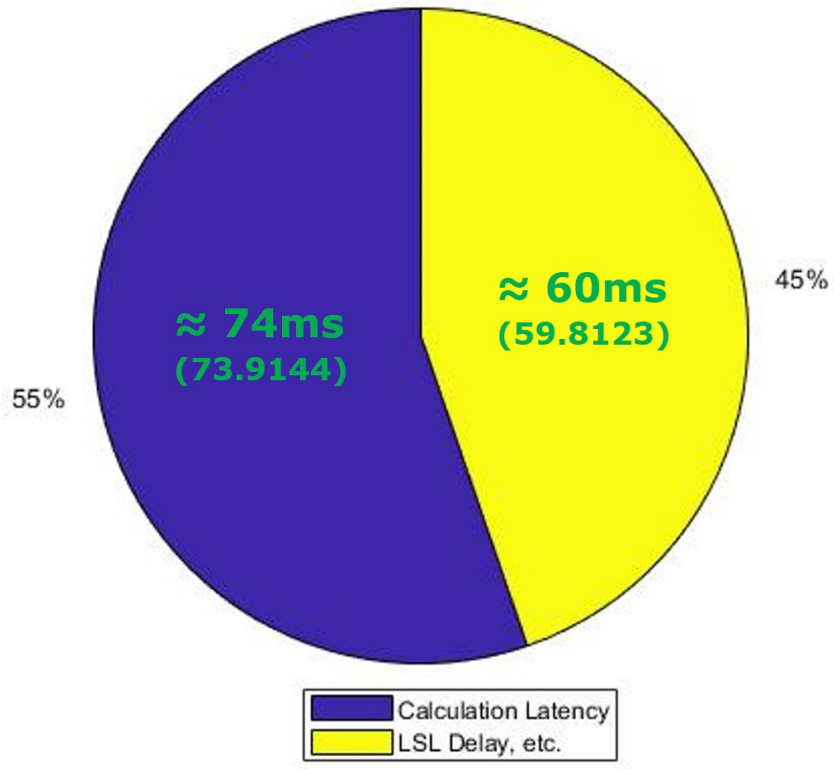


Figure 4.15: *Delay Ratio from total delay of each calculation period.*

4.5.4 TAC Test and Regression Trajectory Emphasis

To demonstrate that the users can control the virtual hand to any motions within the motions they can achieve, and to emphasize the simultaneous regression motions in 3-DOFs, TAC test was designed. During this experiment, the real-time regression performance can be proved, and the demonstration videos of TAC-1 to TAC-3 were provided, please refer to Appendix C.2.

Figure 4.12 demonstrates the participants completed the target postures of all trials within the time limitation. In Figure 4.13, the 0.5 s dwell time were displayed as green color based on the dwell information from completion rate results. To understand the meaning of Figure 4.13, we can take TAC-1 as an example. According to Table 4.4, 3-DOFs angles of the four target postures were [0°, 0°, 0°], [15°, 15°, 15°], [30°, 30°, 30°] and [0°, 0°, 0°], respectively. Namely, for each DOF, with the ±5° acceptance tolerance, the acceptable joint angles were [−5°, 5°], [10°, 20°], [25°, 35°] and [−5°, 5°], respectively. Similarly, readers can understand the experimental results of TAC-1, TAC-2 and TAC-3

for all participants. In Figure 4.13, red lines were used to indicate the upper and lower boundaries of the acceptance tolerance.

Based on the above discussion, from the trajectory curves, we can notice that participants tracked the target postures and 3-DOFs joint angles were estimated (regression prediction) in real-time environment. Compared to the results in Figure 4.8, the joint angles were kept staying for dwell time during TAC test at different intermediate angles, instead of changing from central position to maximum motions, which is a significant demonstration to investigate the difference between real-time regression and real-time classification. In addition, from the trajectory results, we can find a clear desirable regression changes in 3-DOFs simultaneously. However, we can still observe the angles changed within the acceptance tolerance during dwell time, I assumed that it is because of the slight muscle contraction/relaxation during motion adjustment and maintenance. Generally, when we want to keep a motion, we always control our muscle force to keep staying, and in my opinion, this is exactly the difficulty of real-time control approach compared to simple motion classification.

4.6 Conclusions

In the study of this chapter, a deep learning regression model-based (CW-CNN) real-time control system was proposed for virtual hand control. There are thirteen daily motions (M1—M13, including the single-DOF, double-DOFs and triple-DOFs) were used for model training and estimation. The regression accuracy (Pearson CC) and computational latency were evaluated, and a virtual hand model was used as control object in real-time. Eight healthy participants were invited to join the real-time experiments (***Exp.1***—***Exp.3***), and real-time demonstration videos were recorded (Task 3 of ***Exp.3***). Moreover, four of the participants (Sub.3, Sub.4, Sub.5 and Sub.8) conducted the TAC test.

The experimental results shown that:

- The proposed real-time control system performed as expectation not only in single motions, but also the mixture motions (multi-DOFs).
- The computational latency of the system were stable (p-value > 0.5) and acceptable (average 74—75 ms) for real-time control.
- Participants can complete the designed TAC tests using the proposed real-time control system, and the results demonstrated the motion regression in real-time processes.

According to the experimental results, the possibility of using the proposed real-time control system for future application on actual prosthetic hand was verified.

Chapter 5

Conclusion and Future Work

5.1 Summary

To improve the life quality of upper limb amputees, the control strategy of myoelectric prosthesis with high accuracy and robustness is necessary. Even until now, rare research proposed deep learning-based real-time control system, and most of them are related to motion classification. Thus, the research started from designing a deep learning-based regression model.

Considering that the control of a prosthetic hand is actually the control of motors by control commands, joint angles in different DOFs was used as the prediction target and designed the corresponding models and experiments. For model training, the sEMG signal and joint angles were acquired simultaneously as dataset by LSL, and the model accuracy was evaluated by correlation coefficient (CC). As the sEMG signal quality changes on different days, which can degrade the model performance, transfer learning was applied to update model. With the multi-array electrode sleeve, we do not need to adjust the electrode location for each subject, therefore, only FC layer parameters were updated during transfer learning. Experiments were designed in different days to prove the effect of transfer learning, and comparison between the proposed model and conventional regression models was discussed.

Next, a real-time control system using the model above was built. The sEMG signals were processed to IEMG signals in a real-time sliding window, and the joint angles were estimated by proposed regression model. Due to the lower sampling rate in real-time situation, oscillations could be observed in the output angles, which may damage the motors. Thus, Adaptive Kalman Filter was considered to smooth the outputs and used them as system outputs. The system outputs were sent to a virtual hand for real-time control tasks. To evaluate the system performance, CC was used again to analyze the real-time accuracy, and one-way ANOVA was used to prove the stable computational latency. To challenge the difficulty of real-time regression prediction compared to classification, three tasks of TAC test were designed to emphasize the 3-DOFs motions. Participants operated the virtual hand using their forearm sEMG signals to complete not only TAC test but also thirteen daily motions, which demonstrated the usability of the proposed real-time control strategies that it can be applied to prosthetic hand in the future.

In this dissertation, I am committed to achieve the goal of proposing a real-time control strategy for myoelectric prosthetic hand with high accuracy and natural performance. This research made the following contributions:

In Chapter 3

1. A CW-CNN regression model was proposed for daily motion estimation in 3-DOFs (WF/WE, WP/WS and HG/HO);

2. It was demonstrated that the model performed the best compared to another regression models;

3. A transfer learning strategy was proposed to update model on different days, and proved that the proposed layer transfer method can maintain and even improve the model robustness;

4. Based on the channel-wise convolutional layer, the motion patterns were explained *via* geometry plot by backtracking FC layer.

In Chapter 4

1. A real-time control system based on CW-CNN model for virtual hand control was proposed;

2. Experimental results show that the system obtained good performance for both single and mixture motions;

3. Our system has an acceptable (74—75 ms) and stable (p-value>0.5) computational latency;

4. Simultaneous 3-DOFs motion regression was proved in real-time by TAC test;

The above contributions demonstrated the usability of the real-time control strategies for virtual hand. In the future, the proposed approach can be applied to a practical myoelectric prosthetic hand.

