10-1-2009

Myoelectric control techniques for a rehabilitation robot

Alan Smith

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation


This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact rit.scholarworks@rit.edu.
Myoelectric Control Techniques for a Rehabilitation Robot

by

Alan Smith

A Thesis Submitted

in

Partial Fulfillment

of the

Requirements for the Degree of

MASTER OF SCIENCE

in

Electrical Engineering

Approved by:

PROF. ______________________________________________________________________
(Dr. Edward E. Brown Jr, Assistant Professor, Department of Electrical Engineering)

PROF ______________________________________________________________________
(Dr. Daniel B. Phillips, Associate Professor, Department of Electrical Engineering)

PROF ______________________________________________________________________
(Dr. Juan Carlos Cockburn, Associate Professor, Department of Computer Engineering)

PROF ______________________________________________________________________
(Dr. Sohail Dianat, Head of Department, Department of Electrical Engineering)

DEPARTMENT OF ELECTRICAL ENGINEERING

COLLEGE OF ENGINEERING

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NEW YORK

OCTOBER, 2009
Table of Contents

Abstract ......................................................................................................................................................... 1

Chapter 1: Introduction ................................................................................................................................. 3

Chapter 2: Background ................................................................................................................................. 5

  2.1 Anatomy and Physiology of Human Motor Control .............................................................................. 5
  2.2 Electromyography ............................................................................................................................... 7
  2.3 Muscles of Elbow and Forearm Movement .......................................................................................... 10
  2.4 Central Cord Syndrome ...................................................................................................................... 14
  2.5 Pattern Recognition Myoelectric Control Systems ........................................................................... 15
    2.5.1 General Overview ...................................................................................................................... 15
    2.5.2 EMG Data Acquisition .............................................................................................................. 17
    2.5.3 Feature Extraction ...................................................................................................................... 20
    2.5.4 Classifiers ................................................................................................................................. 33
  2.6 Labview Systems ............................................................................................................................... 36
  2.7 Model Based Myoelectric Control Systems ...................................................................................... 37

Chapter 3: Methodology ............................................................................................................................. 42

  3.1 Development of a Real Time Pattern Recognition Myoelectric Control Scheme ............................... 42
    3.1.1 Offline Window and Feature Analysis ...................................................................................... 42
    3.1.2 Real Time Pattern Recognition Myoelectric Control Scheme for Healthy Subjects ................. 46
    3.1.3 Real Time Pattern Recognition Myoelectric Control Scheme for a CCS Subject .................... 50
  3.2 Development of a Myoelectric Control Scheme Based on a Time Delayed Neural Network ......... 52

Chapter 4: Results ....................................................................................................................................... 58

  4.1 Real Time Pattern Recognition Myoelectric Control Scheme Results ............................................. 58
    4.1.1 Offline Window and Feature Analysis Results .......................................................................... 58
    4.1.2 Real Time Pattern Recognition Myoelectric Control Scheme for Healthy Subjects ............... 69
    4.1.3 Real Time Pattern Recognition Myoelectric Control Scheme for a CCS Subject ................... 79
  4.2 Myoelectric Control Scheme Based on a Time Delayed Neural Network Results ....................... 81

Chapter 5: Discussion ............................................................................................................................... 101

  5.1 Real Time Pattern Recognition Myoelectric Control Scheme Discussion .................................... 101
    5.1.1 Offline Window and Feature Analysis Discussion .................................................................... 101
    5.1.2 Real Time Pattern Recognition Myoelectric Control Scheme for Healthy Subjects Discussion 105
    5.1.3 Real Time Pattern Recognition Myoelectric Control Scheme for a CCS Subject Discussion .... 108
  5.2 Myoelectric Control Scheme Based on a Time Delayed Neural Network Discussion ............... 112

Chapter 6: Conclusion ............................................................................................................................... 116
List of Figures

Figure 1. Central Nervous System and Motor Control ................................................................. 5
Figure 2. Schematic of the Hodgkin-Huxley Model ................................................................. 7
Figure 3. Model of the Electrode to Skin Interface During EMG Recording .......................... 8
Figure 4. Differential Configuration for EMG Amplifier ....................................................... 9
Figure 5. Biceps Brachii ......................................................................................................... 10
Figure 6. Brachialis ............................................................................................................. 10
Figure 7. Triceps Brachii, Anconeus, and Other Upper Arm Muscles .................................. 11
Figure 8. Pronator Quadrates, Pronator Teres, Supinator, and Other Forearm Muscles .......... 12
Figure 9. Muscles of the Forearm and Hand ................................................................... 12
Figure 10. Forearm Motions of Pronation and Supination .................................................. 13
Figure 11. Block Diagram of the Myoelectric Control Scheme Using Pattern Recognition .... 16
Figure 12. Adjacent Windowing Technique ......................................................................... 18
Figure 13. Overlapped Windowing Technique ................................................................ 19
Figure 14. Majority Voting Scheme vs. Regular Adjacent Windowing Scheme .................... 20
Figure 15. Feature Space for a Two Class Pattern Recognition Problem Using AR Features .... 21
Figure 16. Time-Series Model Proposed by Graupe for EMG Recordings ......................... 24
Figure 17. Tiling of the STFT, WT, and the WPT ............................................................... 28
Figure 18. Recommendation for Surface Electrode Locations on the Forearm ....................... 30
Figure 19. Forearm Motions: Fist Clench, Fist Stretch, Wrist Supination, Wrist Pronation ....... 32
Figure 20. Example of a Neuron ......................................................................................... 36
Figure 21. Sigmoid Function .............................................................................................. 36
Figure 22. Architecture and Structure of a Feed-Forward Neural Network ....................... 36
Figure 23. Results of the Dynamic Recurrent Neural Network at Predicting the Position of a Figure Eight for a Test Subject’s EMG Signals ........................................................................... 38
Figure 24. Architecture of the TDNN Used by Au and Kirsch ........................................... 40
Figure 25. Optimal Placement of Electrodes for EMG Classification of the Forearm .......... 44
Figure 26. Optimal Placement of Electrodes for EMG Classification of the Elbow .............. 44
Figure 27. 5 DOF Robotic Arm Used in the Biomechatronic Learning Laboratory ............. 47
Figure 28. Flowchart Diagram of the Labview Software .................................................... 48
Figure 29. Lab Environment in the Biomechatronic Learning Lab with the CCS Subject ...... 51
Figure 30. Upper Extremity Motion Capture System ............................................................ 54
Figure 31. Full Range of the Elbow During Single Joint Elbow Motion .............................. 55
Figure 32. Full Range of the Shoulder During Single Joint Shoulder Motion ................... 55
Figure 33. Example of Reaching Motion ............................................................................. 56
Figure 34. Block Diagram of the Algorithm for the TDNN ................................................. 57
Figure 35. Percentage of Each Class that was Misclassified out of the Total Number of Misclassifications for the Healthy Subjects ........................................................................... 76
Figure 36. Percentage of Each Class that was Output from the Classifier During Misclassifications of the Rest Class ........................................................................................................... 76
Figure 37. Percentage of Each Class that was Output from the Classifier During Misclassifications of the Grasping Class ........................................................................................................................................... 77
Figure 38. Percentage of Each Class that was Output from the Classifier During Misclassifications of the Releasing Class ........................................................................................................................................... 77
Figure 39. Percentage of Each Class that was Output from the Classifier During Misclassifications of the Pronation Class ........................................................................................................................................... 78
Figure 40. Percentage of Each Class that was Output from the Classifier During Misclassifications of the Supination Class ........................................................................................................................................... 78
Figure 41. Percentage of Each Class that was Output from the Classifier During Misclassifications of the Extension Class ........................................................................................................................................... 79
Figure 42. Percentage of Each Class that was Misclassified out of the Total Number of Misclassifications for the CCS Subject ........................................................................................................................................... 79
Figure 43. TDNN Output Subject 1 Performing Slow Elbow Movement ......................................................... 81
Figure 44. TDNN Output Subject 1 Performing Fast Elbow Movement ............................................................ 85
Figure 45. TDNN Output Subject 1 Performing Slow Shoulder Movement ..................................................... 86
Figure 46. TDNN Output Subject 1 Performing Fast Shoulder Movement ...................................................... 86
Figure 47. TDNN Output Subject 1 Performing Slow Reaching Movement ................................................... 87
Figure 48. TDNN Output Subject 1 Performing Fast Reaching Movement ..................................................... 87
Figure 49. TDNN Output Subject 2 Performing Slow Elbow Movement ......................................................... 88
Figure 50. TDNN Output Subject 2 Performing Fast Elbow Movement ............................................................ 88
Figure 51. TDNN Output Subject 2 Performing Slow Shoulder Movement ..................................................... 89
Figure 52. TDNN Output Subject 2 Performing Fast Shoulder Movement ...................................................... 89
Figure 53. TDNN Output Subject 2 Performing Slow Reaching Movement ................................................... 90
Figure 54. TDNN Output Subject 2 Performing Fast Reaching Movement ..................................................... 90
Figure 55. TDNN Output Subject 3 Performing Slow Elbow Movement ......................................................... 91
Figure 56. TDNN Output Subject 3 Performing Fast Elbow Movement ............................................................ 91
Figure 57. TDNN Output Subject 3 Performing Slow Shoulder Movement ..................................................... 92
Figure 58. TDNN Output Subject 3 Performing Fast Shoulder Movement ...................................................... 92
Figure 59. TDNN Output Subject 3 Performing Slow Reaching Movement ................................................... 93
Figure 60. TDNN Output Subject 3 Performing Fast Reaching Movement ..................................................... 93
Figure 61. TDNN Output Subject 4 Performing Slow Elbow Movement ......................................................... 94
Figure 62. TDNN Output Subject 4 Performing Fast Elbow Movement ............................................................ 94
Figure 63. TDNN Output Subject 4 Performing Slow Shoulder Movement ..................................................... 95
Figure 64. TDNN Output Subject 4 Performing Fast Shoulder Movement ...................................................... 95
Figure 65. TDNN Output Subject 4 Performing Slow Reaching Movement ................................................... 96
Figure 66. TDNN Output Subject 4 Performing Fast Reaching Movement ..................................................... 96
Figure 67. TDNN Output Subject 5 Performing Slow Elbow Movement ......................................................... 97
Figure 68. TDNN Output Subject 5 Performing Fast Elbow Movement ............................................................ 97
Figure 69. TDNN Output Subject 5 Performing Slow Shoulder Movement ..................................................... 98
Figure 70. TDNN Output Subject 5 Performing Fast Shoulder Movement ...................................................... 98
Figure 71. TDNN Output Subject 5 Performing Slow Reaching Movement ................................................... 99
Figure 72. TDNN Output Subject 5 Performing Fast Reaching Movement ..................................................... 99
Figure 73. TDNN Prediction of Elbow Position for a Subject with CCS ............................................................ 100
Figure 74. Final EMG Locations chosen for the Three DOF Controller for the CCS Subject ......................... 110
List of Tables

Table 1. Summary of Muscles for Elbow Flexion and Extension and Wrist Pronation and Supination .... 13
Table 2. Summary of Muscles for Finger Flexion and Extension .............................................................. 14
Table 3. Randomized Commands for Each Subject for Real Time Myoelectric Control Testing.............. 50
Table 4. Randomized Commands for the Three Single DOF Controllers for the Subject with CCS ........... 52
Table 5. Classification Accuracies of the Offline Window and Feature Analysis for Female Subject 1 ... 59
Table 6. Classification Accuracies of the Offline Window and Feature Analysis for Female Subject 2 ... 60
Table 7. Classification Accuracies of the Offline Window and Feature Analysis for Female Subject 3 ... 61
Table 8. Classification Accuracies of the Offline Window and Feature Analysis for Female Subject 4 ... 62
Table 9. Classification Accuracies of the Offline Window and Feature Analysis for Female Subject 5 ... 63
Table 10. Classification Accuracies of the Offline Window and Feature Analysis for Male Subject 1 ...... 64
Table 11. Classification Accuracies of the Offline Window and Feature Analysis for Male Subject 2 .... 65
Table 12. Classification Accuracies of the Offline Window and Feature Analysis for Male Subject 3 .... 66
Table 13. Classification Accuracies of the Offline Window and Feature Analysis for Male Subject 4 .... 67
Table 14. Classification Accuracies of the Offline Window and Feature Analysis for Male Subject 5 .... 68
Table 15. Summary of Classification Accuracies for Healthy Subjects Using a 250ms Adjacent Windowing Scheme .................................................................................................................................... 69
Table 16. Comparison of an Adjacent Windowing Scheme with a Majority Voting Scheme .................. 69
Table 17. Real Time Pattern Recognition Myoelectric Control Scheme Results for Female Subject 1 .... 71
Table 18. Real Time Pattern Recognition Myoelectric Control Scheme Results for Female Subject 2 .... 71
Table 19. Real Time Pattern Recognition Myoelectric Control Scheme Results for Female Subject 3 .... 72
Table 20. Real Time Pattern Recognition Myoelectric Control Scheme Results for Female Subject 4 .... 72
Table 21. Real Time Pattern Recognition Myoelectric Control Scheme Results for Female Subject 5 .... 73
Table 22. Real Time Pattern Recognition Myoelectric Control Scheme Results for Male Subject 1 ...... 73
Table 23. Real Time Pattern Recognition Myoelectric Control Scheme Results for Male Subject 2 ...... 74
Table 24. Real Time Pattern Recognition Myoelectric Control Scheme Results for Male Subject 3 ...... 74
Table 25. Real Time Pattern Recognition Myoelectric Control Scheme Results for Male Subject 4 ...... 75
Table 26. Real Time Pattern Recognition Myoelectric Control Scheme Results for Male Subject 5 ...... 75
Table 27. Summary of the Real Time Results for all of the Healthy Subjects. ................................................................. 76
Table 28. Results of the Single DOF Controllers used for the CCS Subject ........................................... 80
Table 29. Results of the Three DOF Controller for the CCS Subject ............................................................ 81
Table 30. Results for the Thirty-Six TDNNs for Subject 1 ........................................................................ 83
Table 31. Summarized Results for Subject 1 .......................................................................................... 84
Table 32. TDNN Results for All Subjects for All Movements Using Optimal TDNN Parameters .......... 84
Table 33. Overall Error for the TDNNs for All Subjects ........................................................................ 84
Abstract

The work presented in this thesis consists primarily of two projects. The first project was the design and development of a real time myoelectric controller using a pattern recognition scheme. The myoelectric control scheme controlled three degrees of freedom which included elbow flexion and extension, wrist pronation and supination, and hand grasping and releasing for a robotic arm in the Biomechatronics Learning Lab at the Rochester Institute of Technology. According to the knowledge of the author, no work has ever combined these three DOF. The design started with an offline analysis of common windows and features found in the literature. Data was obtained from ten healthy subjects and was tested to find the optimal window and feature scheme which provided for the highest classification accuracy. The highest classification accuracy was 94.92% for a 250ms windowing scheme with three autoregressive features from a fourth order model. The classifier used in all of the testing was a linear discriminate analysis. The real time myoelectric control scheme was implemented in Labview and used an adjacent windowing scheme of 250ms with AR features. The same ten healthy subjects were then used to test the real time myoelectric control scheme. The average classification accuracy during the real time testing was 89.52%. The real time myoelectric control scheme was then adapted to a subject with Central Cord Syndrome. He was first tested with single degree of freedom controllers and obtained classification accuracies of 100%, 95.24%, and 80.95% for elbow, hand, and wrist controllers respectively. When tested with the full three degrees of freedom controller, the subject achieved a classification accuracy of 68.25%.

The second project involved the theoretical design of a myoelectric controller based on a time delayed neural network. Advantages to this design were that the EMG used for this control scheme were based on complex reaching motions and allowed for the control of multiple degrees of freedom at once. These two advantages are not currently offered in myoelectric control schemes which typically control one degree of freedom at a single instance in time and are based on repeatable isometric contractions. The algorithm was based on previous work completed by Au and Kirsch. Their optimal parameters used a
total delay of 875ms and 125ms as the interval of delay. The total delay suggested by Au and Kirsch is not possible for a real time scheme. This work tested for the feasibility of using the time delayed neural network as a real time myoelectric control scheme by decreasing the total delay. Five subjects were used in this work. All of the time delayed neural networks were trained using multiple types of motion and speeds to make the TDNNs robust. The first subject was tested with different TDNNs that used total delays of 900ms, 600ms, and 300ms, delay intervals of 50ms, 100ms, and 150ms, and hidden layer neurons of 10, 20, 30, and 40. The optimal parameters of a 300ms total delay with a 100ms delay interval, and a hidden layer of 10 neurons resulted in an average error of 15.7° for the first subject. These parameters were then used to test the data from the remaining subjects. Using the optimal parameters an average error of 19.0° for all subjects was obtained. Previous errors reported by Au and Kirsch were on the order of 20°. This work showed that the total time delay could be decreased. The next step for this work would be to implement the algorithm in real time or make attempts to decrease the output error.
Chapter 1: Introduction

The potential for robotics to play a vital role with humans is growing. It is predicted that the population of senior citizens in the United States will double from forty million to eighty million by the year 2050. As there is an increasing population of elderly, there is also an increase in the occurrence of age-related disorders and diseases. These age related disorders and diseases include cerebral vascular accident (stroke), cerebral palsy, multiple sclerosis, spinal cord injury, and Parkinson’s disease [1]. With an increase in the amount of the population suffering from these disorders and diseases, the need to provide solutions of recovery and therapy also increases. Robotics has the potential to provide a variety of solutions to these problems. Using robotics to overcome the aforementioned diseases and disorders is the goal of rehabilitation robotics. The field of rehabilitation robotics consists of using robotic technology and mechatronics to provide disabled people with the tools necessary to provide a better quality of life [2].

It has already been shown that rehabilitation robotics has the ability to provide solutions to many conditions and problems. The last ten years has seen great growth in this research area. The use of robots in the recovery of stroke has shown great promise. Krebs and Hogan have developed a set of modular robots that can be used in single pieces or as a whole set. This allows them to target specific muscles individually or a whole group of muscles at one time [3], [4]. They have developed robots to support the shoulder, elbow, wrist, hand, and even the ankle. They are used in a person’s recovery after a stroke. Their most famous robot is the MIT-MANUS which is for the upper limb. They have tested their rehabilitation robots on over 300 patients and their results prove that robotic therapy applied after a stroke is capable of increasing a person’s motor recovery. In order to see the broad range of disabilities and diseases that robotics can be applied to, the Newman Laboratory for Biomechanics and Human Rehabilitation at MIT, which is led by Hogan, started to use their developed robots for other conditions and disorders [1]. They found positive results when applying robotic therapy to cerebral palsy, multiple sclerosis, spinal cord injury, and Parkinson’s disease.
Among the research and development that has taken place in the field of rehabilitation robotics is a group of researchers that have integrated the use of electromyography (EMG) into their projects. Song used the EMG signal as a source of control for a rehabilitation robot application [5]. He used the triceps EMG signal as a source of control for a robot that assisted stroke patients in elbow extension. Rosen also used the EMG signal as a source of control for an exoskeleton system which aided in the movement of the elbow [6]. The processed EMG signal of the biceps, brachioradialis, and triceps were the primary input along with joint kinematics to a model which predicted muscle moments on the elbow joint that were used to control an exoskeleton system. Also important in the use of rehabilitation robotics is the use of biofeedback for both the user and the therapist or clinician. Some have chosen to use the EMG signal as a source of feedback [2]. The EMG signal has also been found in similar applications such as functional neuromuscular stimulation [7] and prosthetics [8].

The above discussion on rehabilitation robotics and the applications using EMG describes the framework for the work that is being proposed here. Using the EMG signal as a source of control is called myoelectric control and its use for rehabilitation robots will be studied in this work. The goal of this work is to analyze the feasibility of using EMG signals as a source of control for a rehabilitation robot. The objective of this work is to not only to develop a myoelectric control scheme but to also test the real time control scheme for a rehabilitation robot. Following this introductory section is the background and the methodology. The background includes all of the literature review and background reading that has been completed. Major sections included in the background are a review of the anatomy and physiology of human motor control, muscles controlling the upper limb, electromyography, pattern recognition myoelectric control, and model based myoelectric control. The methodology includes the steps and processes that were followed to analyze and develop a myoelectric control scheme for a rehabilitation robot. It also includes how the developed myoelectric control scheme was evaluated and tested. The results are presented after the methodology and a thorough analysis of the results is assessed in the discussion. Major highlights of the work are presented in the conclusion followed by a section on future work to be done.
Chapter 2: Background

2.1 Anatomy and Physiology of Human Motor Control

Movement and motion of the human body originates in the central nervous system prior to any muscle contractions. Figure 1 displays a visual of how the central nervous system plays a role in motor control. The starting point of motor control is located in the areas of the cortex in the brain. Outputs from the motor cortex have an impact on interneurons and motor neurons of the spinal cord. Interneurons lie between motor and sensory neurons and are contained mostly within the central nervous system where integration takes place. The connection of the nervous and muscular system is the motor neuron. Motor neurons have direct control over the muscles they connect to. Motor neurons and the muscle fibers they connect to make up the fundamental block of motor control which is known as the motor unit. A motor unit is defined as the motor neuron in the spinal cord and the muscle fibers it innervates [9].