5.2 Limitations and Further Improvements

5.2.1 Application of Practical Prosthetic Hand Control

In this thesis, participants controlled a virtual hand instead a prosthetic hand, and discussed only the computational latency for real-time motion recognition. If we use a real prosthetic hand, we also need to discussed the hardware performance of the machine. Therefore, in the future, when we apply the approach to control the myoelectric prosthesis directly, in addition to the computational latency, we also need to analyze the delay caused by the command transmission as well as the performance of the prosthesis hardware.

5.2.2 Introduce Force and Tactile Feedback Mechanism

In this thesis, the proposal has ability to estimate 3-DOFs joint angles in both offline and real-time environment. However, to control a robotic hand as

human, only joint angles estimation is not sufficient, we also need to predict the force, torque, stiffness, *etc.*. In the future, when we control an actual prosthetic hand, it will be interesting to include experiments on grasping objects based on high-precision estimation of multi-joint angles. Therefore, grip force regression [93, 94] should also be considered. For enhancing the ability of amputees to interact with objects, tactile feedback is important [95], integrated biomimetic tactile sensors [96–98] and a feedback actuation mechanism [99] should be designed in the future.

5.2.3 Test on actual amputees

In this thesis, the real-time control system was only tested on healthy participants. However, it is still hard to demonstrate if the similar performance level can be implemented on actual amputees or not. As we know, the real-time performance of amputees mainly depends on their muscle residuals level. During the experiment, it is regret to invite actual amputees to participate in the real-time control experiments. In the future, the tasks to apply the work in this thesis to actual amputees should be mentioned.

Bibliography

[1] National Academies of Sciences, Engineering, and Medicine, *The Promise of Assistive Technology to Enhance Activity and Work Participation*. Washington, DC: The National Academies Press, 2017.

[2] W. Frontera, J. Silver, and R. T., *Essentials of Physical Medicine and Rehabilitation (4 th ed.): Upper Limb Amputations*. Elsevier, 2019.

[3] A. Kato, Y. Matsumoto, Y. Kobayashi, M. G. Fujie, and S. Sugano, "Joint angle estimation using the distribution of the muscle bulge on the forearm skin surface of an upper limb amputee," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016, pp. 001 490–001 495.

[4] J. Davidson, "A survey of the satisfaction of upper limb amputees with their prostheses, their lifestyles, and their abilities," *Journal of Hand Therapy*, vol. 15, no. 1, pp. 62–70, 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0894113002500119

[5] S. Watve, G. Dodd, R. MacDonald, and E. R. Stoppard, "Upper limb prosthetic rehabilitation," *Orthopaedics and Trauma*, vol. 25, no. 2, pp. 135–142, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877132710001338

[6] M. Carrozza, G. Cappiello, G. Stellin, F. Zaccone, F. Vecchi, S. Micera, and P. Dario, "A cosmetic prosthetic hand with tendon driven underactuated mechanism and compliant joints: Ongoing research and preliminary results," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 2661–2666.

[7] A. Henson, "An introduction to cosmetic prostheses," accessed 2021. [Online]. Available: https://www.armdynamics.com/upper-limb-library/an-introduction-to-cosmetic-prostheses

[8] S. Carey, D. Lura, and M. Highsmith, "Differences in myoelectric and body-powered upper-limb prostheses: Systematic literature review," *Journal of Rehabilitation Research and Development*, vol. 52, pp. 247–262, 08 2015.

[9] Ottobock, "Body-powered prosthetic solutions," accessed 2021. [Online]. Available: https://www.ottobockus.com/prosthetics/upper-limb-prosthetics/solution-overview/body-powered-prosthetic-solutions/

[10] K. Bergman, L. Örnholmer, K. Zackrisson, and M. Thyberg, "Functional benefit of an adaptive myoelectric prosthetic hand compared to a conventional myoelectric hand," *Prosthetics and Orthotics International*, vol. 16, no. 1, pp. 32–37, 1992, pMID: 1584641. [Online]. Available: https://doi.org/10.3109/03093649209164305

[11] C. Prahm, I. Vujaklija, F. Kayali, P. Purgathofer, and O. C. Aszmann, "Game-based rehabilitation for myoelectric prosthesis control," *JMIR Serious Games*, vol. 5, no. 1, p. e3, Feb 2017. [Online]. Available: https://games.jmir.org/2017/1/e3/

[12] S. Schulz, "First experiences with the vincent hand," in *2011 MyoElectric Controls/Powered Prosthetics Symposium Fredericton*, Aug 2011, pp. 1–4.

[13] Van Der Niet Otr, O. *et al.*, "The i-limb hand and the dmc plus hand compared: A case report," *Prosthetics and Orthotics International*, vol. 34, no. 2, pp. 216–220, 2010, pMID: 20470060. [Online]. Available: https://doi.org/10.3109/03093641003767207

[14] J. T. Belter, J. L. Segil, A. M. Dollar, and R. F. Weir, "Mechanical design and performance specifications of anthropomorphic prosthetic hands: a review." *Journal of rehabilitation research and development*, vol. 50 5, pp. 599–618, 2013.

[15] Ottobock, "Michelongelo hand," accessed 2019. [Online]. Available: https://www.ottobock.com/en-ca/product/8E500

[16] S. B. Godfrey, K. D. Zhao, A. Theuer, M. G. Catalano, M. Bianchi, R. Breighner, D. Bhaskaran, R. Lennon, G. Grioli, M. Santello, A. Bicchi, and K. Andrews, "The softhand pro: Functional evaluation of a novel, flexible, and robust myoelectric prosthesis," *PLOS ONE*, vol. 13, no. 10, pp. 1–20, 10 2018. [Online]. Available: https://doi.org/10.1371/journal.pone.0205653

[17] A. Bayrak and E. Bekiroglu, "Bionic hand: A brief review," *Journal of Bionic Memory*, vol. 2, no. 1, p. 37–43, Mar. 2022. [Online]. Available: http://jbionicmemory.com/index.php/jbm/article/view/25

[18] M. P. Selvan, R. Raj, R. G. Sai, S. Jancy, and V. A. Mary, "Prosthetic hand using emg," in *J. Phys.: Conf. Ser.*, 2021, pp. 4–6.

[19] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. M. Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Sci Data*, vol. 1, no. 140053, 2014. [Online]. Available: https://doi.org/10.1038/sdata.2014.53

[20] W. Yang, D. Yang, Y. Liu, and H. Liu, "Decoding simultaneous multi-dof wrist movements from raw emg signals using a convolutional neural network," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 5, pp. 411–420, 2019.

[21] M. Yoshikawa, M. Mikawa, and K. Tanaka, "Real-time hand motion estimation using emg signals with support vector machines," in *2006 SICE-ICASE International Joint Conference*, 2006, pp. 593–598.

[22] S. P. Arjunan, D. K. Kumar, and G. R. Naik, "A machine learning based method for classification of fractal features of forearm semg using twin support vector machines," in *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, 2010, pp. 4821–4824.

[23] X. Chen and Z. J. Wang, "Pattern recognition of number gestures based on a wireless surface emg system," *Biomedical Signal Processing and Control*, vol. 8, no. 2, pp. 184–192, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1746809412000870

[24] A. Phinyomark, P. Phukpattaranont, and C. Limsakul, "Feature reduction and selection for emg signal classification," *Expert Systems with Applications*, vol. 39, no. 8, pp. 7420–7431, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417412001200

[25] A. J. Young, L. H. Smith, E. J. Rouse, and L. J. Hargrove, "Classification of simultaneous movements using surface emg pattern recognition," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 5, pp. 1250–1258, 2013.

[26] S. Shin, R. Langari, and R. Tafreshi, "A Performance Comparison of EMG Classification Methods for Hand and Finger Motion," ser. Dynamic Systems and Control Conference, vol. Volume 2: Dynamic Modeling and Diagnostics in Biomedical Systems; Dynamics and Control of Wind Energy Systems; Vehicle Energy Management Optimization; Energy Storage, Optimization; Transportation and Grid Applications; Estimation and Identification Methods, Tracking, Detection, Alternative Propulsion Systems; Ground and Space Vehicle Dynamics; Intelligent Transportation Systems and Control; Energy Harvesting; Modeling and Control for Thermo-Fluid Applications, IC Engines, Manufacturing, 10 2014, v002T16A008. [Online]. Available: https://doi.org/10.1115/DSCC2014-5993

[27] W. Jiang and Z. Yin, "Human activity recognition using wearable sensors by deep convolutional neural networks," in *Proceedings of the 23rd ACM International Conference on Multimedia*, ser. MM '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 1307–1310. [Online]. Available: https://doi.org/10.1145/2733373.2806333

[28] M. Zia ur Rehman, A. Waris, S. O. Gilani, M. Jochumsen, I. K. Niazi, M. Jamil, D. Farina, and E. N. Kamavuako, "Multiday emg-based classification of hand motions with deep learning techniques," *Sensors*, vol. 18, no. 8, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/8/2497

[29] T. Triwiyanto, I. P. A. Pawana, and M. H. Purnomo, "An improved performance of deep learning based on convolution neural network to classify the hand motion by evaluating hyper parameter," *IEEE Transactions on*

Neural Systems and Rehabilitation Engineering, vol. 28, no. 7, pp. 1678–1688, 2020.

[30] D. Yang and H. Liu, "An emg-based deep learning approach for multi-dof wrist movement decoding," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 7, pp. 7099–7108, 2022.

[31] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *INTERSPEECH*, 2014, pp. 338–342.

[32] X. Li and X. Wu, "Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4520–4524.

[33] M. Atzori, M. Cognolato, and H. Müller, "Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands," *Frontiers in Neurorobotics*, vol. 10, 2016. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnbot.2016.00009

[34] P. Wang, Q. Song, H. Han, and J. Cheng, "Sequentially supervised long short-term memory for gesture recognition," *Cognitive Computation*, vol. 8, 10 2016.

[35] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, "A novel attention-based hybrid cnn-rnn architecture for semg-based gesture recognition," *PLOS ONE*, vol. 13, no. 10, pp. 1–18, 10 2018. [Online]. Available: https://doi.org/10.1371/journal.pone.0206049

[36] P. Xia, J. Hu, and Y. Peng, "Emg-based estimation of limb movement using deep learning with recurrent convolutional neural networks," *Artificial Organs*, vol. 42, no. 5, pp. E67–E77, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/aor.13004

[37] D. Huang and B. Chen, "Surface emg decoding for hand gestures based on spectrogram and cnn-lstm," in *2019 2nd China Symposium on Cognitive Computing and Hybrid Intelligence (CCHI)*, 2019, pp. 123–126.

[38] D. Bai, T. Liu, X. Han, G. Chen, Y. Jiang, and Y. Hiroshi, "Multi-channel semg signal gesture recognition based on improved cnn-lstm hybrid models," in *2021 IEEE International Conference on Intelligence and Safety for Robotics (ISR)*, 2021, pp. 111–116.

[39] R. V. Godoy, G. J. G. Lahr, A. Dwivedi, T. J. S. Reis, P. H. Polegato, M. Becker, G. A. P. Caurin, and M. Liarokapis, "Electromyography-based, robust hand motion classification employing temporal multi-channel vision transformers," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 200–10 207, 2022.

[40] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/375c71349b295fbe2dcdca9206f20a06-Paper.pdf

[41] U. Côté-Allard, C. L. Fall, A. Campeau-Lecours, C. Gosselin, F. Laviolette, and B. Gosselin, "Transfer learning for semg hand gestures recognition using convolutional neural networks," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017, pp. 1663–1668.

[42] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "A deep transfer learning approach to reducing the effect of electrode shift in emg pattern recognition-based control," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, no. 2, pp. 370–379, 2020.

[43] T. Bao, Y. Zhao, S. A. R. Zaidi, S. Xie, P. Yang, and Z. Zhang, "A deep kalman filter network for hand kinematics estimation using semg," *Pattern Recognition Letters*, vol. 143, pp. 88–94, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865521000118

[44] A. Furui, S. Eto, K. Nakagaki, K. Shimada, G. Nakamura, A. Masuda, T. Chin, and T. Tsuji, "A myoelectric prosthetic hand with muscle synergy-based motion determination and impedance model-based biomimetic control," *Science Robotics*, vol. 4, no. 31, p. eaaw6339, 2019. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.aaw6339

[45] J. O. de Oliveira de Souza, M. D. Bloedow, F. C. Rubo, R. M. de Figueiredo, G. Pessin, and S. J. Rigo, "Investigation of different approaches to real-time control of prosthetic hands with electromyography signals," *IEEE Sensors Journal*, vol. 21, no. 18, pp. 20 674–20 684, 2021.

[46] M. Jafarzadeh, D. C. Hussey, and Y. Tadesse, "Deep learning approach to control of prosthetic hands with electromyography signals," in *2019 IEEE International Symposium on Measurement and Control in Robotics (ISMCR)*, 2019, pp. A1–4–1–A1–4–11.

[47] S. Tam, M. Boukadoum, A. Campeau-Lecours, and B. Gosselin, "A fully embedded adaptive real-time hand gesture classifier leveraging hd-semg and deep learning," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 2, pp. 232–243, 2020.

[48] S. Chen and J. Lu, "Design of a wireless electromyographic signal acquisition device," *Microcomputer Information*, vol. 14, no. 113, pp. 105–106, 2008.

[49] M. Zheng, M. S. Crouch, and M. S. Eggleston, "Surface electromyography as a natural human–machine interface: A review," *IEEE Sensors Journal*, vol. 22, no. 10, pp. 9198–9214, 2022.

[50] U. Kuruganti, "21 - multichannel surface electromyography," in *Bioelectronics and Medical Devices*, ser. Woodhead Publishing Series in Electronic and Optical Materials, K. Pal, H.-B. Kraatz, A. Khasnobish, S. Bag, I. Banerjee, and U. Kuruganti, Eds. Woodhead Publishing, 2019, pp. 513–535. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780081024201000273

[51] J. Wu, X. Li, W. Liu, and Z. J. Wang, "sEMG signal processing methods: A review," *Journal of Physics: Conference Series*, vol. 1237, no. 3, p. 032008, jun 2019. [Online]. Available: https://doi.org/10.1088/1742-6596/1237/3/032008

[52] E. J. Scheme and K. B. Englehart, "Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use." *Journal of rehabilitation research and development*, vol. 48 6, pp. 643–59, 2011.

[53] D. Farina, N. Jiang, H. Rehbaum, A. Holobar, B. Graimann, H. Dietl, and O. C. Aszmann, "The extraction of neural information from the surface emg for the control of upper-limb prostheses: Emerging avenues and challenges," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 797–809, 2014.

[54] Y. Koike, Y. Kim, S. Staponchaisit, Z. Qin, T. Kawase, and N. Yoshimura, "Development of multi-sensor array electrodes for measurement of deeper muscle activation," *Sensors and Materials*, vol. 32, 2020. [Online]. Available: https://sensors.myu-group.co.jp/article.php?ss=2636

[55] L. Myers, M. Lowery, M. O'Malley, C. Vaughan, C. Heneghan, A. St Clair Gibson, Y. Harley, and R. Sreenivasan, "Rectification and non-linear pre-processing of emg signals for cortico-muscular analysis," *Journal of Neuroscience Methods*, vol. 124, no. 2, pp. 157–165, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165027003000049

[56] A. Mannard and R. B. Stein, "Determination of the frequency response of isometric soleus muscle in the cat using random nerve stimulation," *The Journal of Physiology*, vol. 229, no. 2, pp. 275–296, 1973. [Online]. Available: https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/jphysiol.1973.sp010138

[57] J. G. Proakis and D. K. Manolakis, *Digital Signal Processing (4th Edition)*, 4th ed. Prentice Hall, 2006. [Online]. Available: http://www.amazon.com/Digital-Signal-Processing-John-Proakis/dp/0131873741%3FSubscriptionId%3D192BW6DQ43CK9FN0ZGG2%26tag%3Dws%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D0131873741

[58] Y. Koike and M. Kawato, "Estimation of dynamic joint torques and trajectory formation from surface electromyo graphy signals using a neural network model," *Biological Cybernetics*, vol. 73, pp. 291–300, 1995. [Online]. Available: https://doi.org/10.1007/BF00199465

[59] R. A. R. C. GOPURA and K. KIGUCHI, "An exoskeleton robot for human forearm and wrist motion assist," *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, vol. 2, no. 6, pp. 1067–1083, 2008.