Figure 1. A visual display of the central nervous system and motor control [9].
Studies have shown that different types of motor units exist and three types have been identified. Physiological properties such as speed of contraction and fatigability of the motor units differentiate the types. The three types are fast-twitch and fatigable, fast-twitch and fatigue resistant, and slow-twitch and highly fatigable, which are sometimes termed type IIb, type IIa, and type I respectively. The number of motor neurons in a muscle varies depending on the muscle location and muscle size. It can range from 100 for a small muscle or more than 1000 for larger muscles. The typical motions and forces exerted by a muscle determine the type and number of motor units in a muscle [9]. Muscle fibers that need fine control have a small number of motor units while forces needed to produce a large amount of force have a large amount of motor units [10].

A motor unit controls all of the muscle fibers that it innervates with. When a motor unit fires, all of the muscle fibers connected to it will contract. Muscle fibers from the same motor unit are not necessarily in the same location of the muscle and can be spread throughout the muscle. Therefore, a single firing of a motor unit can induce a small contraction of the whole muscle [10]. During voluntary contractions, the force is controlled by the number of motor units recruited and the frequency at which the motor units fire. The exact methodology of motor unit recruitment is not exactly known, but it is currently believed that motor units are recruited in order of increasing size of the motor neuron. Several factors affect the recruitment process such as type of contraction, muscle fatigue, and oxygen availability [9].

When a muscle fiber contracts it must first be stimulated by a nerve ending. As a result of the stimulation, the muscular membrane depolarizes and propagates an action potential down the length of the muscle fiber. The depolarization occurs due to voltage controlled sodium channels. The positively charged sodium ions enter the membrane and initiate the action potential [10]. Other charged ions such as calcium, potassium, and chloride also play a vital role in producing and propagating the action potential. A model describing the electrical interchange of ions during the action potential was developed by Hodgkin and Huxley during their study of a squid axon. The Hodgkin-Huxley schematic model is displayed in Figure 2. The three main ion channels which control the electrical properties of the
membrane are included in the model. The model does not fully encompass all of the complexities of the muscle membrane, but does a good job at describing the general behavior of the muscle membrane [9].

![Figure 2. The schematic of the Hodkin-Huxley Model [9].](image)

2.2 Electromyography

Electromyography (EMG) is the study of the electrical activity of the muscular system that is recorded by electrodes. The electrical activity can be measured by fine wire electrodes that are inserted into the muscle with the purpose of recording a very small number of muscle fibers. Typically fine wire electrodes are only used in research or clinical trials and must be inserted by someone with the proper knowledge of the musculature. A more common technique in recording EMG is to record the electrical activity of the muscles using surface electrodes located on the skin. This type of EMG is termed sEMG. Because the recording of the muscle activity is at the skin, the EMG is a recording of the summation of all of the electrical activity in the surrounding muscle fibers. The sEMG is a good measure of what the
muscle is doing as a whole [9]. All of the work completed and presented here assumes the recording of surface EMG as the technique used.

A number of important issues go into the recording of EMG which includes electrode size, electrode distance, and electrode location. It is also important to note the affect of using electrodes on the skin and how it affects the recorded EMG signal. The ideal case is to have a point electrode connected to an EMG amplifier that has infinite input impedance, but realistically this is never the case. The electrode always has physical dimensions, the skin electrode interface adds impedance, and the amplifier has finite impedance. There are a number of other sources such as power line interference that can also contribute noise to the EMG signal. The electrical model of the typical EMG recording is shown in Figure 3 [9].

![Figure 3. The model of the electrode to skin interface during EMG recording [9].](image)

Several thoughts should be noted about using surface electrodes. The skin is a moderately conductive material whose cells’ electrical activity is carried out by the movement of ions; This is in contrast to the electrode which is a metal that is highly conductive and its current is achieved through the movement of electrons. This results in a noisy skin electrode interface. This interface also contributes capacitive impedance creating a frequency response. It also contributes a DC component termed the “battery” potential. Applying an electrode to the skin will also cause the skin that it is in contact with to
maintain the same voltage potential across all of the skin that the electrode is in contact with. This modifies the “real” electrical potential of the skin. The use of the electrode in conjunction with the EMG amplifier will result in adding an offset and noise to the signal of interest that cannot be avoided [9].

As a result of the above considerations, several generalities can be made when recording EMG signals. Typically smaller electrodes are preferred and electrodes larger than 5mm results in a loss of information. Shaving and cleaning the skin surface and then applying conductive gel can improve the electrical contact of the electrode. The most common electrode configuration used by engineers is the differential configuration. This configuration is displayed in Figure 4. The preferred interelectrode distance is approximately 20mm [9].

![Figure 4. The differential configuration for EMG amplifier [9].](image)

The input impedance to an EMG amplifier is suggested to be at least 1000MΩ or two orders of magnitude greater than the largest skin to electrode impedance. Skin to electrode impedances can be as large as 1MΩ. Vital to EMG amplifiers is a high common mode rejection ratio (CMRR) to eliminate noise. System capacitances also play a major role in amplifier design. Most EMG systems introduce a high-pass filter at about 10-20Hz to eliminate the dc offset and low frequency artifacts. They will also have a low-pass filter with a cutoff at 400-450Hz. For most EMG applications the highest frequency of interest in the EMG signal is 400-450Hz which requires a sampling rate of at least 1000Hz in order to
keep the Nyquist rate. Therefore, the low pass filter acts as an anti-aliasing filter to remove high frequency noise prior to analog to digital conversion. In some cases, a notch filter to eliminate the power line frequency is also included, but this is not recommended as the EMG spectrum includes this frequency [9].

2.3 Muscles of Elbow and Forearm Movement

Two movements of the elbow joint are flexion and extension. Flexion is controlled by three different muscles. The first and strongest is the brachialis. The two other elbow flexors are the biceps brachii and the brachioradialis which decrease in strength respectively. The brachialis and biceps contract simultaneously during flexion and insert into the ulna and radius of the forearm respectively. The biceps brachii is the more familiar elbow flexor and is more superficial than the brachialis which lies just beneath the biceps. The brachioradialis is a weak elbow flexor. Figure 5 and Figure 6 display the biceps and the brachialis respectively [10].

![Figure 5. The biceps brachii [10].](image)

![Figure 6. The brachialis [10].](image)

The opposing motion of flexion in the elbow is extension. The primary muscle responsible for elbow extension is the triceps brachii which is an antagonist of the elbow flexors. The triceps is the only muscle located on the posterior portion of the upper arm. A synergist aiding the triceps in elbow extension is the anconeus which is a small muscle located at the end of the triceps at the elbow. Figure 7 displays both the triceps and the anconeus [10].
The forearm musculature has more muscles than the upper arm. The muscles of the forearm can be divided into two groups. One group of muscles controls the fingers and thumb while the other group is responsible for wrist movements. Many of these muscles taper to long insertion tendons that are anchored by strong ligaments around the wrist. Another way to group and classify the forearm muscles is into anterior flexors and posterior extensors [10].

Two anterior muscles of interest that do not contribute to wrist flexion are the pronator teres and the pronator quadrates. These two muscles are responsible for forearm pronation. The pronator quadrates is the prime mover of pronation and is the deepest muscle of the forearm by the wrist. The pronator teres is located in between the brachioradialis and the flexor carpi radialis. Both the pronator quadrates and the pronator teres are shown in Figure 8. Although it is located in the upper arm, the biceps brachii is also responsible for forearm supination. An antagonist to the muscles responsible for pronation is the supinator which assists the biceps brachii in forearm supination. The supinator is a deeply located muscle posterior at the elbow and can also be seen in Figure 8 and Figure 9 [10]. The motions of forearm pronation and supination are displayed in Figure 10. Table 1 displays a summary of the motions covered up to this point and what muscles are responsible for those motions.
Figure 8. The pronator quadrates, pronator teres, supinator, and other forearm muscles [10].

Figure 9. Muscles of the forearm and hand [10].
There are many muscles that contribute to controlling the hand because of the fine movements that are required of the hand and its many degrees of freedom. Most muscles that control the hand are located in the forearm and can be seen in Figure 8 and Figure 9. The more precise movements of the hand and fingers are assisted and made by muscles solely located in the hand. Two hand motions of interest are finger flexion and finger extension which would correspond to the intentions of hand grasping and hand opening or releasing. As mentioned earlier, the flexors are located in the anterior portion of the forearm and the extensors in the posterior compartment. The muscles responsible for finger flexion are the flexor digitorum superficialis and the flexor digitorum profundus. The muscles responsible for finger extension are the extensor digitorum and the extensor indicis. The flexor pollicis longus controls flexion of the thumb while the abductor pollicis longus and brevis are responsible for abducting and extending the thumb. The above mentioned muscles are displayed in Figure 8 and Figure 9 and are summarized in Table 2 [10].
Table 2. A summary of muscles and what motions they are responsible for producing (Copied From [10]).

<table>
<thead>
<tr>
<th>Muscles</th>
<th>Finger Flexion</th>
<th>Finger Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexor Digitorum Superficialis</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Flexor Pollicis Longus</td>
<td>X (Thumb)</td>
<td></td>
</tr>
<tr>
<td>Flexor Digitorum Profundus</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Extensor Digitorum</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Abductor Pollicis Longus</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Extensor Pollicis Longus and Brevis</td>
<td></td>
<td>X (Thumb)</td>
</tr>
<tr>
<td>Extensor Indicus</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

2.4 Central Cord Syndrome

In myoelectric control systems, it is important to consider the end user of the control scheme. The majority of the time the designed myoelectric control system is not meant for a healthy person. And yet the majority of work in this area is used with healthy subjects. Typically the end user of a myoelectric control scheme is someone with a neuromuscular condition or an amputee. One of the goals of this work was to develop a foundational myoelectric control scheme for healthy subjects and then to extend that scheme to someone with a disability. Included in this work is the development of a myoelectric control scheme for someone with Central Cord Syndrome (CCS).

CCS is the most common form of incomplete spinal cord injury. It is marked by disproportionately more motor loss and impairment in the upper limbs relative to the lower limbs. Other indicators of CCS include bladder dysfunction and varying degrees of sensory loss [11]. It is often seen in older patients with cervical spondylosis which is a condition caused by abnormal wear on cartilage and vertebrae. When the cartilage wears away mineral deposits can form between the vertebrae. Older people with cervical spondylosis can experience injury due to hyperextension that results in pinching of the spinal cord [12]. Although CCS occurs most often in older persons, it can occur in those of any age brought on by a variety of injuries to the spinal cord [13]. Another condition sometimes associated with CCS is spasticity. Spasticity is an involuntary contraction of muscles which results from the nervous system sending signals to the muscles to contract although the person is not trying to contract. Both
Dvorak and Tow included a discussion on spasticity in their analysis of CCS [14], [11]. Due to the compression and pinching that can occur to the spinal cord, some have reported the results of decompressive surgery [15]. It has been reported that those with CCS can have positive neurologic and functional recovery [13]. Although many people are able to regain some neurologic and functional recovery, typically it is not a full recovery. Many CCS patients suffer from neurological deficits that interfere with daily activities [16].

2.5 Pattern Recognition Myoelectric Control Systems

2.5.1 General Overview

The heart of myoelectric control is the fact that the primary source of control information for the system being controlled comes from the EMG signal. Myoelectric control is the technique of using signal processing to extract information from the EMG signal in order to determine a person’s intended motion and aid them in controlling an assisting device. The typical approach to myoelectric control is to use a pattern recognition scheme. This approach recognizes one of several predetermined classes. These classes represent certain motions such as elbow flexion and extension. If elbow flexion is identified then flexion is performed by the controlled device from the relative position. The pattern recognition approach does not predict and identify specific reference angles or positions of joints and limbs. Rather, it defines a specific motion relative to the current position or a whole range of motion to be performed that once it is activated cannot be altered. Myoelectric pattern recognition systems are also dependent on repeatable isometric contractions. Isometric contractions occur when the muscle has reached a state of tension but the muscle is neither increasing nor decreasing in length [10]. In other words, it is a stationary contraction where a constant force is produced. Most pattern recognition systems are also fixed to movement in no more than a single degree of freedom (DOF) at a single instance in time.

A block diagram representing a myoelectric pattern recognition control scheme is shown in Figure 11. The test subject ultimately controls the end device with their EMG signals. Prior to any pattern recognition analysis, the EMG data acquisition is accomplished using surface electrodes and a
bioinstrumentation device. Included in the EMG data acquisition block is any data segmentation or windowing which involves dividing up the EMG signal into windows. It also includes any preprocessing techniques such as filtering. Many bioinstrumentation devices provide the necessary analog filtering for EMG applications. The core pattern recognition blocks are feature extraction and classification. EMG feature extraction is a very important step and can determine the efficiency and accuracy of the classifier. The goal of feature extraction is to use signal processing techniques to extract specific features or information from the signal that can be discriminated into types of classes that represent specific motions. The next step involves classification which is not performed on the raw EMG signal directly, but rather on the EMG features extracted in the previous step. The classification step makes a decision upon what class the features belong to. The classes are movements or motions established beforehand such as elbow flexion or wrist pronation. Once the classifier has identified a specific motion or movement, the controller outputs commands to the assisting device or prosthetic. The rest of this chapter includes sections on EMG data acquisition, feature extraction, and classification. Each section discusses the work that has previously been completed in these areas.

Figure 11. A block diagram of the myoelectric control scheme using pattern recognition.
2.5.2 EMG Data Acquisition

There is widespread agreement among the work completed in myoelectric control about certain aspects of EMG data acquisition. Almost all researchers choose a sampling frequency close to 1000Hz. This sampling frequency is chosen because the maximum frequency content of the EMG signal is less than 500Hz and this allows one to maintain the Nyquist rate as stated previously in the section on electromyography. Also mentioned in the electromyography section are the typical filtering techniques and the majority of previous work shows a consensus of band pass filtering from 10-20Hz to 400-450Hz \[9\]. Because of the similarities among sampling frequency and filtering among previous work, this section will focus primarily on the different types of segmentation or windowing that has been completed.

The assumption often made in literature is that a practical myoelectric controller must consider the constraints of real time. Making myoelectric control schemes real time results in placing several constraints upon the design of myoelectric control schemes that are not necessarily included in offline pattern recognition schemes. A review of literature has shown that the data segmentation length plus the processing time should be no more than 300ms for a myoelectric control scheme. It is important to maintain this constraint so that the user does not perceive any delay. This enables the control scheme to be user friendly. Typically there is a tradeoff between the classification accuracy and the response time of the system, but it is important that the system delay stays less than 300ms \[17\].

The EMG segment or window length of analysis should be large enough that the extracted features maintain consistency and the variance among samples are minimized. Several different data segmentation approaches have been developed. Depending on the type of control scheme being implemented data segment windows will typically range from 30ms to 250ms \[17\]. One of the first to implement a pattern recognition myoelectric control scheme was Hudgins \[8\]. He used a windowing scheme that has been termed the adjacent windowing technique. This technique uses a predefined window length and breaks the EMG signal into these lengths. In each of these adjacent segments features are extracted and then a class decision is made after a processing delay represented as \(\tau\). The total delay using
this technique is the length of the window plus the processing delay. The adjacent windowing technique is displayed in Figure 12.

Figure 12. The adjacent windowing technique [17].

In the adjacent windowing scheme, no processing is performed between the time that the processing of the previous window is completed and the time that the data for the next window is being collected. A second windowing technique takes advantage of the processor down time and is termed the overlapped windowing technique. This scheme uses the processor idle time to make more classifier outputs by continuously making classifications on data collected in the last defined window length. Because the processing delay is shorter than the window length, this technique results in using repeated data in sequential windows. Essentially the technique can be thought of as sliding a constant length window over the EMG signal and moving the window every time a classifier output is made. The window should be moved over the data for a time that is greater than the processing time needed to make a class decision [17]. This technique is displayed in Figure 13.
Englehart was the first to introduce the overlapped windowing technique, and, at the same time, he introduced a post-processing scheme that he termed majority voting [18]. He noted that the overlapped windowing technique produced a class decision stream that was denser than the adjacent windowing scheme. The majority voting technique used several of these classifier outputs to make a final class decision depending on which class had the most occurrences. Englehart’s majority voting scheme is described below. For the current decision point in time $d_i$, the majority vote decision is noted as $d_{mv}$ and includes the previous $m$ samples and the next $m$ samples. The majority vote decision is then chosen as the highest occurring class in the set of class decisions made in the $(2m + 1)$ window. The number of samples is dependent upon the processing time $\tau$, and the acceptable total delay until system response $T_d$. The number of samples is dependent upon the inequality in Equation 1 [18]. For example if the total acceptable delay is 300ms and the processing time is 30ms, then the highest value of $m$ that satisfies Equation 1 is 10. This means that a total of 21 decisions contribute to the majority vote. Englehart demonstrated that the majority voting technique had the ability to remove many incorrect class decisions that were made alone. Using a smaller analysis window results in a decrease in classifier accuracy, but it will also decrease the processing time allowing more votes in the majority voting scheme that can result in an increase in classification accuracy. Englehart’s best results came from using an analysis window of
32ms with an acceptable delay of 128ms which is less than the 300ms requirement of myoelectric control delay. An example comparing Englehart's majority voting scheme versus the unprocessed classifications is shown in Figure 14. It is easily seen how the majority voting offers a more robust system relative to the single classification outputs [18].

![Figure 14](image.png)

Figure 14. An example of Englehart's majority voting scheme compared to the unprocessed classifications using an analysis window of 32ms with a delay of 128ms. The classes corresponded to elbow flexion and extension and wrist pronation and supination [18].

$$\tau \cdot m \leq T_d$$ (1)

### 2.5.3 Feature Extraction

An extensive amount of work has gone into extracting features from the EMG signal. Feature extraction is arguably the most important step in achieving a highly accurate classification. Features need to be extracted from the EMG signal because the raw EMG signal would be impractical to use directly due to its stochastic and random nature. The features extracted from the EMG signal make up a feature vector that can range from one dimension for a simple time domain feature up to one hundred or more dimensions if a time-frequency analysis is used. Feature vectors span what is called the feature space, which is hard to visualize if more than two or three features are used. The goal of feature extraction is to
obtain a feature space that has a clear separation between classes. A simple example of this is displayed in Figure 15. This example considered a two class problem of flexion and extension EMG data. The two features used were the first autoregressive coefficients from a 4th order model of the biceps and triceps EMG channels. A clear division exists between the two classes and each class is clustered with itself. For accurate classification it is essential that each class is clustered with itself and is not overlapping with other classes.

Figure 15. Feature space for a two class pattern recognition problem using autoregressive features.

There have been many proposed features to extract from the EMG signal. All EMG features can be broken up into three main categories: (1) time domain, (2) frequency domain, and (3) time-frequency [17]. The typical evaluation measure for features is typically based on classification accuracy. Other measures such as class separability in the feature space are also used to evaluate features. If a feature has a high class separability measure among classes it will also have high classification accuracy, so classification accuracy is usually the EMG feature evaluation of choice. The ultimate test of a feature is real time implementation of the myoelectric control scheme using that feature and measuring its usability.
in an actual application. The advantage of using classification accuracy is that it is objective and quantitative in nature, but usability is often subjective and qualitative.