[60] G. ElKoura and K. Singh, "Handrix: Animating the human hand," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '03. Goslar, DEU: Eurographics Association, 2003, p. 110–119.

[61] N. Zengeler, T. Kopinski, and U. Handmann, "Hand gesture recognition in automotive human–machine interaction using depth cameras," *Sensors*, vol. 19, no. 1, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/1/59

[62] H. S. Kim, N. Hong, M. Kim, S. G. Yoon, H. W. Yu, H.-J. Kong, S.-J. Kim, Y. J. Chai, H. J. Choi, J. Y. Choi, K. E. Lee, S. Kim, and H. C. Kim, "Application of a perception neuron® system in simulation-based surgical training," *Journal of Clinical Medicine*, vol. 8, no. 1, 2019. [Online]. Available: https://www.mdpi.com/2077-0383/8/1/124

[63] Y. He and Y. Tang, "Neuron data reader runtime api documentation," 2014, accessed May 6, 2021. [Online]. Available: https://docplayer.net/42937137-Neuron-data-reader-runtime-api-documentation.html

[64] J. Chen, J. Qiu, and C. Ahn, "Construction worker's awkward posture recognition through supervised motion tensor decomposition," *Automation in Construction*, vol. 77, pp. 67–81, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580517300651

[65] Aiuto Co., Ltd., accessed March 12, 2021. [Online]. Available: https://aiuto-jp.co.jp/support/file_93.php

[66] Michael A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com/about.html

[67] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. De Mori, "Adaptation of hybrid ann/hmm models using linear hidden transformations and conservative training," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 1, 2006, pp. I–I.

[68] D. Dong, H. Wu, W. He, D. Yu, and H. Wang, "Multi-task learning for multiple language translation," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 1723–1732. [Online]. Available: https://aclanthology.org/P15-1166

[69] C. Boulay, "labstreaminglayer (github)," accessed June 14, 2022. [Online]. Available: https://github.com/sccn/labstreaminglayer

[70] Q. V. Le, "A tutorial on deep learning part 2: autoencoders, convolutional neural networks, and recurrent neural networks," *Google Brain 2015*, pp. 1–20, 2015. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.5244&rep=rep1&type=pdf

[71] S. Sakhavi, C. Guan, and S. Yan, "Learning temporal information for brain-computer interface using convolutional neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5619–5629, 2018.

[72] S. Stapornchaisit, Y. Kim, A. Takagi, N. Yoshimura, and Y. Koike, "Finger angle estimation from array emg system using linear regression model with independent component analysis," *Frontiers in Neurorobotics*, vol. 13, 2019. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnbot.2019.00075

[73] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 36, no. 2, pp. 111–133, 1974. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1974.tb00994.x

[74] J. D. Rodriguez, A. Perez, and J. A. Lozano, "Sensitivity analysis of k-fold cross validation in prediction error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 569–575, 2010.

[75] R. Taylor, "Interpretation of the correlation coefficient: A basic review," *Journal of Diagnostic Medical Sonography*, vol. 6, no. 1, pp. 35–39, 1990. [Online]. Available: https://doi.org/10.1177/875647939000600106

[76] N. Diamantidis, D. Karlis, and E. Giakoumakis, "Unsupervised stratification of cross-validation for accuracy estimation," *Artificial Intelligence*, vol. 116, no. 1, pp. 1–16, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370299000946

[77] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 57, no. 1, pp. 289–300, 1995. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1995.tb02031.x

[78] X. An and Y. Wang, "Smart wearable medical devices for isometric contraction of muscles and joint tracking with gyro sensors for elderly people," *J Ambient Intell Human Comput*, 2021. [Online]. Available: https://doi.org/10.1007/s12652-021-02993-5

[79] Kenhub (2021), "Elbow and forearm," accessed January 28, 2021. [Online]. Available: https://www.kenhub.com/en/library/anatomy/elbow-and-forearm

[80] L. J. Hargrove, G. Li, K. B. Englehart, and B. S. Hudgins, "Principal components analysis preprocessing for improved classification accuracies

in pattern-recognition-based myoelectric control," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 5, pp. 1407–1414, 2009.

[81] A. Simon, L. Hargrove, B. Lock, and T. Kuiken, "Target achievement control test: Evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses," *Journal of rehabilitation research and development*, vol. 48, pp. 619–27, 07 2011.

[82] J. Gusman, E. Mastinu, and M. Ortiz-CataláN, "Evaluation of computer-based target achievement tests for myoelectric control," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 5, pp. 1–10, 2017.

[83] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 03 1960. [Online]. Available: https://doi.org/10.1115/1.3662552

[84] Q. Li, R. Li, K. Ji, and W. Dai, "Kalman filter and its application," in *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, 2015, pp. 74–77.

[85] R. Mehra, "On the identification of variances and adaptive kalman filtering," *IEEE Transactions on Automatic Control*, vol. 15, no. 2, pp. 175–184, 1970.

[86] R. Taylor, "Interpretation of the correlation coefficient: A basic review," *Journal of Diagnostic Medical Sonography*, vol. 6, no. 1, pp. 35–39, 1990. [Online]. Available: https://doi.org/10.1177/875647939000600106

[87] S. Stapornchaisit, Y. Kim, A. Takagi, N. Yoshimura, and Y. Koike, "Finger angle estimation from array emg system using linear regression model with independent component analysis," *Frontiers in Neurorobotics*, vol. 13, 2019. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnbot.2019.00075

[88] Z. He, Z. Qin, and Y. Koike, "Continuous estimation of finger and wrist joint angles using a muscle synergy based musculoskeletal model," *Applied Sciences*, vol. 12, no. 8, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/8/3772

[89] H. Y. Kim, "Analysis of variance (anova) comparing means of more than two groups," *Restor Dent Endod*, vol. 39, no. 1, pp. 74–77, 2014. [Online]. Available: https://www.mdpi.com/2076-3417/12/8/3772

[90] D. M.Lane, "One-factor anova (between subjects)," accessed December 9, 2022. [Online]. Available: https://onlinestatbook.com/2/analysis_of_variance/one-way.html

[91] G. Ganesh, K. Nakamura, S. Saetia, A. M. Tobar, E. Yoshida, H. Ando, N. Yoshimura, and Y. Koike, "Utilizing sensory prediction errors for movement intention decoding: A new methodology," *Science Advances*, vol. 4, no. 5, p. eaaq0183, 2018. [Online]. Available: https://www.science.org/doi/abs/10.1126/sciadv.aaq0183

[92] P. Cavanagh and P. Komi, "Electromechanical delay in human skeletal muscle under concentric and eccentric contractions," *European journal of applied physiology and occupational physiology*, vol. 42, no. 3, p. 159—163, November 1979. [Online]. Available: https://doi.org/10.1007/bf00431022

[93] T. Baldacchino, W. R. Jacobs, S. R. Anderson, K. Worden, and J. Rowson, "Simultaneous force regression and movement classification of fingers via surface emg within a unified bayesian framework," *Frontiers in Bioengineering and Biotechnology*, vol. 6, 2018. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fbioe.2018.00013

[94] Y. Chen, C. Dai, and W. Chen, "Cross-comparison of emg-to-force methods for multi-dof finger force prediction using one-dof training," *IEEE Access*, vol. 8, pp. 13 958–13 968, 2020.

[95] W. T. Navaraj, H. Nassar, and R. Dahiya, "Prosthetic hand with biomimetic tactile sensing and force feedback," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019, pp. 1–4.

[96] Z. Liang, J. Cheng, Q. Zhao, X. Zhao, Z. Han, Y. Chen, Y. Ma, and X. Feng, "High-performance flexible tactile sensor enabling intelligent haptic perception for a soft prosthetic hand," *Advanced Materials Technologies*, vol. 4, no. 8, p. 1900317, 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/admt.201900317

[97] M. A. Abd, M. Al-Saidi, M. Lin, G. Liddle, K. Mondal, and E. D. Engeberg, "Surface feature recognition and grasped object slip prevention with a liquid metal tactile sensor for a prosthetic hand," in *2020 8th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)*, 2020, pp. 1174–1179.

[98] R. A. Romeo, C. Lauretti, C. Gentile, E. Guglielmelli, and L. Zollo, "Method for automatic slippage detection with tactile sensors embedded in prosthetic hands," *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 2, pp. 485–497, 2021.

[99] B. Stephens-Fripp, G. Alici, and R. Mutlu, "A review of non-invasive sensory feedback methods for transradial prosthetic hands," *IEEE Access*, vol. 6, pp. 6878–6899, 2018.

Appendix A

List of Main Abbreviations

ADL	Activities of Daily Living
DOF	Degree of freedom
CP	Central Position
WF/WE	Wrist Flexion/Wrist Extension
WP/WS	Wrist Pronation/Wrist Supination
HG/HO	Hand Grip/Hand Open
MUAP	motor unit action potentials
sEMG	Surface ElectroMyoGraphy
IEMG	Integrated ElectroMyoGraphy
EEG	ElectroEncephaloGraphy
GPU	Graphics Processing Unit
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
CW-CNN	Channel-Wise Convolutional Neural Network
FC	Fully Connected
LR	Linear Regression
SVM	Support Vector Machine
SVR	Support Vector Regression
KNN	K-Nearest Neighbor
DT	Decision Tree
CC	Correlation Coefficient
ADC	Analog-to-Digital Converter
LSL	Lab Streaming Layer
BVH	Bounding Volume Hierarchies
FIR	Finite Impulse Response
GUI	Graphical User Interface
ANOVA	Analysis of variance
TAC	Target Achievement Control

Appendix B

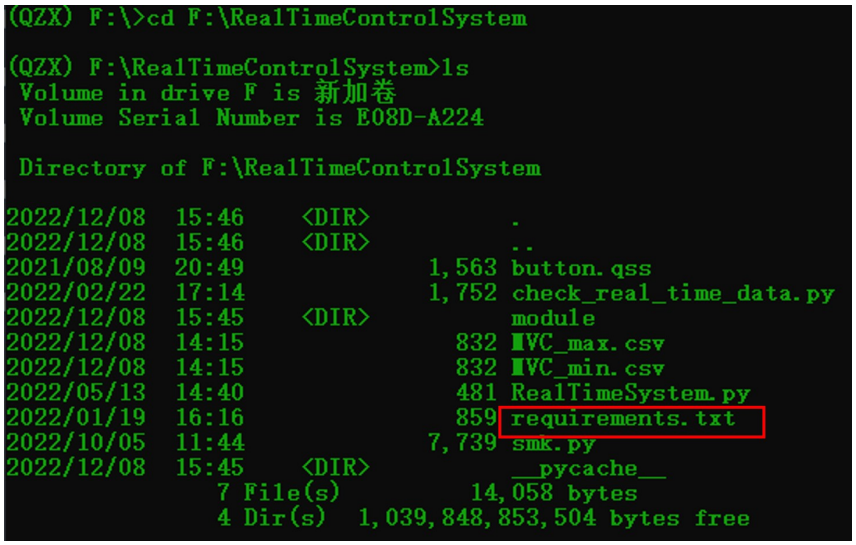
Usage of Real-Time Control GUI Tool

The link of demonstration video is provided in Appendix C.

B.1 Build Environment

In the study of this thesis, the real-time control system was built on Python 3.7.0 environment. Considering that versions of other packages are also affected by the Python version, thus, I suggest to use the version from **Python 3.6** to **3.9**. After we have prepared the Python environment, we can start building the environment for proposed real-time control system.

Next, prepare the source code file and copy the path. Then run the **Command Prompt** (in Windows OS) or **Terminal** (MacOS or Linux), ***cd*** to the corresponding path. You can use ***ls*** command to view all the files under the path. Your terminal will display the following screen:



There is a file named "requirements.txt" (red box), so you can use the following command to install all the packages the GUI tool needs, instead of installing them one by one:

pip3 install -r requirements.txt

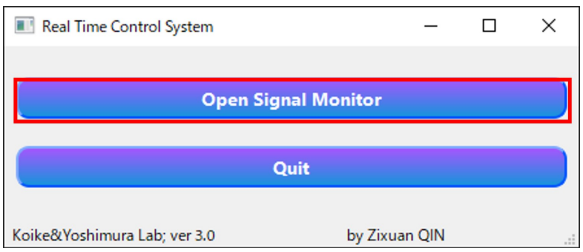
Once the installation is complete, we can start running the control system.

B.2 Run the GUI Tool and Make Connection

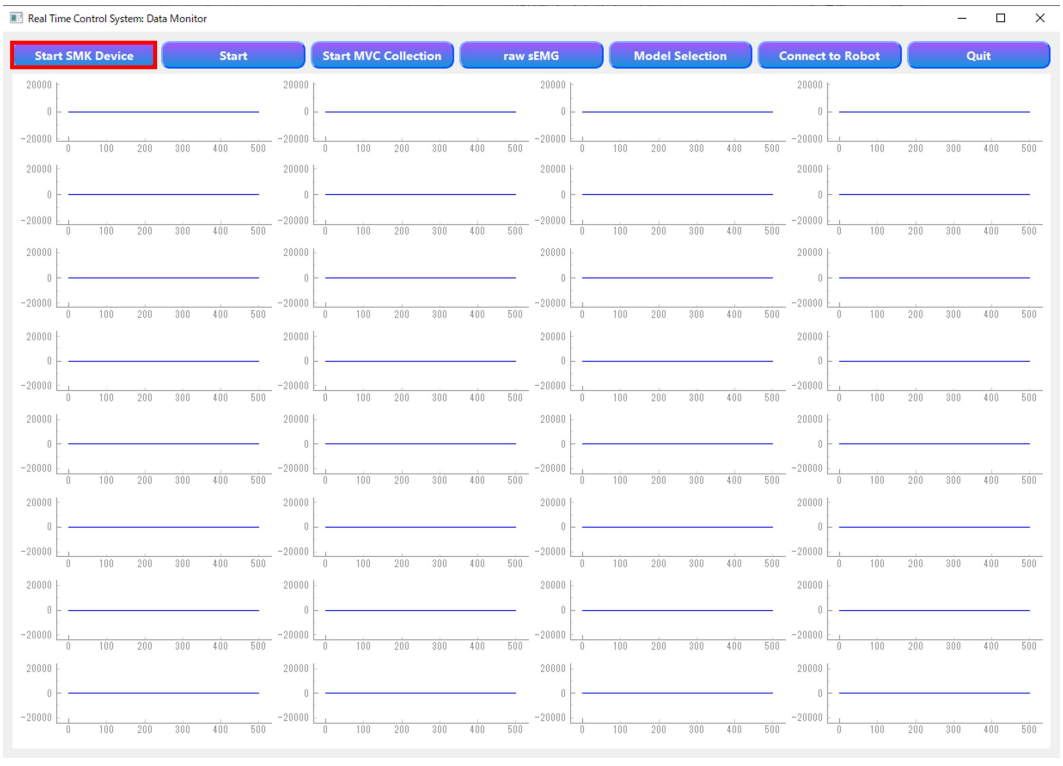
To run the GUI tool, you can use the command:

python3 RealTimeSystem.py

Then you will see the initial screen as Figure B.1(a). You can click "**Open Signal Monitor**" to start the GUI, as shown as Figure B.1(b), or "**Quit**" to kill the program. In the initial signal monitoring interface, only the button "**Start SMK Device**" and "**Quit**" enabled to be clicked. This button is to make connection between PC and multi-array electrode via USB Serial Port.