The first types of assistive devices and prosthetics that incorporated the EMG signal for control only used the power of the signal as the discriminating feature between different types of movements and contractions [19]. Examples of these simplistic control approaches were the Utah arm and the Boston arm [20], [21]. These approaches typically used an agonist with an antagonist pair of muscles such as the biceps and triceps. The earliest schemes compared the power on the biceps EMG channel with the triceps EMG channel and depending on which channel had a higher power, the assistive device would flex the elbow or extend the elbow respectively. These early approaches were successful, but the use of the EMG signal power as the lone feature limits users to a very small set of limb functions. To achieve more distinguishable movements by using EMG signal power, multiple electrode locations must be used. Another limitation to using power is that EMG activation patterns may have the same power resulting in misclassified patterns and movements [19]. The equation for power is shown in Equation 2 where \( y_k \) is the \( k^{th} \) sample of EMG signal \( y \) whose length is \( N \).

\[
\frac{1}{N} \sum_{k=1}^{N} y_k^2
\]  

Equation 2

One of the first to introduce an alternative to using the power as a feature was Graupe [19], [22], [23]. Graupe used time-series parameters of autoregressive models (AR) and autoregressive moving average models (ARMA) as features to discriminate between different types of muscle contractions and movements. Using the time-series coefficients as features allowed Graupe to extend the number of functions of assistive devices to 3-6 functions. Using time-series features allows more information from the EMG signal to be extracted because the model is time dependent whereas the power feature is time independent. The time-series parameters therefore include information of how the EMG signals change with time. Essential to the discrimination of muscle contractions using time-series parameters is the
inclusion of crosstalk between the muscles being recorded. Earlier methods tried to eliminate crosstalk and locate the muscles active during contraction which only allowed for one or two classes to be identified. It is therefore advantageous to use time-series parameters over the earlier methods [22].

The equation defining the AR time-series model is shown in Equation 3 where \( y_k \) is the \( k^{th} \) sample of EMG signal \( y \), \( a_i \) are the corresponding autoregressive coefficients used as features, and \( w_k \) is white noise. Graupe stated that the EMG signal obtained from surface electrodes was stochastic in nature and could be modeled as shown in Equation 3. The EMG signal can be considered to be the output of a filter where the filter is physically the skin and tissues underneath the surface electrodes. The input to the filter is a set of pulse trains that correspond to the firing of individual motor units. The above assumptions about the EMG signal can be seen in Figure 16. An important consideration must be highlighted when considering the EMG signal to be a time-series model. The EMG signal is a nonlinear and nonstationary signal that violates the assumptions made for a time-series model. Fortunately, the EMG signal can be considered a piece-wise stationary signal in intervals from 0.1s to 0.3s. The typical windowing schemes that are used during pattern recognition processing are inside the piecewise stationary range allowing the time-series model to hold. Graupe also commented that the AR model and its parameters convey information about the spectrum of the EMG window being analyzed. The AR model is actually an infinite series but can be shortened while maintaining a good approximation and a minimal error. Graupe stated that a model order of 4 does a good job at estimating the spectrum for EMG signals [22].

\[
y_k = \sum_{i=1}^{m} a_i y_{k-i} + w_k, k = 0, 1, 2, \ldots
\]  
(3)
One should also remember that the goal is not necessarily an accurate representation of the EMG spectrum when using AR coefficients, but rather the importance should be placed on the ability of the features to discriminate different types of contractions and movements. Graupe also stated that the muscle contractions and patterns do not need to be related to the actual limb functions itself. He placed the importance on the ability of a subject to control the device. He also stressed the ability of the user to reproduce muscle contractions that separated the feature space and were easily distinguishable from the other classes [22]. Zardoshti-Kermani also proposed the concept of looking at types of muscle contractions that were easily separable in the feature space [24]. They tested nine different classes of contraction between the biceps and triceps where each muscle was contracted at a high, low, or no level of contraction. They then selected the five most separable classes to match to different movements for a prosthetic device.

In the time since Graupe’s work there have been many others that have investigated the use of time-series models as features of the EMG signal. A lengthy analysis of using an autoregressive model was presented by Hefftner which includes a defense of using the AR model for the EMG signal [25]. Hefftner then explained and presented the actual implementation of using the AR features for functional
neuromuscular stimulation (FNS) [26]. In Hefftner’s work she defended and stated that using an autoregressive model as the strategy for pattern recognition and classification is physiologically sound. The AR model is preferable to the ARMA model because the derivation of the ARMA coefficients is more complex compared to the AR coefficients and computation speed is essential in real time applications. In addition, all models can be approximated by an AR model. Hefftner also defended Graupe’s view that the EMG signal not being stationary can be ignored because the EMG signal is considered to be locally stationary. The least squares approach for calculating the AR coefficients was the employed method in Hefftner’s work and was presented by her [25]. The least squares approach was also the technique used by Graupe. Hefftner also explored and analyzed how to choose the model order and her results supported using a 4th order model.

Many others have also chosen to use time-series features in EMG pattern recognition applications. Kelly used a moving average model as features and used a discrete hopfield neural network to find the actual parameters [27]. He then used the first AR coefficient and the signal power as his two features to input into a multilayer perceptron neural network to classify four different classes, which were elbow flexion and extension and wrist pronation and supination. Hargrove looked into the effects of electrode location and displacement on the classification accuracy [28]. The features that he chose to use were the autoregressive parameters in addition to pairing the autoregressive parameters with other time domain features. He also used the AR features while exploring the application of principal components analysis in a ten channel pattern recognition scheme of forearm movements [29].

Others have attempted to make extensions to the regular autoregressive model. These other models are derived from the AR model and are equivalent in representing the AR model. Each of the alternative models can be derived from the others. One of the other models of interest is the cepstrum. The cepstrum is the inverse Fourier Transform of the logarithmic power spectrum and the features returned using this model are termed the cepstral coefficients. The equation defining the cepstrum model is given in Equation 4 where $c_i$ are the M total cepstral coefficients and $a_i$ are the AR coefficients. Knox compared the use of AR features and Cepstral coefficients while classifying elbow flexion and extension.
and wrist pronation and supination. He used an EMG channel from the biceps and a second channel between the biceps and the triceps on the medial side [30]. Knox was able to achieve a 94% average accuracy between the four classes using the cepstral coefficients and he noted that the performance of the cepstral coefficients was not that much greater than using the AR coefficients. Micera also compared AR and cepstral coefficients as features from EMG signals of shoulder muscles while classifying different types of pointing motions [31]. Kang compared the use of AR coefficients with cepstral coefficients while using three different types of classifiers which included the euclidean distance measure, weighted distance measure, and the modified maximum likelihood ratio [32]. His application of interest was of motions of the neck and bilateral shoulder for those who had tetraplegia at the C5/C6 level of the spine. He also analyzed the affect of using what he called an S-type and C-type electrode configuration in which the S-type had an electrode channel on separate muscles and the C-type had an electrode channel located between muscles. For instance an S-type configuration for the biceps and triceps would involve a separate EMG channel on both muscles while the C-type configuration would involve a single EMG channel with one electrode on the biceps and the other on the triceps. He found that when using the cepstral coefficients the accuracy improved by an average of 5% regardless of which electrode configuration or classifier he used.

$$c_i = -a_i, \quad c_i = -a_i - \sum_{n=1}^{i-1} \left(1 - \frac{n}{i}\right) a_n c_{i-n} \quad 1 < i \leq M$$ (4)

There are several common time domain EMG features that have also been used in pattern recognition control schemes. Hudgins introduced five simple time domain features that were implemented in a myoelectric control scheme [8]. These five features are mean absolute value (MAV), mean absolute value slope (MAVS), zero crossings (ZC), slope sign changes (SSC), and waveform length (WL). MAV is calculated using Equation 5 and is an estimate of the average value of the EMG signal $x_i$ in segment $i$
which is $N$ samples in length. The number of segments in the signal is $I$ and $x_k$ is the $k^{th}$ sample in segment $i$.

$$X_i = \frac{1}{N} \sum_{k=1}^{N} |x_k| \quad \text{for } i = 1, ..., I$$  \hspace{1cm} (5)

The MAVS feature is calculated using Equation 6 and is a measure of the change of MAV between window $i$ and window $i+1$.

$$\Delta X_i = X_{i+1} - X_i \quad \text{for } i = 1, ... I - 1$$  \hspace{1cm} (6)

Although the features shown here are calculated using the time domain representation of the EMG signal, measures of frequency can still be acquired. The ZC feature is a simple measure of frequency and is the amount of times that the EMG signal crosses zero in the given EMG signal of interest. The higher the number of ZC corresponds to higher frequency content. The number of ZC start at zero for the beginning of the calculation and the value is incremented by one any time the consecutive samples $x_k$ and $x_{k+1}$ meet the conditions in Equation 7. It can be helpful to place a threshold around zero when calculating the zero crossings because of noise that exists in the signal.

$$x_k > 0 \text{ and } x_{k+1} < 0, \text{ or } x_k < 0 \text{ and } x_{k+1} > 0 \text{ and } |x_k - x_{k+1}| \geq 0.01V$$  \hspace{1cm} (7)

Another feature that measures frequency is SSC. It counts the number of times that the slope of the EMG signal changes signs. Hudgins also recommended implementing a threshold while calculating this feature. The SSC feature starts at zero for the beginning of the calculation and increments by one if the conditions in Equation 8 are met by the three consecutive samples $x_{k-1}$, $x_k$, and $x_{k+1}$.

$$x_k > x_{k-1} \text{ and } x_k > x_{k+1}, \text{ or } x_k < x_{k-1} \text{ and } x_k < x_{k+1}, \text{ and } |x_k - x_{k+1}| \geq 0.01V \text{ or } |x_k - x_{k-1}| \geq 0.01V$$  \hspace{1cm} (8)

The last feature that Hudgins introduced was the WL. WL is the cumulative sum of the EMG signal under analysis and is calculated using Equation 9.

$$l_0 = \sum_{k=1}^{N} |\Delta x_k| \quad \Delta x_k = x_k - x_{k-1}$$  \hspace{1cm} (9)
Hudgins feature set consisted of these five features which made the feature space span five dimensions. Englehart later used these same features except for MAVS when he introduced the overlapped windowing scheme and majority voting [18]. He stated that the MAVS did not offer any new feature discrimination that was not already present with the other four time domain features. Other time domain features do exist but are typically a simple modification of Hudgin’s time domain features. Sometimes other time domain features are defined the same way but are simply given another feature name.

In recent years there has been a lot of work published in the area of EMG feature extraction using time-frequency techniques. The advantage to using time-frequency methods is the ability to obtain frequency information without losing all time information. Many have proposed the use of using time-frequency methods to overcome the nonstationary and nonlinear properties of the EMG signal during transient and dynamic contractions. There are three different types of time-frequency transforms that have been used in EMG signal analysis which are the short time fourier transform (STFT), wavelet transform (WT), and the wavelet packet transform (WPT). The difference between the time-frequency transforms is how they tile the time-frequency plane. The STFT and the WT have a fixed tiling of the time and frequency axes. The WPT has an adaptive tiling where the tiling can be chosen to maximize the tiling for a given application. Different algorithms and methods are used to determine how the WPT should be tiled, but most use a method which minimizes the reconstruction error [33]. A visual of the time-frequency plane for each transform is displayed in Figure 17. For more information about the STFT, WT, and WPT, one is directed to [34], [35].

Figure 17. The tilings of the STFT (a), WT (b), and the WPT (c) [33].
The time-frequency transforms return many features corresponding to the different locations in the time-frequency plain. Depending on the amount of EMG channels used in the myoelectric control system and how deep the decomposition level of the transform, the number of possible features can easily range to one hundred or more features. It is usually not feasible to use all of the possible time-frequency coefficients that are returned from the corresponding transform. An increase in the number of features will result in higher processing time for a classification decision to be made which increases the system delay. The performance of many classifiers also degrades as the number of features increase. With the higher number of features calculated using time-frequency techniques a need has been created for different methods of dimensionality reduction.

The most common form of dimensionality reduction used with EMG signals is principal components analysis (PCA). The idea behind PCA is to reduce dimensionality while retaining as much variability in the dataset as possible. This is accomplished by transforming the dataset to a new set of variables called principle components (PCs) which are uncorrelated. The PCs are also ranked in order of highest variability to the lowest variability which results in the dataset being able to be represented by the first couple PCs. PCA works to remove the redundancy that exists across multiple dimensions of the dataset. Joliffe does a comprehensive analysis and explanation of PCA [36] and Hargrove presented the use of PCA in EMG pattern recognition [29].

Englehart did a comparison of the three mentioned types of time-frequency transforms with time domain features [33]. He investigated a four class problem of elbow flexion and extension and wrist pronation and supination from EMG signals measured on the biceps and triceps. He also used PCA for dimensionality reduction, but also compared PCA with another method called class separability (CS). PCA can be seen as a feature projection method which transforms the original features whereas CS uses a subset of the original features without changing the original features. He also compared the use of two classifiers, a linear discriminate analysis (LDA) and a multilayered perceptron neural network (MLP). Englehart found that PCA outperformed CS. He also found that the MLP outperformed the LDA because of the MLP’s ability to handle higher dimensional input vectors. The best performance occurred when
using a LDA to classify a PCA reduced WPT feature set and achieved an accuracy of 93.75%. He found that progressing from time domain, STFT, WT, to WPT resulted in an increasing accuracy. Englehart later developed a continuous classification scheme based on use of wavelets [37].

The STFT, WT, and WPT were also used as EMG features and compared to time domain and AR features by Hargrove [38]. In this work he also compares the use of six myoelectric classifiers which use surface EMG as input with the same six classifiers that use intramuscular EMG as inputs. Ten different classes of isometric contraction were collected from fifteen surface EMG channels around the forearm and the intramuscular recordings were collected from the pronator teres, supinator teres, flexor digitorum sublimas, extensor digitorum communis, flexor carpi ulnaris, and the extensor carpi ulnaris. One train of thought prior to his experiments was that the intramuscular electrode recordings would give better class discrimination for myoelectric control applications, but Hargrove’s results gave no indication of this. It was reported that the localized recordings of the intramuscular EMG was no better than the surface EMG which included broad information due to crosstalk. He also found that by carefully choosing three of the fifteen possible channels of surface EMG, 97% classification accuracy could be obtained. These three electrode locations are over the extensors/supinators, flexor carpi ulnaris, and flexor digitorum subliminus and are shown in Figure 18. For a reduced class problem of six classes including wrist flexion and extension, forearm supination and pronation, hand open, and hand close, the AR coefficients produced the highest accuracy followed by the WPT.

![Figure 18. Recommendation for surface electrode locations on a cross-sectional view of the forearm](image-url)
Khushaba used the same EMG database for the forearm that Hargrove developed [39], [40]. In both he used the WPT to obtain the initial set of features. He used two different approaches to dimensionality reduction. In one approach he used a two step process where he first computed a fuzzy entropy measure and removed features that fell below a certain threshold and then further reduced the dimensionality by using PCA [40]. Using this method for the ten class problem he was able to attain an accuracy of 99% using only fifteen PCs as an input to a neural network which is slightly better than Hargrove’s 97% accuracy. It was also found that the majority of the remaining features were obtained from the EMG channels of the extensors/supinator, flexor carpi ulnaris, and flexor digitorum subliminus which supports Hargrove’s findings that these three locations are good enough to perform myoelectric control rather than all fifteen locations. In Khushaba’s other work he implemented a dimensionality reduction technique that he termed Particle Swarm Optimization (PSO) [39]. He was also able to achieve 99% accuracy from a neural network classifier for the total dataset of the ten classes.

Similar approaches of dimensionality reduction for the wavelet transforms have been done by others. Chu used a linear-nonlinear dimensionality reduction of wavelet coefficients using a self organized feature mapping (SOFM) after an initial PCA [41]. Using a neural network as his classifier he was able to obtain 97% accuracy of a nine class problem of forearm and hand motions for the control of a multifunction myoelectric hand. Up to this point all of the presented time-frequency feature extraction methods have extracted features from dimensionality reduction of the transform coefficients directly. Others have attempted to use the time-frequency transforms to extract features from the coefficients in other ways. Khezri used the zero crossings and local extrema of the wavelet transform as features [42], [43]. In addition to his wavelet features, he also used MAV, SSC, and AR features to input to a neuro-fuzzy classifier in order to classify hand opening and closing, wrist flexion and extension, hand pinching, and thumb flexion [43]. He averaged 97% accuracy between all the classes. For more about the properties of the local extrema and zero crossings of the wavelet transform see [44]. Another approach of wavelet feature extraction is to use the energy in each level of decomposition as an EMG feature [45], [46]. Yan used a single electrode channel placed across the flexors and extensors of the forearm to recognize fist
clenching, fist stretching, wrist supination, and wrist pronation as seen in Figure 19 [45]. He achieved an average of 96% and 92% accuracy for a fuzzy least squares support vector machine and a neural network classifier respectively.

Figure 19. The four classes of forearm motions, from left to right: fist clench, fist stretch, wrist supination, wrist pronation [45].

A large investigation of features was performed by Oskoei [47]. He investigated a group of time domain features and a group of frequency domain features. His group of time domain features included MAV, WL, ZC, SSC, root mean square (RMS), variance (VAR), William amplitude (WAMP), and two types of modified mean absolute values (MAV1 and MAV2). The frequency domain features included the power spectrum (PS), AR coefficients of a 2nd order and 6th order model (AR2 and AR6), and the mean and median frequencies (FMN and FMD). He investigated all of the mentioned features separately in addition to four sets of features. The first set was the set proposed by Hudgins mentioned previously, MAV, WL, ZC, and SSC, excluding the MAVS feature [8]. The second set was RMS and AR6 as proposed by Huang [48]. The third and fourth sets are similar to the first and second respectively. The third includes only the MAV and WL, while the fourth is RMS and AR2. The features were extracted from four channels on the forearm that included the biarticulate wrist flexor and triarticulate and biarticulate wrist extensor muscles. The six class problem included rest, flexion, extension, abduction, adduction, and keeping the hand at rest. He tested three different classifiers as well. These included a
linear discriminate analysis (LDA), neural network (NN), and a support vector machine (SVM). Oskoei performed an in-depth analysis of several of the pattern recognition steps. He tested segment lengths from 50ms up to 500ms. He found that WL had an almost fixed accuracy for all window lengths and produced the most stable performance among the single features. He also found that using the multifeature sets resulted in more stability to changes in segment length in comparison to all of the single features. After analyzing the window lengths he recommended that a disjoint windowing scheme of 200ms provides high accuracy and that an overlapped segmentation with a length of 200ms with 50ms increments shortens the response time without affecting accuracy. He also stated that all of the frequency domain features introduce a relatively long processing time to calculate relative to the time domain features. He also removed FMN and FMD from the study because of its low ability to discriminate classes. He recommends the use of the time domain multifeature sets as the best features to use for EMG classification because of the relatively light computation time and stability of performance in various window sizes. If a single feature is to be used, he recommends the WL because it performs best in accuracy, stability, and computational load. He compared the LDA, NN, and SVM one by one with single features and multifeature sets. He found that the SVM and LDA performed the best with average accuracies of 95.5% and 94.5%. He also noted that the LDA is easier and faster to train. The NN is the slowest to train. His results for the NN were also not as stable as the LDA and SVM. By implementing the post processing scheme of majority voting he was able to obtain slightly higher accuracies.

2.5.4 Classifiers

A number of classifiers that have been used in the pattern recognition approach to myoelectric control have already been presented while presenting the literature published in EMG feature extraction. The purpose of the classification step in pattern recognition is to take all of the features extracted and make a decision for which class the features belonged to. The classifier can be thought of as a decision maker. The two primary classifiers that have been used in EMG pattern recognition and myoelectric control are the linear discriminate analysis (LDA) and neural network (NN).
LDA is considered to be a parametric type of classification. A parametric approach refers to the assumption that the features belong to a probability density function which is assumed to be known. The most commonly used probability density function used is the Gaussian or normal distribution. The Gaussian distribution is quite often safe to use as it occurs frequently in nature, but one must note that this assumption is made. It should be known though that not all applications can assume a Gaussian distribution [49]. In LDA the underlying distribution of features is assumed to be Gaussian and is sometimes called a Bayesian classifier. The Gaussian distribution is defined in Equation 10 where 
\[ p(x \mid \omega_i) = \frac{1}{(2\pi)^{l/2} |\Sigma_i|^{1/2}} \exp\left( -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) \right) \quad i = 1, ..., M \]

The mean and covariance matrix for a given class is found by using training data of that specific class. The features of that class are extracted from the training data and the mean vector and covariance matrix are found. Once the mean vector and covariance matrix are found for each class, features from an unknown sample can be extracted and input to the probability density function of each class. Because each class has an equal probability of occurring, the Bayes classification rule states that the unknown sample should be classified as belonging to the class that gives the greatest probability from the probability density functions of the classes [50]. A number of EMG pattern recognition applications have used LDA or a similar Bayesian approach and have already been mentioned in addition to several others [28], [29], [32], [33], [37], [38], [46], [47], [51], [52], [53].