(a) Initial Selection Interface



(b) Initial Signal Monitoring Interface

Figure B.1: *Initial Interface after running.*

After selecting the "**Start SMK Device**" button, an additional Powershell (Windows OS) or Terminal (Linux and MacOS) appears, as shown in the top of Figure B.2. You can choose the serial port which connects with multi-array electrode, and type "**Enter**" to for connection, then you will see the data flow in real-time as bottom of Figure B.2. From the data flow, the 32 numbers within the square bracket denotes the sEMG amplitude of each channel, respectively. Then we can go back to the GUI tool, and click "**Start**" button to start the sEMG signal acquisition.

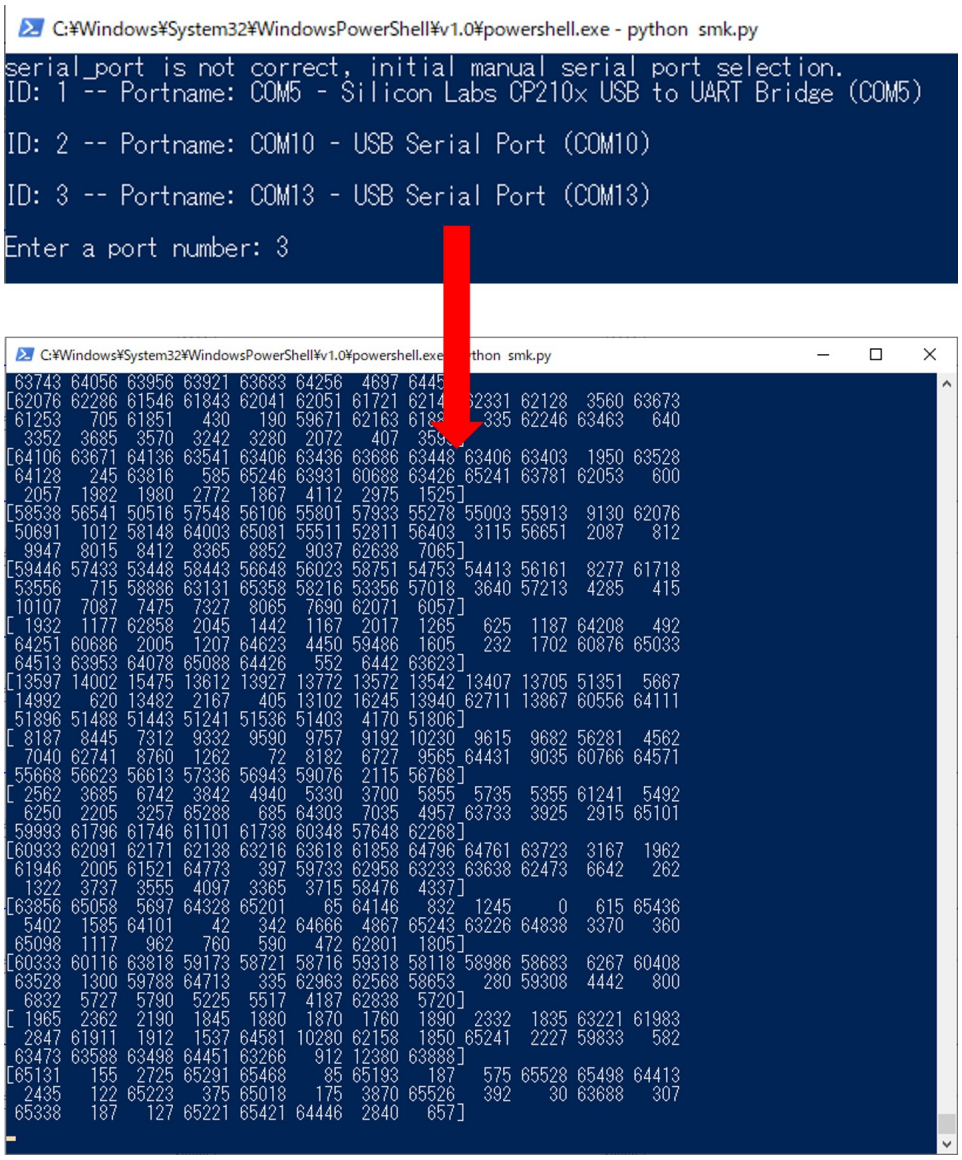


Figure B.2: *Make connection to multi-array electrode. Select the corresponding serial port in an additional PowerShell or Terminal, and enter to check if the signal data stream is successfully acquired.*

B.3 Data Acquisition and Monitoring

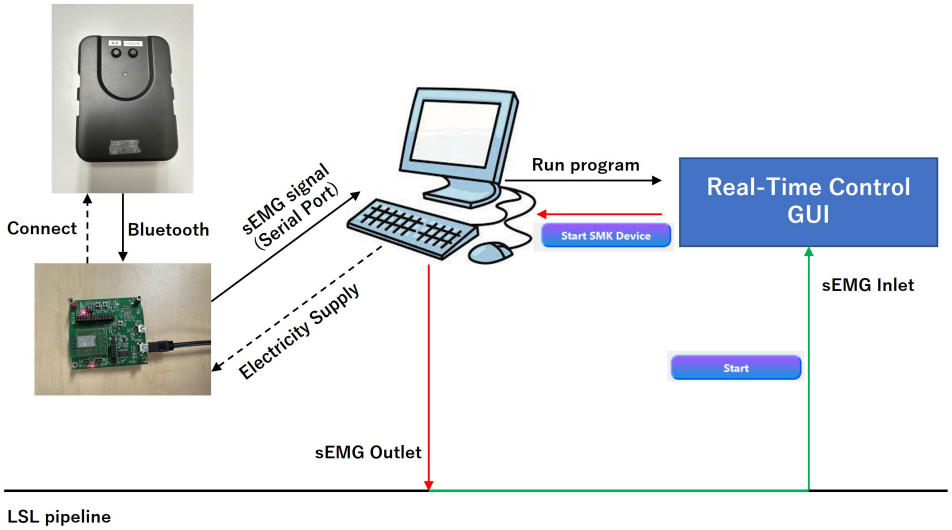


Figure B.3: *Illustration of sEMG signal acquisition using the proposed GUI tool.*

Figure B.3 explains the whole process of the signal acquisition and monitoring. The PC provides power to Bluetooth chip *via* USB cable, so that the chip can build connection with multi-array electrode device. As the red arrow, after clicking the "**Start SMK Device**" button, PC will make connection with multi-array electrode to receive sEMG signals, the data will be pushed to LSL pipeline *via* sEMG outlet, and we can also check the data from LabRecorder (Figure B.4. Not necessary, just as proof of data transmission). About the usage of LSL and LabRecorder, please refer to [69].

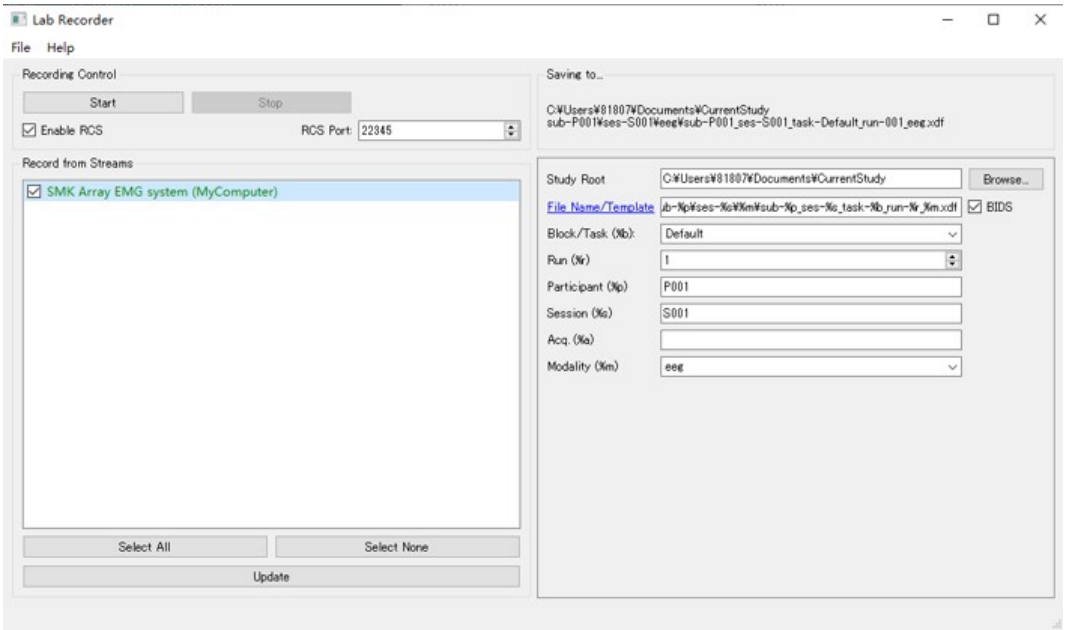


Figure B.4: *sEMG data flow can be recorded from LabRecorder.*

However until now, we can only check data flow from additional PowerShell or Terminal (Figure B.2), which are obtained from LSL pipeline. The data is not visualized because we do not interact them with the GUI tool. So as the green arrow in Figure B.3, next we need to click "**Start**" button to connect the GUI tool to LSL and receive sEMG signal stream from LSL pipeline by sEMG inlet. Thus, we can see the sEMG signals in real-time from the data monitoring window, as shown in Figure B.5.

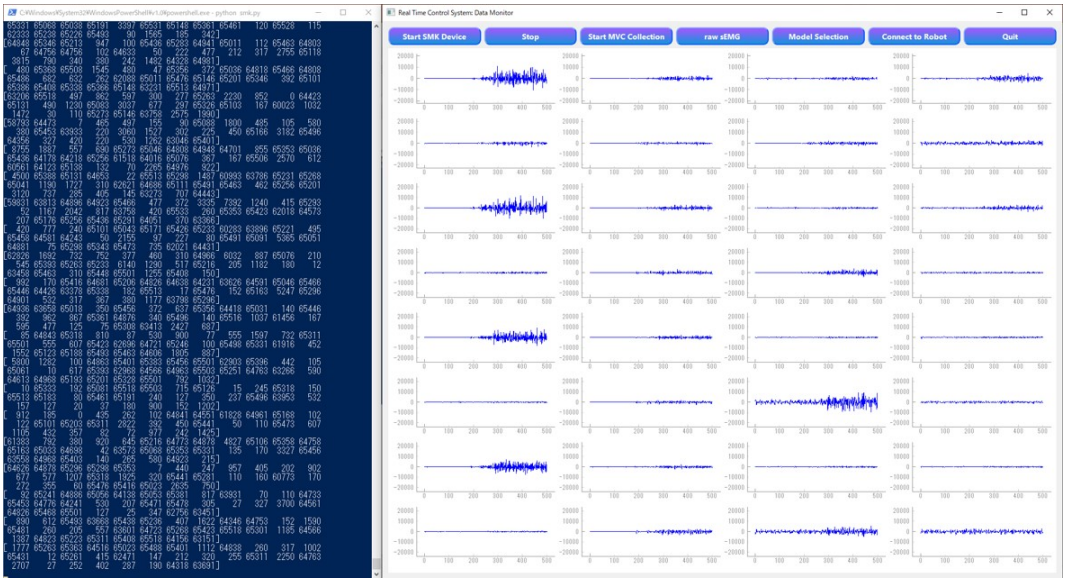


Figure B.5: *sEMG signal monitoring. The left part shows the real-time acquired data stream in the additional PowerShell corresponding to the observed sEMG signals.*

We can find that the button name changed to "**Stop**" during data monitoring, we can click the "**Stop**" button to stop receiving signal from LSL pipeline any time. Until now, only "**Start/Stop**" button, "**Quit**" button and "**Start MVC Collection**" button enable to be clicked. The logic is designed to guide the user to start MVC collection, and prevent other wrong operations.

B.4 MVC Collection and IEMG Monitoring

If we click "**Start MVC Collection**" button, the button will turn to "**Stop MVC Collection**" (Figure B.6), the GUI tool will start recording the sEMG signal until the user click "**Stop MVC Collection**" button. After finishing the MVC data collection, we can find two csv files (*MVC_min.csv* and *MVC_max.csv*) are saved in the current path.

Figure B.6: *Button switch for MVC collection.*

After the MVC collection, we can find that the button "**raw sEMG**" button enables us to click, the button name indicates the signal state. With the saved MVC data, sEMG signal can be processed to normalized IEMG signals in real-time based on Equation 2.8—2.10. If we click on "**raw sEMG**", the button turn to "**filtered sEMG**", indicates the state of monitored signals are IEMG signals. We can switch from raw sEMG signal and IEMG signals freely.

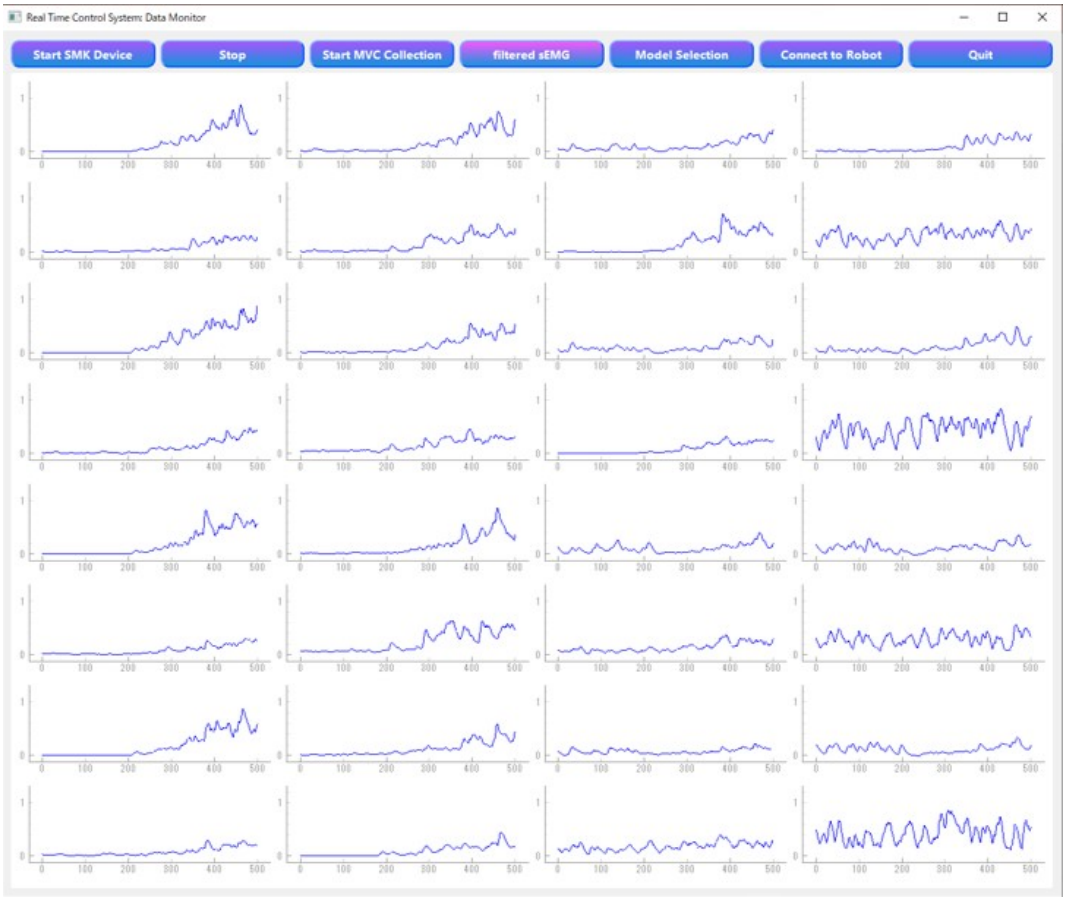


Figure B.7: *IEMG signal monitoring.*

B.5 Model Selection and Angles Monitoring

After 5-trial new data acquisition in another days, we can update the model by transfer learning. The updated models can be saved, then we can click "**Model Selection**" button to choose the model we want to use, as shown in Figure B.8.