The theory of NNs has been applied to a number of different problems and applications. NNs have been used as a classifier for EMG pattern recognition. The goal of using a NN as a classifier is to divide the feature space into different regions corresponding to the different classes. Given a set of features from an unknown sample as input, the output of the neural network will determine which class
the sample belongs to. The LDA was considered to be a parametric approach where the underlying probability was defined by the two parameters, the mean vector and covariance matrix, but NNs are not considered to be parametric. NNs are considered to be nonlinear and make no assumptions about the distributions of features. The basic foundation of NNs is that they behave similar to actual biological neurons. The neurons or nodes receive input from other nodes and produce one output that is often a weighted input to other nodes. The exact connection of nodes is determined by the network architecture which is part of the design process as well as the number of input nodes and layers. The overall goal remains to create decision regions between classes for discrimination by mapping the corresponding inputs to predefined outputs during training [54]. Training data must be used in order for the neural network to learn the proper relationships of features and classes.

An example of a neuron is shown in Figure 20 which has four inputs and whose output can be calculated using Equation 11 where \( f() \) is the transfer function being used and \( w_i \) is the weight corresponding to input \( x_i \). Different types of transfer functions are used in neural networks, but the sigmoid function is used primarily which is shown in Figure 21. The sigmoid function has useful mathematical properties including monotonicity, continuity, and differentiability which are important properties to have when training a neural network. If using the sigmoid function for the neuron in Figure 20, the output would be calculated using Equation 12 where \( i \) is the index of the inputs, \( x_i \) is the input, \( w_i \) is the weighting factor attached to that input, and \( w_o \) is the bias to the neuron. Neural networks have the ability to map very complex decision regions through training. During training the weights are adjusted in order to find the boundaries between classes in the feature space. One of the most familiar classifiers used is the perceptron. The perceptron has the ability to create complex hyperplanes to discriminate classes. The most common network architecture is the feed forward neural network which is displayed in Figure 22 and is capable of mapping nonlinear properties [55]. It is important to note that a neural network's ability to discriminate classes is compromised if overlapping features from different classes exists in the feature space. Several EMG pattern recognition applications have used NN classifiers and have already
been mentioned in addition to several others [8], [27], [33], [37], [38], [39], [40], [41], [45], [47], [56], [57], [58], [59].

\[
    z = f \left( \sum_{i=0}^{3} W_i x_i \right)
\]

\[
    z = \frac{1}{1 + e^{-\left( \sum_{i=0} w_i x_i + w_0 \right)}}
\]

Figure 20. An example of a neuron [55].

Figure 21. The sigmoid function often used as the transfer function of neurons [55].

Figure 22. The architecture and structure of a feed-forward neural network [55].

2.6 Labview Systems

In prototyping a real time myoelectric control system, Labview Version 8.5 was chosen as the development platform. Labview allows for a fast development time where time is saved relative to designing a complete embedded system based on a microcontroller, digital signal processor (DSP), or
field programmable gate array (FPGA). Scientists and engineers have used Labview for a variety of applications which include sophisticated measurement, test, and control systems [60]. The Labview platform allows for the easy integration of hardware devices including data acquisition, motors, and robots. Advanced signal processing algorithms can also be implemented using Labview which allows for taking advantage of a PC’s resources. Labview is a programming language based on visual functions rather than a text based programming language based on lines of code. Although the graphical functions exist, programmers can still use multiple programming languages such as C, C++, or Matlab, while using Labview. Labview systems are also known for their graphical user interfaces which make the final designed system easy to use [61]. A wide variety of applications have been used with Labview. Haptic devices are often used in rehabilitation systems and Black developed a controller for a passive haptic manipulator using Labview [62]. In another biomedical application, Beach used Labview for a system using single photon emission computed tomography (SPECT) which is an imaging technique [63]. A portable system for measuring body movements was also developed in Labview by Bertolotti [64]. In yet another biomedical application, Benitez developed instrumentation for clinical assessment of cardiovascular and autonomic function using Labview [65].

2.7 Model Based Myoelectric Control Systems

It is advantageous to develop myoelectric control systems that are able to identify the actual joint angles of a user’s wrist, elbow, and shoulder during dynamic movements. If accurate joint position could be accurately identified from EMG signals, assisting devices could provide users aid in moving to a specific location. This approach would be more natural for the user as the EMG signals obtained would come from natural movements and contractions rather than repeatable isometric contraction. Although this non pattern recognition approach sounds very attractive, it has yet to be proven that the EMG signal can provide the information needed to predict joint angle position accurately and robustly. Several attempts have been made to identify joint kinematics from EMG signals which could potentially serve as a model based approach to myoelectric control.
Cheron was one of the first to investigate the use of EMG signals to predict positional information of the arm [66]. They restricted movement to the right arm where it was extended and movements were of figure eight motions. EMG signals were obtained during the movements from three deltoid channels, two pectoralis muscle locations, and the latissimus dorsi. At the same time a motion capture system captured the positional data of the figure eight motion. The raw EMG signals were bandpass filtered and digitized at 2kHz. The signals were then full-wave rectified and smoothed by a third-order averaging filter. They then trained what they called a dynamic recurrent neural network (DRNN) to output the position of the figure eight in two dimensions given the EMG data as input. Figure 23 shows the results for one of the subjects. Their work showed that it was indeed possible to predict position from EMG data for this figure eight motion.

![Figure 23. Results of the dynamic recurrent neural network at predicting the position of a figure eight for a test subject’s EMG signals [66].](image)

Suryanarayanan stated that a nonlinear, adaptive, and intelligent system has the potential to track arm movements using EMG signals [67]. His objective was to develop an interface that could track arm movements using EMG signals. The system that they proposed consisted of a neural network and fuzzy logic that took the EMG signals of the biceps to predict the elbow’s joint angle during elbow flexion and
extension. They used a computer model of a robotic manipulator to test their interface. The EMG signals were rectified and a 100 point moving average window was applied and then low-pass filtered. The present and past magnitude and slope of the processed signal were inputs to the system. Because of variations that existed in the EMG signal due to different speeds of flexion and extension, their fuzzy logic system was used to adjust the EMG signal parameters prior to being input to the neural network. They used a three layer feed forward neural network with four neurons in the hidden layer and the output was the elbow angle. They used the backpropogation algorithm to train the system. The results reported that the neural network output had less than 20% error during testing.

Au and Kirsch advanced the work of predicting elbow and shoulder joint kinematics from EMG signals [68]. Their approach used a time delayed artificial neural network (TDNN) to predict elbow position and velocity as well as shoulder position and velocity in its three DOF. Unlike prior work in predicting joint kinematics they tracked multiple degrees of freedom which included more complex motions. EMG channels were collected from the biceps, triceps, pectoralis major, and three deltoid channels. They also collected position data using a three dimensional motion tracking system. They trained their system with single joint movements where only one DOF was moving at once, reaching movements that involved moving the arm to different points in space, and complex movements that included circles, squares, and figure eights. They also performed the previously mentioned motions at different speeds and with different loads. The structure of the TDNN is shown in Figure 24. Key components of the TDNN design is the total delay and delay intervals of the inputs. Their optimal parameters were a total delay of 875ms with 125ms delay intervals. They recommend that the hidden layer have five to twenty neurons. They reported that their results maintained less than 20° of error. They were also successful at tracking movements of widely varying complexities including different speeds and different hand loads. Although their application was for functional neuromuscular stimulation, a similar approach could be applied for a myoelectric control system.
Rittenhouse examined a similar problem as the schemes already discussed. He investigated the ability of a feed forward neural network (FFNN) to map shoulder and elbow joint moments and angles to EMG data collected from eight muscles of the shoulder \[2\]. This approach can be considered as an inverse to the approaches already used. Instead of using EMG to predict joint kinematics, the joint kinematics and forces were used to predict EMG. The FFNN could then be used as biofeedback. The FFNN’s output of predicted EMG signals could be compared to measured EMG signals which could be used in therapy. They also proposed that this type of biofeedback could be used for a rehabilitation robot to improve the efficiency of the rehabilitation task. The muscles used in this study included three channels of deltoid EMG, infraspinatus, latissimus dorsi, subscapularis, supraspinatus, and the triceps brachii. Kinematic data was measured using a six camera motion analysis system and other sensors and transducers measured forces and moments. The motions used for the system included push-ups, chin-ups, and press-ups. Two different experiments were investigated in this work. They analyzed the FFNN’s ability to predict the EMG behavior of a subject whose data was not used to train the FFNN and the second experiment was to analyze the FFNN’s ability to predict EMG data from an activity not used for
training. The results show that the FFNN was able to predict EMG values for a subject whose data was not used to train the FFNN. The FFNN was also able to predict EMG data for an activity that was not used to train the FFNN.
Chapter 3: Methodology

3.1 Development of a Real Time Pattern Recognition Myoelectric Control Scheme

One of the primary goals of this work was to develop a real time pattern recognition myoelectric control scheme for a rehabilitation robot. The control scheme controlled three different motors on a robot and is the equivalent of controlling three DOF of a person’s upper limb. In a physical sense, the motors controlled were the elbow joint for flexion and extension, the wrist joint controlling supination and pronation of the forearm, and the hand which controlled grasping and releasing. One should note that the hand grasping and releasing corresponded to a gripper on the robot. The majority of myoelectric control schemes in literature appear to be grouped into two categories. The first category is where a controller is implemented for the elbow using the biceps and triceps. The second includes the wrist and hand using the forearm. After performing a literature review, the author was unable to find any myoelectric control schemes which combine these three DOF.

The work developed here can be considered a three stage process. The first step involved an offline analysis to develop the myoelectric control scheme. In the offline analysis, the primary focus was on feature extraction and the windowing scheme applied to the EMG signals. The second step was the real time implementation of the control scheme on a robot for healthy subjects. This allowed the myoelectric control scheme to be evaluated and tested. The features and windowing scheme that gave the optimal performance from the offline analysis were used in the real time development. In the third step, the same myoelectric control scheme developed and tested for healthy subjects was adapted and tested for a patient with Central Cord Syndrome (CCS). To the knowledge of the author, no attempts have ever been made to develop a myoelectric control scheme for a patient with CCS.

3.1.1 Offline Window and Feature Analysis

The offline window and feature analysis was performed in order to make a decision as to what techniques should be used for the real time myoelectric control system. Essential to the development of
the myoelectric control scheme is the placement of electrodes. Placing electrodes on muscles that have direct control over the desired type of movement in the rehabilitation robot is important. For instance, to control the elbow joint on the robot, one would want to place electrodes on the muscles that control a person’s elbow. Prior to the work completed for this thesis, many experiments were performed in the BLL to develop an understanding of EMG and the signal processing techniques used on EMG. One of the prior tests involved the optimal placement of electrodes for classification of forearm EMG. The optimal locations for forearm EMG channels were found to be the locations shown in Figure 25 as used by Englehart and Hargrove [18], [69]. The four channels were spread evenly around the circumference of the forearm approximately one third the length of the forearm below the elbow. Two other electrode configurations were also tested. These included a single electrode channel placed across the extensors and flexors of the forearm as was used by Yan [45]. He used the same classes for the forearm that this work uses and his electrode configuration is shown in Figure 19. The last configuration tested used a three channel configuration suggested by Hargrove [38] and also by Khushaba [39]. The locations are shown in Figure 18 and correspond to channels being located across the extensors/supinator, flexor carpi ulnaris, and flexor digitorum subliminus. Using classification accuracy as the performance indicator for the best electrode locations, the four electrode configuration stated above outperformed the other two configurations. The four electrode configuration used on the forearms was used to extract information about the wrist and hand DOF, but other EMG channels needed to be used to extract information about the elbow DOF. Much of the published work in myoelectric control has shown that electrodes on the biceps and triceps have the ability to discriminate elbow flexion and extension as shown by Hudgins [8]. The electrodes used in this work were dual electrodes spaced 1cm apart purchased from MVAP Medical Supplies and are displayed in Figures 25 and 26 [70]. The data acquisition device used in this work was the BioRadio from CleveMed [71]. According to recommended sampling frequencies suggested in the literature, a sampling frequency of 960Hz was used for all of the work presented [9].
Figure 25. Shown here is the optimal placement of electrodes for EMG classification of the forearm. There were four differential EMG channels spread evenly around the circumference of the forearm approximately one third the length of the forearm below the elbow. The reference electrode was placed on the elbow and can be seen on the left.

Figure 26. Shown here is the optimal placement of electrodes for EMG classification of the elbow. A biceps EMG channel is shown on the left and a triceps EMG channel is shown on the right.

It can be argued that the chosen classifier in a pattern recognition scheme is of little importance if the extracted features are not clustered in the feature space, and that the success of classification is dependent more on the features extracted than the classifier [38], [72]. This is why the offline analysis focused primarily on the features. The importance was placed on using features that created a well separated feature space rather than the testing of different types of classifiers. The classifier that was used in the offline testing as well as the real time implementation of the myoelectric control scheme was a linear discriminate analysis (LDA). Preliminary experiments performed prior to this work in the BLL showed that LDA classified better than the neural network. In the prior work, it was found that a neural network could be found that matched the classification accuracy of the LDA, but this required much
adjusting of the neural network parameters. In addition to classification accuracy, a LDA is much faster and easier to train relative to a neural network.

For the offline analysis, EMG data from ten healthy subjects, five males and five females, was collected. EMG data for each class was collected which included elbow flexion and extension, wrist pronation and supination, hand grasping and releasing, and rest. EMG channels were placed as shown previously in Figures 25 and 26. Ten trials of EMG data for each class were collected that lasted five seconds in length. Subjects were asked to produce isometric contractions that were fifty percent of their maximum voluntary contraction for each trial. The first five trials of data for each class were used to obtain the mean and covariance matrices to define the class PDFs for the LDA. The last five trials were used for testing.

In the offline testing, two different windowing schemes were tested. The first was the adjacent windowing scheme and window lengths of 50ms, 100ms, and 250ms were tested. The adjacent windowing scheme was explained in the background section and introduced by Hudgins [8]. The corresponding window sizes tested are also found in the literature [18], [47]. It was believed that the window of 250ms would be the optimal choice because it is known that features become less variable with an increasing window and a window of this size will meet the requirements of having a real time delay less than or equal to 300ms [17]. The overlapped windowing technique in conjunction with the majority voting technique was also tested [18]. The majority voting technique used a total window length of 250ms where the features extracted came from 50ms windows inside that 250ms window. The measurement used to distinguish the best window was classification accuracy.

Several types of features were analyzed. Previous experiments in the BLL leading up to this research had already tested many EMG features. One of the prior experiments used an EMG dataset collected from the biceps and triceps for elbow flexion and extension, wrist pronation and supination, and rest. Features considered for this EMG dataset included MAV, ZC, SSC, WL, AR coefficients, FFT, WT, and WPT energy. AR coefficients, FFT features, and the time domain features proved to be the most accurate for this dataset. Testing the different types of features in this preliminary work showed that the
EMG data which was misclassified for a portion of time would be misclassified independent of whatever feature was being used.

Based on preliminary experiments, the features chosen to be analyzed in the offline testing included MAV, ZC, SSC, WL, [8], [18] and AR coefficients [19], [22], [23]. The frequency and time-frequency features involve more processing time relative to time domain features and were not recommended to be used when considering many of the real time constraints [47]. Although the AR features can be considered frequency features, they will still be used because they were the most accurate in the preliminary tests. Additionally, the AR features were already used for a real time myoelectric controller for the elbow motor of the robotic arm. The time domain features being tested were tested individually and as a set. The AR features came from a fourth order model and were extracted using the Yule Walker Least Squares Method. The AR features were tested using one, two, three, and then all four coefficients as features. All of the features were tested using the windowing schemes explained previously. The majority voting scheme was only tested using the time domain set and all four of the AR features. Classification accuracy was used as the main measurement of feature performance. All of the offline testing was performed using Matlab Version 7.8.

3.1.2 Real Time Pattern Recognition Myoelectric Control Scheme for Healthy Subjects

As a result of the offline experiments described in the previous section, the real time myoelectric control scheme used an adjacent windowing technique of 250ms with AR features. The same data acquisition setup that was used in the offline experiments was repeated for the real time experiments. This included the locations of EMG electrodes as well as the BioRadio settings. The LDA classifier also remained the same. The ten subjects that were used in the offline experiments were also used for the real time experiments. The real time myoelectric control scheme was developed in Labview Version 8.5 in order to control the BLL’s robotic arm, displayed in Figure 27. A flowchart representing the Labview code that was developed is shown in Figure 28.
Figure 27. Displayed here is the 5 DOF robotic arm used in the Biomechatronic Learning Laboratory for the real time myoelectric control scheme.
Figure 28. Displayed here is a flowchart diagram of the Labview software.
In order to train the real time system, one five second trial of data for each class was collected and saved using the Labview environment. These five second trials of data were then processed offline using Matlab to extract the necessary mean and covariance matrices to define the class PDFs. Once the mean and covariance matrices were found, the system was ready to be run in real time to control the robotic arm. The subjects were then asked to practice controlling the robot arm’s elbow, wrist, and gripper joints. If the subject could not proficiently control any of the classes, the system was retrained with new data. Once the user felt confident in controlling each class, the real time system was tested quantitatively.

The real time test consisted of three different sets of randomized commands. Each set consisted of twenty-one commands that corresponded to three commands for each of the seven classes. The randomized set of commands is displayed below in Table 3 and filled in with the results of female subject one. Subjects were given a two minute rest period in between each set. Each test set was given to the subject via spoken commands from the author. The subject was asked to execute the classes as they were commanded. The author ran the tests and gave a new command every three to four seconds. If the subjects were able to hold the class commanded for the given three to four seconds, the command was considered to be a correct classification. If subjects were unable to hold the command for three to four seconds, the command was considered to be misclassified. When a misclassification occurred, the incorrect output(s) of the classifier were recorded. Some subjective analysis was needed by the author when deciding whether a command was classified correctly or not. One or two windows were allowed to be misclassified only during the switch of one commanded class to another, but no more and at no other time. This was a difference from the way classification accuracies were found for the offline analysis. In the offline analysis all of the data that was saved for a certain class was known which class it belonged to and therefore each of the windows could be objectively classified. In the real time system, the commands were given audibly and the classification output was written down by the author who was administering the randomized commands. Because the classification occurred in 250ms windows for the real time scheme, it was not feasible for the author to write down each and every classification. Rather the author judged whether the test subject could sustain the commanded class for an extended period of time.
Table 3. Displayed in this table are the three sets of randomized commands that each subject was given. The table has been filled in with the results from female subject 1.

3.1.3 Real Time Pattern Recognition Myoelectric Control Scheme for a CCS Patient

The myoelectric control scheme developed for healthy subjects which was presented in the previous section was applied to a subject with CCS. The author had seen no attempts at developing a myoelectric control scheme for a subject with CCS after a review of the literature. Because no prior work had been performed, there was an uncertainty as to whether or not the subject would have the ability to control all three DOF at once. Due to this uncertainty, three different one DOF controllers were developed to test individually prior to testing the more complex three DOF controller. A controller for elbow flexion and extension was created using only EMG input from the subject’s biceps and triceps. The other two controllers were for the wrist and the gripper. The wrist corresponded to forearm pronation and supination by the subject while the opening and closing of the gripper corresponded to the grasping and releasing of the subject’s hand. The wrist and gripper controllers used the four EMG channels from the forearm. Each single DOF controller had three classes which included a class for rest. The same methodology and theory
used for the full three DOF controller was used for the one DOF controllers. These similarities included
the same data acquisition, windowing technique applied, feature extraction, and classifier. The controllers
were trained and tested in the same manner as was completed for the healthy subjects. Five seconds of
data for each class were collected in order to train each controller. Testing of each controller consisted of
three sets of randomized commands. Each set consisted of three commands for each class which resulted
in nine total commands per set. Table 4 displays the randomized commands for each controller with the
results shown. Once the CCS subject showed that they were able to control each DOF individually, the
subject was tested using the full three DOF controller using the same randomized command set that the
healthy subjects were given. Minor adjustments needed to be made with the placement of EMG channels
for the wrist single DOF controller as well as the full three DOF controller. These adjustments will be
highlighted and discussed in Chapter 5. A photo of the CCS subject in the Biomechatronic Learning Lab
is shown in Figure 29 while he is controlling the robotic arm.

![Figure 29. Displayed here is the lab environment in the Biomechatronic Learning Lab while the CCS subject is controlling the robotic arm.](image-url)
3.2 Development of a Myoelectric Control Scheme Based on a Time Delayed Neural Network

Another goal of this work was to develop a myoelectric control scheme that incorporates dynamic and complex motions of the arm. This type of myoelectric control scheme would allow for a more natural type of movement to control a rehabilitation robot, unlike the repeatable isometric contractions needed for myoelectric control schemes based on pattern recognition. A myoelectric control scheme based on complex motions of the arm would also allow for the control of more than one DOF at once. The pattern recognition based myoelectric control schemes only allow one DOF to be controlled at once. The work

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy 100.00% 100.00% 100.00%

Total Accuracy 100.00%

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pronation</td>
<td>Pro/Sup</td>
<td></td>
<td></td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>1 Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>2 Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>Supination</td>
<td>Pro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>Pro/Sup</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy 88.89% 77.78% 88.89%

Total Accuracy 80.95%

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy 88.89% 100.00% 100.00%

Total Accuracy 95.24%

Table 4. Displayed in this table are the randomized commands for the three single DOF controllers for the subject with CCS.
presented in this section is the analysis of a myoelectric control scheme based on a TDNN. A myoelectric control scheme based on a TDNN would allow for movement in more than one DOF at once and is controlled using complex motions of the arm.

A large portion of this work was based on previous work completed by Au and Kirsch and was presented in the background section [68]. Their work used six EMG channels to predict joint kinematics for all three DOF in the shoulder as well as the elbow using a TDNN. They used EMG channels from the biceps, triceps, pectoralis, and three channels from the deltoid. Their optimal TDNN parameters for predicting joint angle position were a total delay of 875ms and an interval of delay of 125ms. Using a total delay of 875ms is not feasible for a myoelectric control scheme [17]. This work assessed the ability of the TDNN as a potential control architecture for a rehabilitation robotic system. To verify performance for this application, the time delay must be less than or equal to 300ms while maintaining accuracy of the TDNN’s ability to predict joint position [17]. Therefore it was the goal of this work to see if the total delay of the TDNN could be shortened without sacrificing the accuracy of the TDNN.

Rather than use Au and Kirsch’s exact approach, a simplified version of their experimental setup was implemented. Instead of using the full four DOF that they used, motion was constrained to the sagittal plane resulting in one DOF in the shoulder as well as one DOF in the elbow. Only four EMG channels were used in this work which is two less than the previous work. EMG data was collected from the biceps, triceps, pectoralis, and deltoid. The data was collected using the Upper Extremity Motion Capture System in the Biomechatronic Learning Laboratory [73]. The Upper Extremity Motion Capture System is displayed in Figure 30 and also displays the EMG locations. Data was obtained from five healthy subjects, 4 males and 1 female, ages 23 to 24 years of age. The first subject’s data was used to determine the optimal parameters for the TDNN. The rest of the data from the other subjects were then used to test the optimal TDNN parameters found using the results of the first subject.

In order for the TDNN to track complex motions and to make the system robust, several types of movement were collected. Single joint movement consisted of movement from rest over the full range of motion for the specified joint and back to rest. Rest was considered to be the arm located at the subject’s
side at full elbow extension. Full elbow extension was considered to be 180° and full elbow flexion ranged from 50° to 60° depending on the subject. Figure 31 displays the joint angle range of the elbow for single joint motion. The rest position for the shoulder was considered to be 0° and the full range of motion for the shoulder varied between 90° and 120°, depending on the subject. Figure 32 displays the joint angle range of the shoulder in the sagittal plane for single joint motion. Reaching motions were also collected that resulted in the movement of both the elbow and shoulder DOF at once. The reaching motions consisted of reaching towards different points in space from rest and then returning to rest. Figure 33 displays an example of reaching motion. Each type of motion was recorded for slow and fast repetitions. Slow movements lasted for approximately 3 to 4 seconds per repetition whereas fast movements lasted for 2 to 3 seconds. In order to train the TDNNs for variability, the exact times were not constrained. For each subject, twelve different trials of data were collected with each trial lasting 30 seconds in length. Six trials were reserved for training the TDNN, and six trials were used for testing. The six training and testing trials consisted of elbow, shoulder, and reaching motions for both fast and slow motions.

Figure 30. Displayed here is the Upper Extremity Motion Capture System that collected all of the data used for testing the TDNN as a myoelectric control scheme. Marked in the picture are the biceps, triceps, deltoid, and pectoralis EMG channels. The reference electrode is also displayed.
Figure 31. Displayed here is the full range of the elbow during single joint elbow motion. Full extension was considered to be 180° and full flexion was considered to be 50° to 60° depending on the subject.

Figure 32. Displayed here is the full range of the shoulder during single joint shoulder motion. At rest the shoulder was considered to be at 0° and at full range was considered to be 90° to 120° depending on the subject.
Much of the signal processing methods used in this algorithm were the same as the methods used by Au and Kirsch [68]. The raw EMG signals obtained from the muscles were rectified and filtered prior to being input to the TDNN. The lowpass filter was a fourth order Butterworth filter located at 4Hz. The filtering was implemented because the movements had no frequency content above this frequency. The algorithm is displayed in Figure 34. It should be noted that multiple inputs come from each EMG channel. Associated with a TDNN is an interval of delay represented by $\Delta t$ and a total delay which is $n\Delta t$ for $n$ delay intervals. For instance, if the total delay was 300ms, and the time interval was 50ms, then $n$ is equal to 6. Because the goal this work was to investigate the possibility of using the TDNN output as a myoelectric control input, different time delays were analyzed. The time delays tested were 300ms, 600ms, and 900ms. The different delay intervals tested were 50ms, 100ms, and 150ms. A single layer was used and the number of neurons tested was 10, 20, 30, and 40 neurons. The use of the TDNN was a two step process consisting of testing and training. As recommended by Au and Kirsch, the position data was normalized between 0 and 1. All of the neural network simulations were executed using Matlab’s neural network toolbox, Version 7.6. The neural networks created were feed-forward, back propagation networks. The “tansig transfer function” was used for the hidden layer and a “linear transfer function” was used for the output layer. The training was limited to a maximum of 250 iterations.
Figure 34. Displayed here is the block diagram of the algorithm for the TDNN.
Chapter 4: Results

4.1 Real Time Pattern Recognition Myoelectric Control Scheme Results

The results of the real time pattern recognition myoelectric control scheme are presented in three different sections. The first section shows the results of the offline window and feature analysis. Although this first section is not technically a presentation of real time results, these results contributed to the design and development of the real time system. The second section shows the results of the real time myoelectric control scheme for healthy subjects while the third section presents the real time myoelectric control scheme results for the subject with CCS.

4.1.1 Offline Window and Feature Analysis Results

Once all of the data was collected from the ten healthy subjects, the data was processed for all of the windowing schemes and features being tested. Classification accuracies were found for each of the possible windowing schemes and features to determine what techniques would be used in the real time myoelectric control scheme. For each subject, the data was compiled as shown in Table 5 for the first female subject. This allowed for an easy comparison of the results. Tables 6 through 14 follow afterwards and show the compiled offline results for the rest of the subjects. It is obvious from viewing the results that as the window size increased, so did the accuracy of classification. Because of the obvious trend of improved accuracy with a greater window size, several tables were compiled to summarize the most important results. Table 15 shows the results of using an adjacent windowing scheme of 250ms for all subjects. A comparison of the adjacent windowing scheme and the majority voting scheme is shown in Table 16 for all of the subjects. It should be noted that in Table 15 and 16 the column gives the feature(s) used and the feature(s) were extracted from each of the six EMG channels.
Table 5. Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 1.

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>50ms Windows</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>87.74%</td>
<td>88.19%</td>
<td>68.65%</td>
<td>56.07%</td>
<td>55.30%</td>
<td>39.17%</td>
<td>84.79%</td>
<td>69.74%</td>
</tr>
<tr>
<td>MAVS</td>
<td>89.43%</td>
<td>48.77%</td>
<td>35.06%</td>
<td>45.09%</td>
<td>40.32%</td>
<td>35.23%</td>
<td>62.75%</td>
<td>48.48%</td>
</tr>
<tr>
<td>ZC</td>
<td>93.58%</td>
<td>54.62%</td>
<td>57.53%</td>
<td>60.00%</td>
<td>85.94%</td>
<td>72.19%</td>
<td>63.09%</td>
<td>68.36%</td>
</tr>
<tr>
<td>SSC</td>
<td>29.62%</td>
<td>23.16%</td>
<td>46.40%</td>
<td>32.37%</td>
<td>30.41%</td>
<td>19.81%</td>
<td>53.36%</td>
<td>32.73%</td>
</tr>
<tr>
<td>WL</td>
<td>99.43%</td>
<td>88.77%</td>
<td>79.89%</td>
<td>91.33%</td>
<td>95.85%</td>
<td>87.37%</td>
<td>88.93%</td>
<td>90.29%</td>
</tr>
<tr>
<td>TD Set</td>
<td>94.91%</td>
<td>95.56%</td>
<td>91.46%</td>
<td>94.68%</td>
<td>92.17%</td>
<td>90.15%</td>
<td>91.61%</td>
<td>93.64%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>98.49%</td>
<td>62.22%</td>
<td>64.72%</td>
<td>77.34%</td>
<td>95.97%</td>
<td>86.33%</td>
<td>87.14%</td>
<td>78.14%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>99.43%</td>
<td>75.44%</td>
<td>82.02%</td>
<td>80.69%</td>
<td>96.89%</td>
<td>91.08%</td>
<td>92.84%</td>
<td>85.85%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>99.43%</td>
<td>82.22%</td>
<td>84.61%</td>
<td>83.93%</td>
<td>96.54%</td>
<td>91.89%</td>
<td>90.94%</td>
<td>88.50%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>96.42%</td>
<td>89.24%</td>
<td>86.63%</td>
<td>90.17%</td>
<td>95.16%</td>
<td>91.77%</td>
<td>92.62%</td>
<td>91.09%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>100ms Windows</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>86.04%</td>
<td>95.06%</td>
<td>73.36%</td>
<td>60.79%</td>
<td>65.82%</td>
<td>47.80%</td>
<td>87.87%</td>
<td>75.31%</td>
</tr>
<tr>
<td>MAVS</td>
<td>80.00%</td>
<td>35.06%</td>
<td>35.44%</td>
<td>33.64%</td>
<td>22.40%</td>
<td>28.54%</td>
<td>56.63%</td>
<td>38.06%</td>
</tr>
<tr>
<td>ZC</td>
<td>95.85%</td>
<td>65.18%</td>
<td>66.59%</td>
<td>74.25%</td>
<td>88.22%</td>
<td>86.31%</td>
<td>72.13%</td>
<td>76.51%</td>
</tr>
<tr>
<td>SSC</td>
<td>33.58%</td>
<td>25.41%</td>
<td>54.18%</td>
<td>40.37%</td>
<td>34.64%</td>
<td>29.70%</td>
<td>60.22%</td>
<td>38.11%</td>
</tr>
<tr>
<td>WL</td>
<td>99.25%</td>
<td>96.71%</td>
<td>85.55%</td>
<td>95.59%</td>
<td>97.92%</td>
<td>95.82%</td>
<td>93.26%</td>
<td>94.59%</td>
</tr>
<tr>
<td>TD Set</td>
<td>94.34%</td>
<td>99.06%</td>
<td>93.00%</td>
<td>96.06%</td>
<td>96.30%</td>
<td>93.97%</td>
<td>92.36%</td>
<td>95.84%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>99.62%</td>
<td>79.06%</td>
<td>78.56%</td>
<td>90.02%</td>
<td>98.85%</td>
<td>94.20%</td>
<td>97.75%</td>
<td>88.33%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>88.71%</td>
<td>91.65%</td>
<td>93.50%</td>
<td>98.85%</td>
<td>98.38%</td>
<td>98.65%</td>
<td>94.09%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>92.94%</td>
<td>94.36%</td>
<td>94.90%</td>
<td>98.38%</td>
<td>98.84%</td>
<td>94.83%</td>
<td>95.79%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>96.98%</td>
<td>97.41%</td>
<td>94.13%</td>
<td>97.22%</td>
<td>96.54%</td>
<td>97.22%</td>
<td>95.06%</td>
<td>96.39%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>250ms Windows</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>86.96%</td>
<td>98.82%</td>
<td>76.70%</td>
<td>64.71%</td>
<td>73.10%</td>
<td>64.71%</td>
<td>87.64%</td>
<td>79.31%</td>
</tr>
<tr>
<td>MAVS</td>
<td>64.13%</td>
<td>33.14%</td>
<td>21.59%</td>
<td>23.53%</td>
<td>13.45%</td>
<td>18.82%</td>
<td>44.94%</td>
<td>27.76%</td>
</tr>
<tr>
<td>ZC</td>
<td>97.83%</td>
<td>83.43%</td>
<td>71.02%</td>
<td>88.82%</td>
<td>91.23%</td>
<td>92.94%</td>
<td>80.90%</td>
<td>85.22%</td>
</tr>
<tr>
<td>SSC</td>
<td>22.83%</td>
<td>38.46%</td>
<td>69.32%</td>
<td>46.47%</td>
<td>46.20%</td>
<td>41.18%</td>
<td>78.65%</td>
<td>47.04%</td>
</tr>
<tr>
<td>WL</td>
<td>97.83%</td>
<td>99.41%</td>
<td>90.34%</td>
<td>98.24%</td>
<td>99.42%</td>
<td>98.82%</td>
<td>97.19%</td>
<td>96.92%</td>
</tr>
<tr>
<td>TD Set</td>
<td>80.43%</td>
<td>99.41%</td>
<td>97.73%</td>
<td>95.29%</td>
<td>97.66%</td>
<td>97.06%</td>
<td>93.82%</td>
<td>95.50%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>100.00%</td>
<td>95.86%</td>
<td>88.07%</td>
<td>96.47%</td>
<td>98.83%</td>
<td>97.65%</td>
<td>100.00%</td>
<td>95.37%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>97.63%</td>
<td>95.45%</td>
<td>98.24%</td>
<td>98.83%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>97.81%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>99.41%</td>
<td>96.02%</td>
<td>98.24%</td>
<td>98.25%</td>
<td>100.00%</td>
<td>95.51%</td>
<td>98.20%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>97.83%</td>
<td>100.00%</td>
<td>96.59%</td>
<td>99.41%</td>
<td>96.49%</td>
<td>99.41%</td>
<td>95.51%</td>
<td>98.07%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority Voting 50ms Windows and 250ms Total Delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TD Set</td>
<td>96.77%</td>
<td>98.12%</td>
<td>95.71%</td>
<td>97.54%</td>
<td>97.68%</td>
<td>97.56%</td>
<td>93.71%</td>
<td>96.89%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>97.72%</td>
<td>97.88%</td>
<td>95.03%</td>
<td>99.41%</td>
<td>97.33%</td>
<td>99.07%</td>
<td>95.39%</td>
<td>97.24%</td>
</tr>
</tbody>
</table>

Table 5. Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 1.
Table 6. Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 2.

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>50ms Windows</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>50.73%</td>
<td>72.55%</td>
<td>12.65%</td>
<td>34.50%</td>
<td>31.25%</td>
<td>42.66%</td>
<td>43.65%</td>
<td>39.43%</td>
</tr>
<tr>
<td>MAVS</td>
<td>34.55%</td>
<td>75.54%</td>
<td>6.15%</td>
<td>35.34%</td>
<td>38.22%</td>
<td>50.29%</td>
<td>17.06%</td>
<td>38.02%</td>
</tr>
<tr>
<td>ZC</td>
<td>80.73%</td>
<td>81.74%</td>
<td>29.23%</td>
<td>48.25%</td>
<td>28.61%</td>
<td>37.02%</td>
<td>24.71%</td>
<td>51.62%</td>
</tr>
<tr>
<td>SSC</td>
<td>29.45%</td>
<td>60.86%</td>
<td>10.21%</td>
<td>23.52%</td>
<td>43.27%</td>
<td>20.80%</td>
<td>44.94%</td>
<td>33.62%</td>
</tr>
<tr>
<td>WL</td>
<td>94.73%</td>
<td>93.56%</td>
<td>91.42%</td>
<td>68.28%</td>
<td>89.18%</td>
<td>86.72%</td>
<td>80.12%</td>
<td>86.96%</td>
</tr>
<tr>
<td>TD Set</td>
<td>81.82%</td>
<td>95.23%</td>
<td>79.93%</td>
<td>56.09%</td>
<td>89.78%</td>
<td>89.42%</td>
<td>74.59%</td>
<td>80.52%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>88.36%</td>
<td>91.05%</td>
<td>81.90%</td>
<td>44.03%</td>
<td>70.55%</td>
<td>68.16%</td>
<td>85.88%</td>
<td>74.33%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>94.55%</td>
<td>92.96%</td>
<td>86.08%</td>
<td>53.20%</td>
<td>85.22%</td>
<td>81.67%</td>
<td>87.18%</td>
<td>81.59%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>95.64%</td>
<td>93.44%</td>
<td>88.98%</td>
<td>62.85%</td>
<td>86.42%</td>
<td>85.08%</td>
<td>90.00%</td>
<td>84.79%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>94.36%</td>
<td>94.39%</td>
<td>88.28%</td>
<td>66.95%</td>
<td>88.94%</td>
<td>85.19%</td>
<td>88.71%</td>
<td>86.06%</td>
</tr>
<tr>
<td>100ms Windows</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>56.00%</td>
<td>73.62%</td>
<td>13.95%</td>
<td>31.96%</td>
<td>31.88%</td>
<td>44.10%</td>
<td>46.23%</td>
<td>40.28%</td>
</tr>
<tr>
<td>MAVS</td>
<td>36.73%</td>
<td>70.50%</td>
<td>4.42%</td>
<td>39.23%</td>
<td>32.37%</td>
<td>57.08%</td>
<td>19.81%</td>
<td>36.43%</td>
</tr>
<tr>
<td>ZC</td>
<td>81.09%</td>
<td>86.57%</td>
<td>36.74%</td>
<td>54.24%</td>
<td>35.99%</td>
<td>56.13%</td>
<td>28.30%</td>
<td>57.21%</td>
</tr>
<tr>
<td>SSC</td>
<td>30.91%</td>
<td>67.39%</td>
<td>18.14%</td>
<td>29.30%</td>
<td>51.95%</td>
<td>32.55%</td>
<td>53.30%</td>
<td>39.97%</td>
</tr>
<tr>
<td>WL</td>
<td>97.82%</td>
<td>93.76%</td>
<td>98.37%</td>
<td>76.03%</td>
<td>93.72%</td>
<td>94.81%</td>
<td>95.99%</td>
<td>91.59%</td>
</tr>
<tr>
<td>TD Set</td>
<td>78.18%</td>
<td>94.72%</td>
<td>93.02%</td>
<td>55.21%</td>
<td>96.38%</td>
<td>95.28%</td>
<td>87.50%</td>
<td>83.99%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>96.36%</td>
<td>94.48%</td>
<td>94.42%</td>
<td>63.44%</td>
<td>89.61%</td>
<td>91.98%</td>
<td>98.58%</td>
<td>87.12%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>98.55%</td>
<td>94.48%</td>
<td>97.67%</td>
<td>74.33%</td>
<td>96.62%</td>
<td>96.23%</td>
<td>98.35%</td>
<td>91.94%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>98.55%</td>
<td>96.40%</td>
<td>96.28%</td>
<td>77.97%</td>
<td>96.38%</td>
<td>96.70%</td>
<td>98.82%</td>
<td>92.77%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>95.64%</td>
<td>95.68%</td>
<td>96.05%</td>
<td>78.93%</td>
<td>96.62%</td>
<td>96.23%</td>
<td>96.93%</td>
<td>92.41%</td>
</tr>
<tr>
<td>250ms Windows</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>63.00%</td>
<td>79.39%</td>
<td>18.13%</td>
<td>27.44%</td>
<td>36.36%</td>
<td>48.52%</td>
<td>53.25%</td>
<td>43.14%</td>
</tr>
<tr>
<td>MAVS</td>
<td>39.00%</td>
<td>58.79%</td>
<td>6.43%</td>
<td>27.44%</td>
<td>29.09%</td>
<td>44.97%</td>
<td>34.91%</td>
<td>31.37%</td>
</tr>
<tr>
<td>ZC</td>
<td>77.00%</td>
<td>90.91%</td>
<td>50.29%</td>
<td>59.15%</td>
<td>53.94%</td>
<td>71.60%</td>
<td>33.73%</td>
<td>65.23%</td>
</tr>
<tr>
<td>SSC</td>
<td>28.00%</td>
<td>80.61%</td>
<td>40.35%</td>
<td>48.17%</td>
<td>67.27%</td>
<td>46.75%</td>
<td>75.74%</td>
<td>54.90%</td>
</tr>
<tr>
<td>WL</td>
<td>97.00%</td>
<td>93.94%</td>
<td>99.42%</td>
<td>76.22%</td>
<td>96.97%</td>
<td>96.45%</td>
<td>99.41%</td>
<td>92.42%</td>
</tr>
<tr>
<td>TD Set</td>
<td>63.00%</td>
<td>94.55%</td>
<td>99.42%</td>
<td>38.41%</td>
<td>96.97%</td>
<td>97.04%</td>
<td>98.82%</td>
<td>80.00%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>99.00%</td>
<td>94.55%</td>
<td>98.25%</td>
<td>85.37%</td>
<td>98.18%</td>
<td>97.63%</td>
<td>100.00%</td>
<td>94.77%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>96.36%</td>
<td>100.00%</td>
<td>89.02%</td>
<td>98.79%</td>
<td>98.82%</td>
<td>100.00%</td>
<td>96.60%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>97.58%</td>
<td>100.00%</td>
<td>92.07%</td>
<td>98.79%</td>
<td>98.82%</td>
<td>100.00%</td>
<td>97.52%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>99.00%</td>
<td>97.58%</td>
<td>100.00%</td>
<td>92.68%</td>
<td>98.79%</td>
<td>97.63%</td>
<td>98.82%</td>
<td>97.52%</td>
</tr>
<tr>
<td>Majority Voting 50ms Windows and 250ms Total Delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>88.00%</td>
<td>84.00%</td>
<td>71.07%</td>
<td>97.69%</td>
<td>93.34%</td>
<td>98.61%</td>
<td>86.15%</td>
<td></td>
</tr>
<tr>
<td>MAVS</td>
<td>94.96%</td>
<td>88.00%</td>
<td>71.07%</td>
<td>97.69%</td>
<td>93.34%</td>
<td>98.61%</td>
<td>86.15%</td>
<td></td>
</tr>
<tr>
<td>ZC</td>
<td>87.00%</td>
<td>87.00%</td>
<td>71.07%</td>
<td>97.69%</td>
<td>93.34%</td>
<td>98.61%</td>
<td>86.15%</td>
<td></td>
</tr>
<tr>
<td>SSC</td>
<td>98.17%</td>
<td>96.16%</td>
<td>97.09%</td>
<td>85.29%</td>
<td>97.94%</td>
<td>96.93%</td>
<td>98.35%</td>
<td>92.93%</td>
</tr>
</tbody>
</table>