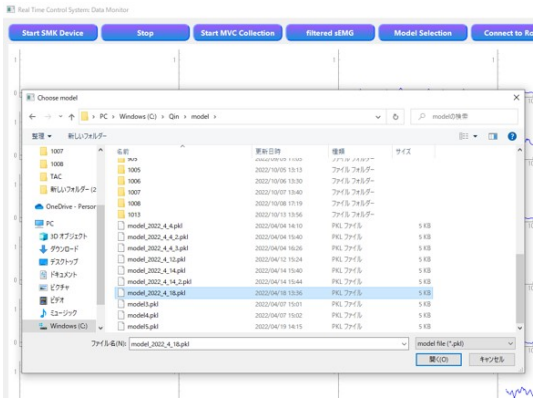


Figure B.8: *Model selection.*

After the model was loaded successfully, the button turn to "**Joint Angle Estimation**", we can click it to check the estimated angles in real-time, as Figure B.9. Now, the sEMG and corresponding joint angles can be monitored, blue curve is the predicted results by CW-CNN model, and the red curve is the result after processing by Adaptive Kalman Filter.

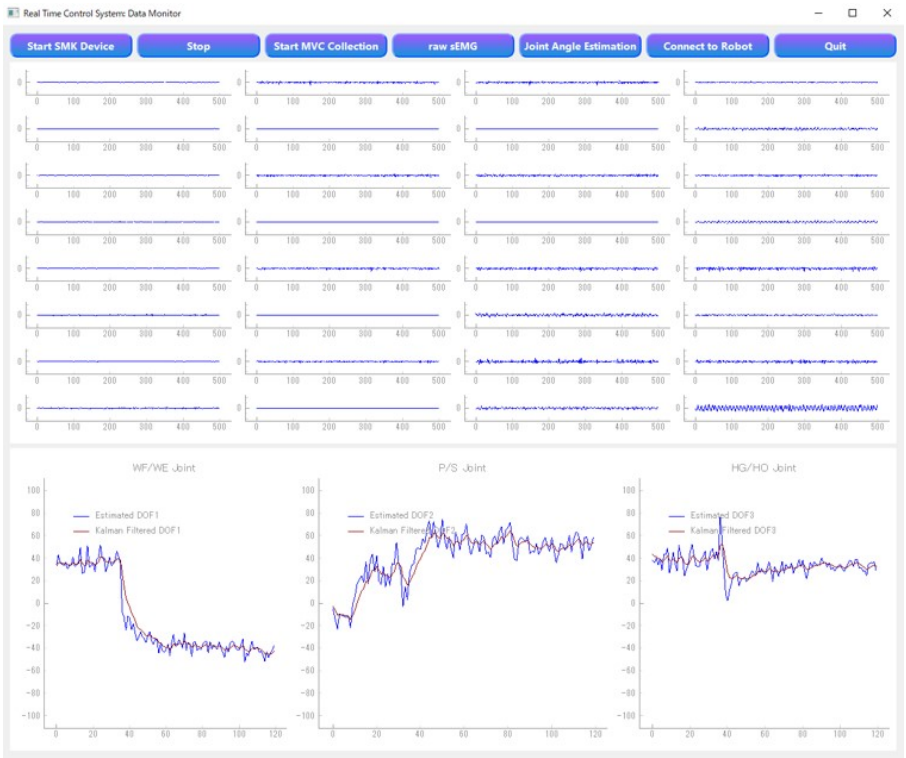


Figure B.9: *Joint angle estimation in real-time.*

B.6 Connect to Virtual Hand

After we click the "**Joint Angle Estimation**" button, the "**Connect to Robot**" button enables to be clicked. The button will turn to "**Data Collection**" after connecting to virtual hand, and it can switch between "**Data Collection**" and "**Stop**".

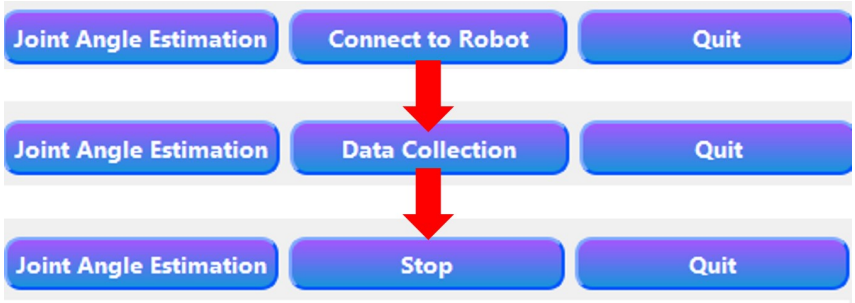


Figure B.10: *Button switch for sending control command and data collection.*

According to Figure 4.3, the "**Data Collection**" allows the GUI tool to push the data needed in the analysis of Task 1—3 to LSL pipeline, we can receive them using LabRecorder (Figure B.11). The "Stimulus PC Real Time" tag denotes the trigger data sent by Real-Time control system, to control the start and end of virtual hand programs.

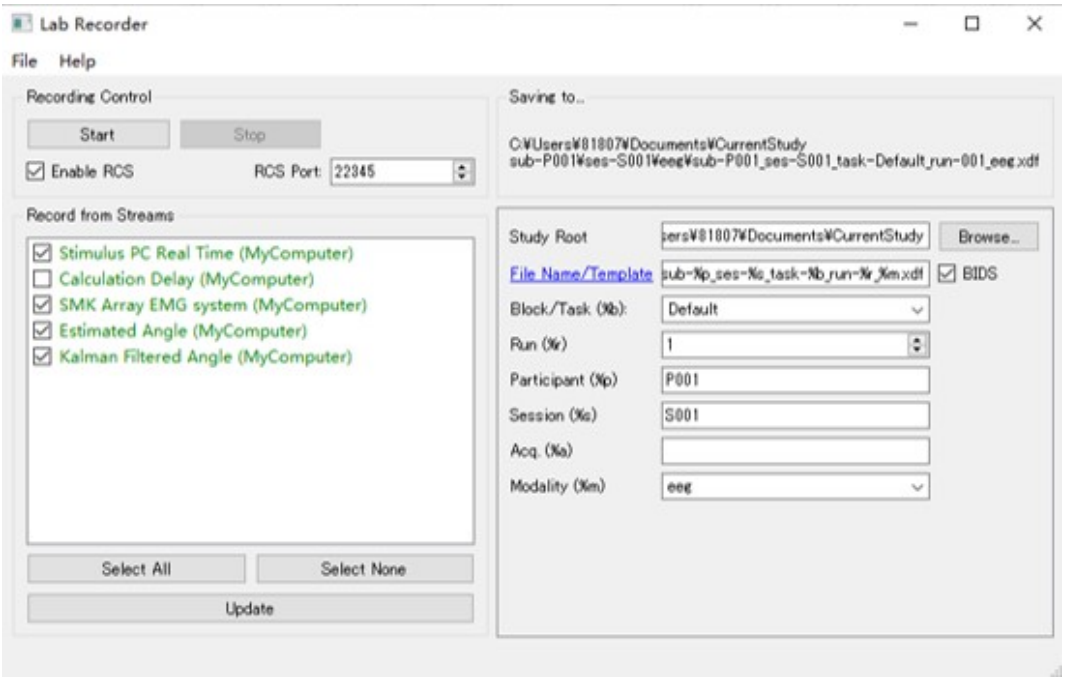


Figure B.11: *The experimental data (Task 1—3 in Chapter 4) can be recorded from LabRecorder.*

Appendix C

Real-Time Demonstration Videos

Please check or download from: https://www.frontiersin.org/articles/10.3389/fnbot.2022.1072365/full

- Video_1: Usage of proposed GUI tool

C.1 Real-time demonstration (M1—M13)

- Video_2: Right-handed control (in order)
- Video_3: Left-handed control (in order)
- Video_4: Left-handed control (random order)

C.2 TAC test demonstration

- Video_5: TAC-1
- Video_6: TAC-2
- Video_7: TAC-3