Table 6. Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 2.
<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>86.47%</td>
<td>27.57%</td>
<td>61.45%</td>
<td>58.79%</td>
<td>38.49%</td>
<td>82.24%</td>
<td>54.52%</td>
<td>52.02%</td>
</tr>
<tr>
<td>MAVS</td>
<td>93.24%</td>
<td>34.70%</td>
<td>25.00%</td>
<td>34.58%</td>
<td>41.81%</td>
<td>61.31%</td>
<td>50.00%</td>
<td>42.17%</td>
</tr>
<tr>
<td>ZC</td>
<td>94.52%</td>
<td>62.97%</td>
<td>62.50%</td>
<td>64.73%</td>
<td>63.00%</td>
<td>46.49%</td>
<td>36.31%</td>
<td>67.58%</td>
</tr>
<tr>
<td>SSC</td>
<td>37.48%</td>
<td>43.93%</td>
<td>41.24%</td>
<td>43.54%</td>
<td>42.15%</td>
<td>16.97%</td>
<td>87.16%</td>
<td>41.99%</td>
</tr>
<tr>
<td>WL</td>
<td>97.26%</td>
<td>92.76%</td>
<td>91.24%</td>
<td>89.52%</td>
<td>87.86%</td>
<td>84.39%</td>
<td>99.27%</td>
<td>91.28%</td>
</tr>
<tr>
<td>TD Set</td>
<td>95.06%</td>
<td>96.26%</td>
<td>90.30%</td>
<td>85.10%</td>
<td>82.25%</td>
<td>85.18%</td>
<td>99.14%</td>
<td>89.35%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>92.32%</td>
<td>92.87%</td>
<td>75.93%</td>
<td>65.42%</td>
<td>71.59%</td>
<td>72.51%</td>
<td>89.49%</td>
<td>78.60%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>98.72%</td>
<td>95.91%</td>
<td>89.25%</td>
<td>89.87%</td>
<td>84.77%</td>
<td>86.88%</td>
<td>94.87%</td>
<td>91.13%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>98.54%</td>
<td>96.26%</td>
<td>89.02%</td>
<td>87.08%</td>
<td>84.42%</td>
<td>86.65%</td>
<td>94.13%</td>
<td>90.45%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>96.53%</td>
<td>96.14%</td>
<td>90.07%</td>
<td>88.47%</td>
<td>85.34%</td>
<td>86.31%</td>
<td>95.11%</td>
<td>90.88%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>86.86%</td>
<td>30.21%</td>
<td>68.38%</td>
<td>61.12%</td>
<td>38.39%</td>
<td>85.68%</td>
<td>58.48%</td>
<td>54.62%</td>
</tr>
<tr>
<td>MAVS</td>
<td>96.72%</td>
<td>19.67%</td>
<td>21.08%</td>
<td>22.72%</td>
<td>24.60%</td>
<td>49.09%</td>
<td>26.78%</td>
<td>32.31%</td>
</tr>
<tr>
<td>ZC</td>
<td>94.16%</td>
<td>70.26%</td>
<td>70.73%</td>
<td>69.09%</td>
<td>72.64%</td>
<td>56.82%</td>
<td>44.23%</td>
<td>73.92%</td>
</tr>
<tr>
<td>SSC</td>
<td>31.39%</td>
<td>54.10%</td>
<td>53.63%</td>
<td>52.46%</td>
<td>53.33%</td>
<td>28.18%</td>
<td>93.37%</td>
<td>50.35%</td>
</tr>
<tr>
<td>WL</td>
<td>98.54%</td>
<td>95.55%</td>
<td>96.25%</td>
<td>94.61%</td>
<td>88.74%</td>
<td>82.95%</td>
<td>99.26%</td>
<td>94.42%</td>
</tr>
<tr>
<td>TD Set</td>
<td>94.16%</td>
<td>95.78%</td>
<td>91.57%</td>
<td>90.63%</td>
<td>83.68%</td>
<td>84.32%</td>
<td>99.51%</td>
<td>90.90%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>96.72%</td>
<td>97.19%</td>
<td>89.70%</td>
<td>77.99%</td>
<td>84.83%</td>
<td>82.27%</td>
<td>95.82%</td>
<td>88.69%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>97.19%</td>
<td>95.55%</td>
<td>95.32%</td>
<td>92.87%</td>
<td>92.95%</td>
<td>99.02%</td>
<td>95.88%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>99.64%</td>
<td>97.19%</td>
<td>95.78%</td>
<td>92.51%</td>
<td>93.33%</td>
<td>91.59%</td>
<td>99.02%</td>
<td>95.38%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>97.08%</td>
<td>97.42%</td>
<td>96.02%</td>
<td>92.74%</td>
<td>93.33%</td>
<td>91.82%</td>
<td>98.77%</td>
<td>95.18%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>87.76%</td>
<td>34.12%</td>
<td>71.18%</td>
<td>61.40%</td>
<td>40.46%</td>
<td>83.91%</td>
<td>66.05%</td>
<td>56.27%</td>
</tr>
<tr>
<td>MAVS</td>
<td>90.82%</td>
<td>17.65%</td>
<td>16.47%</td>
<td>22.81%</td>
<td>27.75%</td>
<td>31.03%</td>
<td>28.40%</td>
<td>29.92%</td>
</tr>
<tr>
<td>ZC</td>
<td>90.82%</td>
<td>77.06%</td>
<td>76.47%</td>
<td>74.27%</td>
<td>81.50%</td>
<td>62.64%</td>
<td>54.94%</td>
<td>79.03%</td>
</tr>
<tr>
<td>SSC</td>
<td>16.33%</td>
<td>70.59%</td>
<td>64.71%</td>
<td>63.16%</td>
<td>77.46%</td>
<td>50.57%</td>
<td>98.15%</td>
<td>62.40%</td>
</tr>
<tr>
<td>WL</td>
<td>98.98%</td>
<td>95.88%</td>
<td>97.06%</td>
<td>95.91%</td>
<td>86.71%</td>
<td>83.33%</td>
<td>98.77%</td>
<td>94.50%</td>
</tr>
<tr>
<td>TD Set</td>
<td>63.27%</td>
<td>97.06%</td>
<td>92.94%</td>
<td>94.74%</td>
<td>80.35%</td>
<td>85.06%</td>
<td>100.00%</td>
<td>87.72%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>96.94%</td>
<td>97.65%</td>
<td>96.47%</td>
<td>89.47%</td>
<td>84.39%</td>
<td>92.53%</td>
<td>96.30%</td>
<td>92.58%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>97.06%</td>
<td>97.06%</td>
<td>95.91%</td>
<td>95.95%</td>
<td>94.83%</td>
<td>100.00%</td>
<td>96.93%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>97.06%</td>
<td>98.24%</td>
<td>96.49%</td>
<td>95.95%</td>
<td>93.68%</td>
<td>100.00%</td>
<td>97.31%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>95.92%</td>
<td>97.06%</td>
<td>98.24%</td>
<td>96.49%</td>
<td>95.95%</td>
<td>92.53%</td>
<td>99.38%</td>
<td>96.80%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>97.42%</td>
<td>98.00%</td>
<td>92.25%</td>
<td>88.98%</td>
<td>88.11%</td>
<td>88.07%</td>
<td>100.00%</td>
<td>92.29%</td>
</tr>
<tr>
<td>MAVS</td>
<td>97.79%</td>
<td>96.95%</td>
<td>96.01%</td>
<td>97.26%</td>
<td>95.30%</td>
<td>91.82%</td>
<td>99.63%</td>
<td>95.77%</td>
</tr>
</tbody>
</table>

Table 7. Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 3.
### Table 8

Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 4.

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>50ms Windows</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>92.24%</td>
<td>69.59%</td>
<td>50.95%</td>
<td>13.16%</td>
<td>42.47%</td>
<td>43.07%</td>
<td>92.67%</td>
<td>50.53%</td>
</tr>
<tr>
<td>MAVS</td>
<td>95.75%</td>
<td>58.01%</td>
<td>40.38%</td>
<td>13.74%</td>
<td>35.41%</td>
<td>18.98%</td>
<td>53.32%</td>
<td>44.87%</td>
</tr>
<tr>
<td>ZC</td>
<td>98.15%</td>
<td>66.78%</td>
<td>39.07%</td>
<td>34.64%</td>
<td>56.71%</td>
<td>37.49%</td>
<td>80.56%</td>
<td>55.97%</td>
</tr>
<tr>
<td>SSC</td>
<td>24.58%</td>
<td>51.11%</td>
<td>11.28%</td>
<td>37.64%</td>
<td>24.82%</td>
<td>9.20%</td>
<td>68.22%</td>
<td>30.40%</td>
</tr>
<tr>
<td>WL</td>
<td>99.08%</td>
<td>83.04%</td>
<td>64.85%</td>
<td>46.54%</td>
<td>84.35%</td>
<td>48.89%</td>
<td>96.97%</td>
<td>73.65%</td>
</tr>
<tr>
<td>TD Set</td>
<td>98.15%</td>
<td>85.96%</td>
<td>83.25%</td>
<td>45.38%</td>
<td>76.35%</td>
<td>58.32%</td>
<td>96.62%</td>
<td>76.10%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>99.26%</td>
<td>87.49%</td>
<td>55.94%</td>
<td>52.77%</td>
<td>58.08%</td>
<td>44.59%</td>
<td>96.86%</td>
<td>68.41%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>87.13%</td>
<td>61.88%</td>
<td>54.73%</td>
<td>73.90%</td>
<td>50.76%</td>
<td>97.90%</td>
<td>73.58%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>89.82%</td>
<td>64.13%</td>
<td>55.77%</td>
<td>75.06%</td>
<td>51.92%</td>
<td>98.14%</td>
<td>75.11%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>99.26%</td>
<td>88.89%</td>
<td>70.07%</td>
<td>56.12%</td>
<td>73.90%</td>
<td>60.65%</td>
<td>98.02%</td>
<td>75.89%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>100ms Windows</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>91.48%</td>
<td>75.59%</td>
<td>57.86%</td>
<td>13.89%</td>
<td>41.98%</td>
<td>47.66%</td>
<td>95.09%</td>
<td>53.25%</td>
</tr>
<tr>
<td>MAVS</td>
<td>89.26%</td>
<td>46.24%</td>
<td>21.43%</td>
<td>11.81%</td>
<td>18.63%</td>
<td>21.50%</td>
<td>50.00%</td>
<td>33.37%</td>
</tr>
<tr>
<td>ZC</td>
<td>98.15%</td>
<td>75.12%</td>
<td>47.38%</td>
<td>43.98%</td>
<td>68.40%</td>
<td>43.46%</td>
<td>85.98%</td>
<td>64.10%</td>
</tr>
<tr>
<td>SSC</td>
<td>26.67%</td>
<td>60.09%</td>
<td>12.62%</td>
<td>36.81%</td>
<td>35.61%</td>
<td>14.25%</td>
<td>81.07%</td>
<td>35.04%</td>
</tr>
<tr>
<td>WL</td>
<td>99.63%</td>
<td>89.67%</td>
<td>84.05%</td>
<td>60.65%</td>
<td>91.75%</td>
<td>62.15%</td>
<td>97.66%</td>
<td>83.92%</td>
</tr>
<tr>
<td>TD Set</td>
<td>96.30%</td>
<td>89.20%</td>
<td>90.00%</td>
<td>54.86%</td>
<td>79.48%</td>
<td>62.38%</td>
<td>96.26%</td>
<td>80.73%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>99.26%</td>
<td>91.55%</td>
<td>82.62%</td>
<td>70.60%</td>
<td>65.74%</td>
<td>45.09%</td>
<td>98.36%</td>
<td>80.51%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>92.96%</td>
<td>83.81%</td>
<td>78.01%</td>
<td>84.95%</td>
<td>56.07%</td>
<td>98.36%</td>
<td>86.97%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>93.90%</td>
<td>88.57%</td>
<td>76.39%</td>
<td>87.04%</td>
<td>57.48%</td>
<td>98.13%</td>
<td>88.28%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>99.63%</td>
<td>93.66%</td>
<td>89.29%</td>
<td>79.86%</td>
<td>84.49%</td>
<td>62.85%</td>
<td>98.13%</td>
<td>88.54%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>250ms Windows</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>93.75%</td>
<td>81.66%</td>
<td>68.26%</td>
<td>19.19%</td>
<td>53.29%</td>
<td>49.41%</td>
<td>96.47%</td>
<td>60.18%</td>
</tr>
<tr>
<td>MAVS</td>
<td>81.25%</td>
<td>37.87%</td>
<td>7.78%</td>
<td>17.44%</td>
<td>22.75%</td>
<td>23.53%</td>
<td>44.71%</td>
<td>28.92%</td>
</tr>
<tr>
<td>ZC</td>
<td>97.92%</td>
<td>78.11%</td>
<td>67.07%</td>
<td>58.14%</td>
<td>69.46%</td>
<td>56.47%</td>
<td>91.18%</td>
<td>71.85%</td>
</tr>
<tr>
<td>SSC</td>
<td>18.75%</td>
<td>74.56%</td>
<td>25.15%</td>
<td>37.21%</td>
<td>50.90%</td>
<td>27.65%</td>
<td>93.53%</td>
<td>43.45%</td>
</tr>
<tr>
<td>WL</td>
<td>98.96%</td>
<td>89.94%</td>
<td>95.81%</td>
<td>63.95%</td>
<td>94.61%</td>
<td>74.71%</td>
<td>98.24%</td>
<td>87.55%</td>
</tr>
<tr>
<td>TD Set</td>
<td>91.67%</td>
<td>97.04%</td>
<td>94.01%</td>
<td>61.63%</td>
<td>78.44%</td>
<td>66.47%</td>
<td>95.88%</td>
<td>83.79%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>100.00%</td>
<td>92.31%</td>
<td>97.01%</td>
<td>77.33%</td>
<td>75.58%</td>
<td>40.59%</td>
<td>98.82%</td>
<td>87.24%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>97.63%</td>
<td>97.01%</td>
<td>82.56%</td>
<td>91.28%</td>
<td>51.76%</td>
<td>98.82%</td>
<td>93.04%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>97.63%</td>
<td>97.60%</td>
<td>84.30%</td>
<td>91.28%</td>
<td>55.88%</td>
<td>98.24%</td>
<td>93.56%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>100.00%</td>
<td>97.63%</td>
<td>97.60%</td>
<td>86.63%</td>
<td>88.37%</td>
<td>60.00%</td>
<td>98.24%</td>
<td>93.43%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Majority Voting 50ms Windows and 250ms Total Delay</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TD Set</td>
<td>99.44%</td>
<td>91.54%</td>
<td>94.99%</td>
<td>55.09%</td>
<td>88.60%</td>
<td>73.92%</td>
<td>98.25%</td>
<td>82.23%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>100.00%</td>
<td>94.59%</td>
<td>88.19%</td>
<td>78.13%</td>
<td>90.89%</td>
<td>74.50%</td>
<td>98.36%</td>
<td>86.20%</td>
</tr>
</tbody>
</table>

Table 8. Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 4.
<table>
<thead>
<tr>
<th>Female Subject 5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>50ms Windows</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td><strong>EMG Features</strong></td>
</tr>
<tr>
<td>MAV</td>
</tr>
<tr>
<td>MAVS</td>
</tr>
<tr>
<td>ZC</td>
</tr>
<tr>
<td>SSC</td>
</tr>
<tr>
<td>WL</td>
</tr>
<tr>
<td>TD Set</td>
</tr>
<tr>
<td>1 AR Feature</td>
</tr>
<tr>
<td>2 AR Features</td>
</tr>
<tr>
<td>3 AR Features</td>
</tr>
<tr>
<td>4 AR Features</td>
</tr>
</tbody>
</table>

| **100ms Windows** |
| | MAV | MAVS | ZC | SSC | WL | TD Set | 1 AR Feature | 2 AR Features | 3 AR Features | 4 AR Features |
| | MAV | MAVS | ZC | SSC | WL | TD Set | 1 AR Feature | 2 AR Features | 3 AR Features | 4 AR Features |
| **EMG Features** | Rest | Grasping | Releasing | Pronation | Supination | Extension | Flexion | Overall |
| MAV | 84.67% | 71.01% | 70.82% | 72.71% | 36.32% | 88.43% | 72.25% | 66.21% |
| MAVS | 68.98% | 19.32% | 23.29% | 19.32% | 13.04% | 45.60% | 35.17% | 26.02% |
| ZC | 3.65% | 72.46% | 80.24% | 89.61% | 40.15% | 93.52% | 84.69% | 61.47% |
| SSC | 20.44% | 53.14% | 37.88% | 46.62% | 47.06% | 47.92% | 41.87% | 42.44% |
| WL | 10.22% | 89.37% | 85.18% | 96.38% | 74.17% | 98.84% | 91.87% | 75.55% |
| TD Set | 3.28% | 98.07% | 94.82% | 97.10% | 73.91% | 99.54% | 95.93% | 78.68% |
| 1 AR Feature | 2.92% | 94.69% | 84.24% | 79.95% | 45.41% | 97.92% | 96.41% | 65.79% |
| 2 AR Features | 4.74% | 97.83% | 94.82% | 94.20% | 82.61% | 99.54% | 96.89% | 80.01% |
| 3 AR Features | 9.12% | 98.55% | 95.53% | 93.72% | 83.82% | 99.77% | 96.89% | 81.09% |
| 4 AR Features | 3.28% | 97.34% | 96.24% | 93.48% | 80.43% | 99.07% | 98.33% | 79.39% |

| **250ms Windows** |
| | MAV | MAVS | ZC | SSC | WL | TD Set | 1 AR Feature | 2 AR Features | 3 AR Features | 4 AR Features |
| | MAV | MAVS | ZC | SSC | WL | TD Set | 1 AR Feature | 2 AR Features | 3 AR Features | 4 AR Features |
| **EMG Features** | Rest | Grasping | Releasing | Pronation | Supination | Extension | Flexion | Overall |
| MAV | 89.80% | 81.10% | 73.96% | 76.83% | 42.21% | 92.40% | 75.30% | 71.70% |
| MAVS | 26.53% | 17.68% | 20.71% | 33.54% | 12.34% | 29.82% | 33.13% | 21.90% |
| ZC | 57.14% | 81.71% | 80.47% | 92.68% | 43.51% | 92.40% | 89.16% | 72.76% |
| SSC | 12.24% | 70.12% | 57.99% | 59.76% | 66.88% | 70.76% | 54.22% | 56.88% |
| WL | 25.51% | 90.24% | 93.49% | 96.34% | 82.47% | 100.00% | 96.39% | 82.24% |
| TD Set | 42.86% | 100.00% | 96.45% | 97.56% | 82.47% | 100.00% | 98.80% | 87.58% |
| 1 AR Feature | 51.02% | 97.56% | 90.53% | 80.49% | 64.02% | 99.42% | 98.80% | 79.05% |
| 2 AR Features | 47.96% | 98.78% | 97.04% | 97.56% | 88.41% | 99.42% | 97.59% | 89.33% |
| 3 AR Features | 89.80% | 99.39% | 97.04% | 98.78% | 90.85% | 98.25% | 96.39% | 95.65% |
| 4 AR Features | 93.88% | 98.17% | 97.63% | 99.39% | 91.46% | 100.00% | 96.39% | 96.31% |

| Majority Voting 50ms Windows and 250ms Total Delay |
| | MAV | MAVS | ZC | SSC | WL | TD Set | 1 AR Feature | 2 AR Features | 3 AR Features | 4 AR Features |
| | MAV | MAVS | ZC | SSC | WL | TD Set | 1 AR Feature | 2 AR Features | 3 AR Features | 4 AR Features |
| **EMG Features** | Rest | Grasping | Releasing | Pronation | Supination | Extension | Flexion | Overall |
| MAV | 7.55% | 99.40% | 97.53% | 99.02% | 84.59% | 99.77% | 96.52% | 82.00% |
| MAVS | 3.87% | 98.31% | 97.41% | 97.78% | 76.43% | 100.00% | 96.64% | 79.19% |

Table 9. Displayed in this table are the classification accuracies of the offline window and feature analysis results for female subject 5.
Table 10. Displayed in this table are the classification accuracies of the offline window and feature analysis results for male subject 1.

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>84.45%</td>
<td>60.63%</td>
<td>71.91%</td>
<td>38.41%</td>
<td>20.41%</td>
<td>66.94%</td>
<td>35.20%</td>
<td>53.22%</td>
</tr>
<tr>
<td>MAVS</td>
<td>91.32%</td>
<td>50.85%</td>
<td>69.23%</td>
<td>34.43%</td>
<td>9.07%</td>
<td>26.03%</td>
<td>34.27%</td>
<td>48.29%</td>
</tr>
<tr>
<td>ZC</td>
<td>90.60%</td>
<td>84.90%</td>
<td>81.05%</td>
<td>56.79%</td>
<td>52.74%</td>
<td>75.85%</td>
<td>54.50%</td>
<td>71.99%</td>
</tr>
<tr>
<td>SSC</td>
<td>20.98%</td>
<td>34.90%</td>
<td>29.77%</td>
<td>37.59%</td>
<td>16.11%</td>
<td>22.86%</td>
<td>17.31%</td>
<td>28.41%</td>
</tr>
<tr>
<td>WL</td>
<td>94.58%</td>
<td>93.84%</td>
<td>90.86%</td>
<td>90.52%</td>
<td>78.28%</td>
<td>92.15%</td>
<td>87.60%</td>
<td>89.27%</td>
</tr>
<tr>
<td>TD Set</td>
<td>88.25%</td>
<td>94.20%</td>
<td>85.28%</td>
<td>94.96%</td>
<td>77.21%</td>
<td>93.08%</td>
<td>90.88%</td>
<td>87.93%</td>
</tr>
</tbody>
</table>

100ms Windows

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>85.20%</td>
<td>59.66%</td>
<td>73.60%</td>
<td>32.71%</td>
<td>24.40%</td>
<td>69.41%</td>
<td>37.24%</td>
<td>53.15%</td>
</tr>
<tr>
<td>MAVS</td>
<td>85.56%</td>
<td>45.65%</td>
<td>54.59%</td>
<td>21.65%</td>
<td>2.15%</td>
<td>23.29%</td>
<td>16.63%</td>
<td>38.92%</td>
</tr>
<tr>
<td>ZC</td>
<td>88.45%</td>
<td>85.75%</td>
<td>83.22%</td>
<td>65.18%</td>
<td>64.59%</td>
<td>80.71%</td>
<td>65.34%</td>
<td>76.68%</td>
</tr>
<tr>
<td>SSC</td>
<td>18.41%</td>
<td>36.71%</td>
<td>37.14%</td>
<td>46.82%</td>
<td>21.29%</td>
<td>22.59%</td>
<td>24.59%</td>
<td>33.17%</td>
</tr>
<tr>
<td>WL</td>
<td>98.19%</td>
<td>94.93%</td>
<td>91.72%</td>
<td>97.18%</td>
<td>80.86%</td>
<td>94.35%</td>
<td>92.97%</td>
<td>92.18%</td>
</tr>
<tr>
<td>TD Set</td>
<td>83.75%</td>
<td>94.20%</td>
<td>84.12%</td>
<td>98.82%</td>
<td>77.51%</td>
<td>95.76%</td>
<td>94.61%</td>
<td>87.94%</td>
</tr>
</tbody>
</table>

1 AR Feature

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>97.83%</td>
<td>96.38%</td>
<td>90.16%</td>
<td>93.88%</td>
<td>82.12%</td>
<td>90.35%</td>
<td>91.80%</td>
<td>91.60%</td>
</tr>
<tr>
<td>MAVS</td>
<td>97.47%</td>
<td>96.38%</td>
<td>91.72%</td>
<td>98.12%</td>
<td>88.00%</td>
<td>92.00%</td>
<td>100.00%</td>
<td>94.06%</td>
</tr>
<tr>
<td>ZC</td>
<td>98.19%</td>
<td>96.62%</td>
<td>92.39%</td>
<td>96.24%</td>
<td>92.47%</td>
<td>93.88%</td>
<td>100.00%</td>
<td>94.92%</td>
</tr>
<tr>
<td>SSC</td>
<td>91.70%</td>
<td>94.93%</td>
<td>90.60%</td>
<td>98.35%</td>
<td>90.82%</td>
<td>93.18%</td>
<td>99.06%</td>
<td>93.36%</td>
</tr>
</tbody>
</table>

3 AR Features

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>84.16%</td>
<td>58.54%</td>
<td>73.45%</td>
<td>24.85%</td>
<td>32.12%</td>
<td>70.24%</td>
<td>42.01%</td>
<td>52.32%</td>
</tr>
<tr>
<td>MAVS</td>
<td>58.42%</td>
<td>25.00%</td>
<td>39.55%</td>
<td>20.71%</td>
<td>1.82%</td>
<td>19.64%</td>
<td>14.79%</td>
<td>26.80%</td>
</tr>
<tr>
<td>ZC</td>
<td>89.11%</td>
<td>84.76%</td>
<td>81.92%</td>
<td>72.78%</td>
<td>69.09%</td>
<td>85.12%</td>
<td>71.60%</td>
<td>78.74%</td>
</tr>
<tr>
<td>SSC</td>
<td>4.95%</td>
<td>46.95%</td>
<td>45.76%</td>
<td>50.30%</td>
<td>34.55%</td>
<td>36.90%</td>
<td>46.15%</td>
<td>39.30%</td>
</tr>
<tr>
<td>WL</td>
<td>98.02%</td>
<td>94.51%</td>
<td>92.09%</td>
<td>99.41%</td>
<td>79.39%</td>
<td>95.24%</td>
<td>93.49%</td>
<td>92.27%</td>
</tr>
<tr>
<td>TD Set</td>
<td>70.30%</td>
<td>93.29%</td>
<td>82.49%</td>
<td>100.00%</td>
<td>72.73%</td>
<td>95.24%</td>
<td>98.82%</td>
<td>84.92%</td>
</tr>
</tbody>
</table>

250ms Windows

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>97.03%</td>
<td>96.95%</td>
<td>93.22%</td>
<td>97.04%</td>
<td>91.12%</td>
<td>92.26%</td>
<td>97.04%</td>
<td>94.87%</td>
</tr>
<tr>
<td>MAVS</td>
<td>99.01%</td>
<td>97.56%</td>
<td>93.22%</td>
<td>97.04%</td>
<td>94.08%</td>
<td>92.26%</td>
<td>100.00%</td>
<td>95.90%</td>
</tr>
<tr>
<td>ZC</td>
<td>100.00%</td>
<td>97.56%</td>
<td>94.35%</td>
<td>94.08%</td>
<td>95.86%</td>
<td>93.45%</td>
<td>100.00%</td>
<td>96.03%</td>
</tr>
<tr>
<td>SSC</td>
<td>92.08%</td>
<td>95.73%</td>
<td>91.53%</td>
<td>98.82%</td>
<td>95.86%</td>
<td>94.05%</td>
<td>100.00%</td>
<td>95.00%</td>
</tr>
</tbody>
</table>

Majority Voting 50ms Windows and 250ms Total Delay

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>90.53%</td>
<td>94.66%</td>
<td>87.35%</td>
<td>99.06%</td>
<td>90.28%</td>
<td>96.00%</td>
<td>97.06%</td>
<td>91.27%</td>
</tr>
<tr>
<td>MAVS</td>
<td>92.35%</td>
<td>95.39%</td>
<td>91.94%</td>
<td>99.29%</td>
<td>92.77%</td>
<td>93.99%</td>
<td>100.00%</td>
<td>93.97%</td>
</tr>
<tr>
<td>EMG Features</td>
<td>Rest</td>
<td>Grasping</td>
<td>Releasing</td>
<td>Pronation</td>
<td>Supination</td>
<td>Extension</td>
<td>Flexion</td>
<td>Overall</td>
</tr>
<tr>
<td>--------------</td>
<td>------</td>
<td>----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>MAV</td>
<td>46.58%</td>
<td>18.01%</td>
<td>2.23%</td>
<td>11.07%</td>
<td>36.47%</td>
<td>9.00%</td>
<td>66.36%</td>
<td>21.13%</td>
</tr>
<tr>
<td>MAVS</td>
<td>56.65%</td>
<td>16.65%</td>
<td>2.70%</td>
<td>10.21%</td>
<td>22.77%</td>
<td>5.01%</td>
<td>37.40%</td>
<td>19.25%</td>
</tr>
<tr>
<td>ZC</td>
<td>73.20%</td>
<td>20.72%</td>
<td>35.72%</td>
<td>40.71%</td>
<td>14.54%</td>
<td>10.14%</td>
<td>37.17%</td>
<td>34.17%</td>
</tr>
<tr>
<td>SSC</td>
<td>30.22%</td>
<td>25.82%</td>
<td>19.86%</td>
<td>13.16%</td>
<td>23.96%</td>
<td>28.82%</td>
<td>62.49%</td>
<td>22.15%</td>
</tr>
<tr>
<td>WL</td>
<td>90.65%</td>
<td>27.52%</td>
<td>49.82%</td>
<td>75.77%</td>
<td>51.13%</td>
<td>33.03%</td>
<td>88.26%</td>
<td>56.22%</td>
</tr>
<tr>
<td>TD Set</td>
<td>58.63%</td>
<td>41.45%</td>
<td>6.58%</td>
<td>72.20%</td>
<td>76.64%</td>
<td>34.51%</td>
<td>92.59%</td>
<td>50.18%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>77.88%</td>
<td>30.58%</td>
<td>51.12%</td>
<td>75.40%</td>
<td>46.36%</td>
<td>34.97%</td>
<td>98.52%</td>
<td>54.29%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>87.05%</td>
<td>36.01%</td>
<td>66.27%</td>
<td>82.16%</td>
<td>55.66%</td>
<td>34.74%</td>
<td>96.92%</td>
<td>63.44%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>92.81%</td>
<td>38.05%</td>
<td>67.33%</td>
<td>83.89%</td>
<td>57.33%</td>
<td>35.31%</td>
<td>95.67%</td>
<td>65.65%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>85.79%</td>
<td>40.09%</td>
<td>66.98%</td>
<td>78.23%</td>
<td>51.13%</td>
<td>31.78%</td>
<td>95.55%</td>
<td>62.56%</td>
</tr>
</tbody>
</table>

Table 11. Displayed in this table are the classification accuracies of the offline window and feature analysis results for male subject 2.
Table 12. Displayed in this table are the classification accuracies of the offline window and feature analysis results for male subject 3.

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>68.38%</td>
<td>42.57%</td>
<td>60.56%</td>
<td>24.02%</td>
<td>37.24%</td>
<td>54.61%</td>
<td>16.65%</td>
<td>45.10%</td>
</tr>
<tr>
<td>MAVS</td>
<td>81.62%</td>
<td>34.89%</td>
<td>49.53%</td>
<td>10.12%</td>
<td>27.11%</td>
<td>29.46%</td>
<td>10.75%</td>
<td>37.79%</td>
</tr>
<tr>
<td>ZC</td>
<td>89.15%</td>
<td>59.11%</td>
<td>62.21%</td>
<td>40.12%</td>
<td>14.04%</td>
<td>70.42%</td>
<td>30.06%</td>
<td>50.45%</td>
</tr>
<tr>
<td>SSC</td>
<td>25.55%</td>
<td>55.52%</td>
<td>32.04%</td>
<td>22.93%</td>
<td>35.16%</td>
<td>33.53%</td>
<td>27.17%</td>
<td>34.92%</td>
</tr>
<tr>
<td>WL</td>
<td>89.89%</td>
<td>97.12%</td>
<td>84.51%</td>
<td>80.73%</td>
<td>76.31%</td>
<td>91.86%</td>
<td>90.75%</td>
<td>85.45%</td>
</tr>
<tr>
<td>TD Set</td>
<td>87.13%</td>
<td>91.49%</td>
<td>85.56%</td>
<td>91.10%</td>
<td>78.75%</td>
<td>87.66%</td>
<td>82.89%</td>
<td>86.79%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>93.20%</td>
<td>82.37%</td>
<td>59.51%</td>
<td>78.17%</td>
<td>54.02%</td>
<td>73.53%</td>
<td>88.21%</td>
<td>71.96%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>94.67%</td>
<td>90.53%</td>
<td>65.49%</td>
<td>83.29%</td>
<td>82.32%</td>
<td>88.26%</td>
<td>92.95%</td>
<td>82.33%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>97.61%</td>
<td>89.45%</td>
<td>72.77%</td>
<td>86.59%</td>
<td>83.54%</td>
<td>88.86%</td>
<td>95.61%</td>
<td>85.06%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>97.43%</td>
<td>89.81%</td>
<td>68.31%</td>
<td>89.02%</td>
<td>81.22%</td>
<td>88.74%</td>
<td>95.26%</td>
<td>84.16%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>70.22%</td>
<td>43.86%</td>
<td>61.65%</td>
<td>24.51%</td>
<td>43.14%</td>
<td>56.49%</td>
<td>16.94%</td>
<td>47.25%</td>
</tr>
<tr>
<td>MAVS</td>
<td>75.37%</td>
<td>28.43%</td>
<td>29.65%</td>
<td>10.54%</td>
<td>22.55%</td>
<td>30.53%</td>
<td>9.51%</td>
<td>30.29%</td>
</tr>
<tr>
<td>ZC</td>
<td>90.07%</td>
<td>58.55%</td>
<td>66.82%</td>
<td>44.36%</td>
<td>22.06%</td>
<td>71.39%</td>
<td>37.59%</td>
<td>54.10%</td>
</tr>
<tr>
<td>SSC</td>
<td>22.79%</td>
<td>56.87%</td>
<td>40.24%</td>
<td>29.66%</td>
<td>43.87%</td>
<td>44.47%</td>
<td>41.07%</td>
<td>39.89%</td>
</tr>
<tr>
<td>WL</td>
<td>93.01%</td>
<td>99.04%</td>
<td>86.59%</td>
<td>88.24%</td>
<td>88.73%</td>
<td>94.23%</td>
<td>95.36%</td>
<td>90.98%</td>
</tr>
<tr>
<td>TD Set</td>
<td>86.76%</td>
<td>93.73%</td>
<td>85.41%</td>
<td>97.06%</td>
<td>85.05%</td>
<td>89.42%</td>
<td>93.50%</td>
<td>89.78%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>98.53%</td>
<td>94.70%</td>
<td>69.18%</td>
<td>87.01%</td>
<td>61.52%</td>
<td>81.49%</td>
<td>95.36%</td>
<td>80.96%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>98.90%</td>
<td>98.31%</td>
<td>56.71%</td>
<td>91.18%</td>
<td>95.59%</td>
<td>93.03%</td>
<td>98.14%</td>
<td>87.14%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>99.63%</td>
<td>96.39%</td>
<td>70.82%</td>
<td>95.59%</td>
<td>94.61%</td>
<td>93.51%</td>
<td>98.84%</td>
<td>90.66%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>98.16%</td>
<td>92.77%</td>
<td>68.00%</td>
<td>95.83%</td>
<td>93.63%</td>
<td>91.83%</td>
<td>99.07%</td>
<td>88.90%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>77.32%</td>
<td>46.95%</td>
<td>60.95%</td>
<td>25.93%</td>
<td>41.36%</td>
<td>60.00%</td>
<td>17.54%</td>
<td>48.28%</td>
</tr>
<tr>
<td>MAVS</td>
<td>19.59%</td>
<td>14.02%</td>
<td>54.44%</td>
<td>11.11%</td>
<td>20.99%</td>
<td>19.39%</td>
<td>16.37%</td>
<td>24.67%</td>
</tr>
<tr>
<td>ZC</td>
<td>87.63%</td>
<td>60.37%</td>
<td>70.41%</td>
<td>45.68%</td>
<td>35.80%</td>
<td>71.52%</td>
<td>43.86%</td>
<td>57.69%</td>
</tr>
<tr>
<td>SSC</td>
<td>36.08%</td>
<td>72.56%</td>
<td>55.62%</td>
<td>50.00%</td>
<td>54.32%</td>
<td>61.82%</td>
<td>61.99%</td>
<td>55.31%</td>
</tr>
<tr>
<td>WL</td>
<td>95.88%</td>
<td>99.39%</td>
<td>87.57%</td>
<td>83.95%</td>
<td>96.30%</td>
<td>93.94%</td>
<td>94.74%</td>
<td>92.31%</td>
</tr>
<tr>
<td>TD Set</td>
<td>82.47%</td>
<td>92.68%</td>
<td>89.35%</td>
<td>98.15%</td>
<td>90.12%</td>
<td>91.52%</td>
<td>99.42%</td>
<td>91.25%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>100.00%</td>
<td>97.56%</td>
<td>69.82%</td>
<td>94.44%</td>
<td>86.42%</td>
<td>92.12%</td>
<td>99.42%</td>
<td>88.59%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>99.39%</td>
<td>40.24%</td>
<td>97.53%</td>
<td>99.38%</td>
<td>96.36%</td>
<td>100.00%</td>
<td>85.81%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>98.78%</td>
<td>65.09%</td>
<td>96.30%</td>
<td>99.38%</td>
<td>96.36%</td>
<td>100.00%</td>
<td>90.98%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>97.94%</td>
<td>92.68%</td>
<td>67.46%</td>
<td>96.91%</td>
<td>98.77%</td>
<td>91.52%</td>
<td>100.00%</td>
<td>89.92%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAVS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TD Set</td>
<td>95.19%</td>
<td>92.89%</td>
<td>91.04%</td>
<td>99.38%</td>
<td>87.75%</td>
<td>92.54%</td>
<td>91.52%</td>
<td>92.57%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>99.44%</td>
<td>94.58%</td>
<td>75.47%</td>
<td>96.19%</td>
<td>94.23%</td>
<td>92.30%</td>
<td>99.65%</td>
<td>91.22%</td>
</tr>
</tbody>
</table>
Table 13. Displayed in this table are the classification accuracies of the offline window and feature analysis results for male subject 4.

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>87.74%</td>
<td>88.19%</td>
<td>68.65%</td>
<td>56.07%</td>
<td>55.30%</td>
<td>39.17%</td>
<td>84.79%</td>
<td>69.74%</td>
</tr>
<tr>
<td>MAVS</td>
<td>89.43%</td>
<td>48.77%</td>
<td>35.06%</td>
<td>35.09%</td>
<td>40.32%</td>
<td>35.23%</td>
<td>62.75%</td>
<td>48.48%</td>
</tr>
<tr>
<td>ZC</td>
<td>93.58%</td>
<td>54.62%</td>
<td>57.53%</td>
<td>60.00%</td>
<td>85.94%</td>
<td>72.19%</td>
<td>63.09%</td>
<td>68.36%</td>
</tr>
<tr>
<td>SSC</td>
<td>29.62%</td>
<td>23.16%</td>
<td>46.40%</td>
<td>32.37%</td>
<td>30.41%</td>
<td>19.81%</td>
<td>53.36%</td>
<td>32.73%</td>
</tr>
<tr>
<td>WL</td>
<td>99.43%</td>
<td>88.77%</td>
<td>79.89%</td>
<td>91.33%</td>
<td>95.85%</td>
<td>87.37%</td>
<td>88.93%</td>
<td>90.29%</td>
</tr>
<tr>
<td>TD Set</td>
<td>94.91%</td>
<td>95.56%</td>
<td>91.46%</td>
<td>94.68%</td>
<td>92.17%</td>
<td>90.15%</td>
<td>91.61%</td>
<td>93.64%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>97.96%</td>
<td>93.84%</td>
<td>87.09%</td>
<td>88.56%</td>
<td>88.10%</td>
<td>75.03%</td>
<td>100.00%</td>
<td>90.49%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>99.63%</td>
<td>98.49%</td>
<td>90.38%</td>
<td>90.62%</td>
<td>95.19%</td>
<td>92.30%</td>
<td>100.00%</td>
<td>94.47%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>98.87%</td>
<td>90.02%</td>
<td>90.39%</td>
<td>96.11%</td>
<td>93.93%</td>
<td>100.00%</td>
<td>94.61%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>98.70%</td>
<td>99.37%</td>
<td>87.44%</td>
<td>89.47%</td>
<td>96.22%</td>
<td>92.30%</td>
<td>100.00%</td>
<td>93.80%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>98.88%</td>
<td>99.75%</td>
<td>96.94%</td>
<td>48.62%</td>
<td>74.77%</td>
<td>94.38%</td>
<td>99.75%</td>
<td>82.11%</td>
</tr>
<tr>
<td>MAVS</td>
<td>94.05%</td>
<td>68.18%</td>
<td>59.76%</td>
<td>17.20%</td>
<td>50.69%</td>
<td>63.93%</td>
<td>25.99%</td>
<td>54.69%</td>
</tr>
<tr>
<td>ZC</td>
<td>95.54%</td>
<td>99.49%</td>
<td>90.35%</td>
<td>81.19%</td>
<td>92.20%</td>
<td>85.25%</td>
<td>99.26%</td>
<td>91.28%</td>
</tr>
<tr>
<td>SSC</td>
<td>47.58%</td>
<td>42.42%</td>
<td>48.71%</td>
<td>49.08%</td>
<td>44.50%</td>
<td>40.05%</td>
<td>85.64%</td>
<td>46.43%</td>
</tr>
<tr>
<td>WL</td>
<td>99.63%</td>
<td>99.75%</td>
<td>99.06%</td>
<td>94.04%</td>
<td>96.33%</td>
<td>94.15%</td>
<td>100.00%</td>
<td>97.55%</td>
</tr>
<tr>
<td>TD Set</td>
<td>97.40%</td>
<td>100.00%</td>
<td>99.53%</td>
<td>92.89%</td>
<td>97.48%</td>
<td>96.02%</td>
<td>100.00%</td>
<td>97.40%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>100.00%</td>
<td>100.00%</td>
<td>89.18%</td>
<td>96.10%</td>
<td>94.95%</td>
<td>87.12%</td>
<td>100.00%</td>
<td>95.67%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>100.00%</td>
<td>91.76%</td>
<td>95.41%</td>
<td>97.02%</td>
<td>97.66%</td>
<td>100.00%</td>
<td>95.63%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>100.00%</td>
<td>93.65%</td>
<td>95.18%</td>
<td>97.48%</td>
<td>97.42%</td>
<td>100.00%</td>
<td>96.99%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>99.26%</td>
<td>99.75%</td>
<td>88.71%</td>
<td>94.27%</td>
<td>96.79%</td>
<td>93.91%</td>
<td>100.00%</td>
<td>95.41%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV</td>
<td>98.95%</td>
<td>100.00%</td>
<td>97.63%</td>
<td>61.27%</td>
<td>84.39%</td>
<td>94.08%</td>
<td>100.00%</td>
<td>87.11%</td>
</tr>
<tr>
<td>MAVS</td>
<td>86.32%</td>
<td>63.92%</td>
<td>46.75%</td>
<td>13.29%</td>
<td>32.37%</td>
<td>55.03%</td>
<td>19.38%</td>
<td>44.40%</td>
</tr>
<tr>
<td>ZC</td>
<td>96.84%</td>
<td>100.00%</td>
<td>94.67%</td>
<td>86.13%</td>
<td>95.95%</td>
<td>86.39%</td>
<td>99.38%</td>
<td>94.40%</td>
</tr>
<tr>
<td>SSC</td>
<td>66.32%</td>
<td>50.63%</td>
<td>62.72%</td>
<td>66.47%</td>
<td>54.91%</td>
<td>62.13%</td>
<td>93.13%</td>
<td>59.77%</td>
</tr>
<tr>
<td>WL</td>
<td>98.95%</td>
<td>99.37%</td>
<td>98.82%</td>
<td>95.38%</td>
<td>97.11%</td>
<td>94.08%</td>
<td>99.38%</td>
<td>97.79%</td>
</tr>
<tr>
<td>TD Set</td>
<td>96.84%</td>
<td>100.00%</td>
<td>91.33%</td>
<td>97.69%</td>
<td>95.27%</td>
<td>100.00%</td>
<td>97.14%</td>
<td></td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>100.00%</td>
<td>100.00%</td>
<td>92.31%</td>
<td>95.95%</td>
<td>96.53%</td>
<td>94.67%</td>
<td>100.00%</td>
<td>96.61%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>100.00%</td>
<td>92.90%</td>
<td>95.95%</td>
<td>97.11%</td>
<td>98.82%</td>
<td>100.00%</td>
<td>96.88%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>100.00%</td>
<td>93.49%</td>
<td>95.95%</td>
<td>97.69%</td>
<td>98.82%</td>
<td>100.00%</td>
<td>97.14%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>100.00%</td>
<td>100.00%</td>
<td>89.35%</td>
<td>94.80%</td>
<td>97.69%</td>
<td>97.63%</td>
<td>100.00%</td>
<td>95.96%</td>
</tr>
</tbody>
</table>

Majority Voting 50ms Windows and 250ms Total Delay

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>TD Set</td>
<td>100.00%</td>
<td>100.00%</td>
<td>99.88%</td>
<td>94.34%</td>
<td>97.01%</td>
<td>98.48%</td>
<td>100.00%</td>
<td>97.93%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>99.44%</td>
<td>100.00%</td>
<td>93.51%</td>
<td>95.33%</td>
<td>97.35%</td>
<td>95.31%</td>
<td>100.00%</td>
<td>96.53%</td>
</tr>
</tbody>
</table>
Table 14. Displayed in this table are the classification accuracies of the offline window and feature analysis results for male subject 5.

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>50ms Windows</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>89.78%</td>
<td>82.08%</td>
<td>81.48%</td>
<td>16.86%</td>
<td>43.25%</td>
<td>82.46%</td>
<td>80.76%</td>
<td>60.68%</td>
</tr>
<tr>
<td>MAVS</td>
<td>90.69%</td>
<td>59.84%</td>
<td>79.09%</td>
<td>40.80%</td>
<td>26.56%</td>
<td>81.18%</td>
<td>65.65%</td>
<td>56.94%</td>
</tr>
<tr>
<td>ZC</td>
<td>90.33%</td>
<td>82.20%</td>
<td>63.98%</td>
<td>53.77%</td>
<td>36.55%</td>
<td>81.77%</td>
<td>75.43%</td>
<td>63.28%</td>
</tr>
<tr>
<td>SSC</td>
<td>33.94%</td>
<td>52.81%</td>
<td>39.09%</td>
<td>34.43%</td>
<td>45.18%</td>
<td>29.50%</td>
<td>61.09%</td>
<td>41.66%</td>
</tr>
<tr>
<td>WL</td>
<td>97.63%</td>
<td>92.97%</td>
<td>97.16%</td>
<td>82.31%</td>
<td>69.81%</td>
<td>93.03%</td>
<td>96.96%</td>
<td>87.19%</td>
</tr>
<tr>
<td>TD Set</td>
<td>91.97%</td>
<td>96.96%</td>
<td>96.59%</td>
<td>88.92%</td>
<td>68.44%</td>
<td>91.87%</td>
<td>94.02%</td>
<td>88.23%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>97.45%</td>
<td>75.29%</td>
<td>79.43%</td>
<td>93.75%</td>
<td>66.17%</td>
<td>90.36%</td>
<td>97.61%</td>
<td>81.13%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>99.09%</td>
<td>85.95%</td>
<td>83.52%</td>
<td>93.99%</td>
<td>83.31%</td>
<td>89.66%</td>
<td>97.50%</td>
<td>88.33%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>98.72%</td>
<td>85.25%</td>
<td>86.14%</td>
<td>93.28%</td>
<td>87.17%</td>
<td>91.29%</td>
<td>97.28%</td>
<td>89.40%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>95.26%</td>
<td>90.52%</td>
<td>80.57%</td>
<td>93.99%</td>
<td>71.28%</td>
<td>94.77%</td>
<td>97.17%</td>
<td>85.49%</td>
</tr>
<tr>
<td><strong>100ms Windows</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>92.34%</td>
<td>85.41%</td>
<td>87.24%</td>
<td>16.08%</td>
<td>49.20%</td>
<td>82.52%</td>
<td>84.31%</td>
<td>64.15%</td>
</tr>
<tr>
<td>MAVS</td>
<td>82.85%</td>
<td>45.65%</td>
<td>60.14%</td>
<td>6.15%</td>
<td>23.92%</td>
<td>67.37%</td>
<td>50.11%</td>
<td>40.80%</td>
</tr>
<tr>
<td>ZC</td>
<td>90.88%</td>
<td>88.71%</td>
<td>74.49%</td>
<td>67.14%</td>
<td>48.87%</td>
<td>86.25%</td>
<td>83.88%</td>
<td>71.70%</td>
</tr>
<tr>
<td>SSC</td>
<td>33.21%</td>
<td>68.00%</td>
<td>50.57%</td>
<td>45.15%</td>
<td>54.67%</td>
<td>45.45%</td>
<td>67.97%</td>
<td>51.65%</td>
</tr>
<tr>
<td>WL</td>
<td>98.18%</td>
<td>97.41%</td>
<td>98.63%</td>
<td>89.83%</td>
<td>86.33%</td>
<td>94.87%</td>
<td>97.17%</td>
<td>93.75%</td>
</tr>
<tr>
<td>TD Set</td>
<td>91.24%</td>
<td>98.12%</td>
<td>98.41%</td>
<td>91.49%</td>
<td>76.54%</td>
<td>92.31%</td>
<td>93.46%</td>
<td>91.10%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>99.27%</td>
<td>80.71%</td>
<td>87.93%</td>
<td>98.82%</td>
<td>79.95%</td>
<td>94.87%</td>
<td>99.35%</td>
<td>88.50%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>93.18%</td>
<td>93.39%</td>
<td>98.82%</td>
<td>94.53%</td>
<td>95.80%</td>
<td>98.04%</td>
<td>95.65%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>100.00%</td>
<td>93.41%</td>
<td>95.44%</td>
<td>98.58%</td>
<td>95.90%</td>
<td>96.27%</td>
<td>97.60%</td>
<td>96.40%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>95.62%</td>
<td>94.35%</td>
<td>92.94%</td>
<td>98.58%</td>
<td>87.70%</td>
<td>96.50%</td>
<td>97.60%</td>
<td>93.65%</td>
</tr>
<tr>
<td><strong>250ms Windows</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAV</td>
<td>91.92%</td>
<td>90.53%</td>
<td>87.93%</td>
<td>17.26%</td>
<td>56.00%</td>
<td>82.94%</td>
<td>83.98%</td>
<td>66.75%</td>
</tr>
<tr>
<td>MAVS</td>
<td>76.77%</td>
<td>26.63%</td>
<td>32.76%</td>
<td>6.55%</td>
<td>25.14%</td>
<td>55.88%</td>
<td>39.78%</td>
<td>29.68%</td>
</tr>
<tr>
<td>ZC</td>
<td>89.90%</td>
<td>94.67%</td>
<td>81.61%</td>
<td>75.60%</td>
<td>61.14%</td>
<td>82.94%</td>
<td>87.85%</td>
<td>79.62%</td>
</tr>
<tr>
<td>SSC</td>
<td>31.31%</td>
<td>79.88%</td>
<td>75.29%</td>
<td>63.69%</td>
<td>69.14%</td>
<td>62.35%</td>
<td>78.45%</td>
<td>66.88%</td>
</tr>
<tr>
<td>WL</td>
<td>95.96%</td>
<td>97.04%</td>
<td>99.43%</td>
<td>90.48%</td>
<td>95.43%</td>
<td>93.53%</td>
<td>97.24%</td>
<td>95.67%</td>
</tr>
<tr>
<td>TD Set</td>
<td>87.88%</td>
<td>98.22%</td>
<td>99.43%</td>
<td>91.07%</td>
<td>86.86%</td>
<td>91.18%</td>
<td>90.61%</td>
<td>93.12%</td>
</tr>
<tr>
<td>1 AR Feature</td>
<td>100.00%</td>
<td>87.57%</td>
<td>91.38%</td>
<td>100.00%</td>
<td>90.29%</td>
<td>95.29%</td>
<td>100.00%</td>
<td>93.25%</td>
</tr>
<tr>
<td>2 AR Features</td>
<td>100.00%</td>
<td>97.63%</td>
<td>100.00%</td>
<td>99.40%</td>
<td>99.43%</td>
<td>97.65%</td>
<td>98.90%</td>
<td>99.24%</td>
</tr>
<tr>
<td>3 AR Features</td>
<td>98.99%</td>
<td>98.82%</td>
<td>99.43%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>97.65%</td>
<td>98.34%</td>
<td>99.49%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>95.96%</td>
<td>98.82%</td>
<td>98.28%</td>
<td>100.00%</td>
<td>97.14%</td>
<td>93.53%</td>
<td>98.34%</td>
<td>98.22%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMG Features</th>
<th>Rest</th>
<th>Grasping</th>
<th>Releasing</th>
<th>Pronation</th>
<th>Supination</th>
<th>Extension</th>
<th>Flexion</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority Voting 50ms Windows and 250ms Total Delay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TD Set</td>
<td>96.14%</td>
<td>98.47%</td>
<td>99.54%</td>
<td>97.39%</td>
<td>73.31%</td>
<td>96.15%</td>
<td>99.56%</td>
<td>92.53%</td>
</tr>
<tr>
<td>4 AR Features</td>
<td>96.51%</td>
<td>96.71%</td>
<td>92.58%</td>
<td>99.76%</td>
<td>80.25%</td>
<td>98.60%</td>
<td>98.03%</td>
<td>92.68%</td>
</tr>
</tbody>
</table>
Table 15. Displayed here are the classification accuracies for all the healthy subjects using a 250ms adjacent windowing scheme across all types of features. The percentages are the average across all classes using the features specified at the top of each column.

<table>
<thead>
<tr>
<th>Subject</th>
<th>MAV</th>
<th>MAVS</th>
<th>ZC</th>
<th>SSC</th>
<th>WL</th>
<th>TD Set</th>
<th>1 AR</th>
<th>2 AR</th>
<th>3 AR</th>
<th>4 AR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female 1</td>
<td>79.31%</td>
<td>27.76%</td>
<td>85.22%</td>
<td>47.04%</td>
<td>96.92%</td>
<td>95.50%</td>
<td>95.37%</td>
<td>97.81%</td>
<td>98.20%</td>
<td>98.07%</td>
</tr>
<tr>
<td>Female 2</td>
<td>43.14%</td>
<td>31.37%</td>
<td>65.23%</td>
<td>54.90%</td>
<td>92.42%</td>
<td>80.00%</td>
<td>94.77%</td>
<td>96.60%</td>
<td>97.52%</td>
<td>97.52%</td>
</tr>
<tr>
<td>Female 3</td>
<td>56.27%</td>
<td>29.92%</td>
<td>79.03%</td>
<td>62.40%</td>
<td>94.50%</td>
<td>87.72%</td>
<td>92.58%</td>
<td>96.93%</td>
<td>97.31%</td>
<td>96.80%</td>
</tr>
<tr>
<td>Female 4</td>
<td>60.18%</td>
<td>28.92%</td>
<td>71.85%</td>
<td>43.45%</td>
<td>87.55%</td>
<td>83.79%</td>
<td>87.24%</td>
<td>93.04%</td>
<td>93.56%</td>
<td>93.43%</td>
</tr>
<tr>
<td>Female 5</td>
<td>71.70%</td>
<td>21.90%</td>
<td>72.76%</td>
<td>56.88%</td>
<td>82.24%</td>
<td>87.58%</td>
<td>79.05%</td>
<td>89.33%</td>
<td>95.65%</td>
<td>96.31%</td>
</tr>
<tr>
<td>Male 1</td>
<td>52.32%</td>
<td>26.80%</td>
<td>78.74%</td>
<td>39.30%</td>
<td>92.27%</td>
<td>84.92%</td>
<td>94.87%</td>
<td>95.90%</td>
<td>96.03%</td>
<td>95.00%</td>
</tr>
<tr>
<td>Male 2</td>
<td>23.29%</td>
<td>20.31%</td>
<td>46.83%</td>
<td>37.13%</td>
<td>65.72%</td>
<td>59.90%</td>
<td>75.16%</td>
<td>83.05%</td>
<td>83.31%</td>
<td>80.08%</td>
</tr>
<tr>
<td>Male 3</td>
<td>48.28%</td>
<td>24.67%</td>
<td>57.69%</td>
<td>55.31%</td>
<td>92.31%</td>
<td>91.25%</td>
<td>88.59%</td>
<td>85.81%</td>
<td>90.98%</td>
<td>89.92%</td>
</tr>
<tr>
<td>Male 4</td>
<td>87.11%</td>
<td>44.40%</td>
<td>94.40%</td>
<td>59.77%</td>
<td>97.79%</td>
<td>97.14%</td>
<td>96.61%</td>
<td>96.88%</td>
<td>97.14%</td>
<td>95.96%</td>
</tr>
<tr>
<td>Male 5</td>
<td>66.75%</td>
<td>29.68%</td>
<td>79.62%</td>
<td>66.88%</td>
<td>95.67%</td>
<td>93.12%</td>
<td>93.25%</td>
<td>99.24%</td>
<td>99.49%</td>
<td>98.22%</td>
</tr>
<tr>
<td>Average</td>
<td>58.83%</td>
<td>28.57%</td>
<td>73.14%</td>
<td>52.31%</td>
<td>89.74%</td>
<td>86.09%</td>
<td>89.75%</td>
<td>93.46%</td>
<td>94.92%</td>
<td>94.13%</td>
</tr>
</tbody>
</table>

Table 16. Displayed here are the offline results for all healthy subjects comparing an adjacent windowing scheme of 250ms with a majority voting scheme of 50ms with a 250ms window. The percentages are the average across all classes using the scheme specified at the top of each column.

<table>
<thead>
<tr>
<th>Subject</th>
<th>4 AR</th>
<th>4 AR + MV</th>
<th>TD Set</th>
<th>TD Set + MV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female 1</td>
<td>98.07%</td>
<td>97.24%</td>
<td>95.50%</td>
<td>96.89%</td>
</tr>
<tr>
<td>Female 2</td>
<td>97.52%</td>
<td>92.93%</td>
<td>80.00%</td>
<td>86.15%</td>
</tr>
<tr>
<td>Female 3</td>
<td>96.80%</td>
<td>95.77%</td>
<td>87.72%</td>
<td>92.29%</td>
</tr>
<tr>
<td>Female 4</td>
<td>93.43%</td>
<td>86.20%</td>
<td>83.79%</td>
<td>82.23%</td>
</tr>
<tr>
<td>Female 5</td>
<td>96.31%</td>
<td>79.19%</td>
<td>87.58%</td>
<td>82.00%</td>
</tr>
<tr>
<td>Male 1</td>
<td>95.00%</td>
<td>93.97%</td>
<td>84.92%</td>
<td>91.27%</td>
</tr>
<tr>
<td>Male 2</td>
<td>80.08%</td>
<td>70.30%</td>
<td>59.90%</td>
<td>55.86%</td>
</tr>
<tr>
<td>Male 3</td>
<td>89.92%</td>
<td>91.22%</td>
<td>91.25%</td>
<td>92.57%</td>
</tr>
<tr>
<td>Male 4</td>
<td>95.96%</td>
<td>96.53%</td>
<td>97.14%</td>
<td>97.93%</td>
</tr>
<tr>
<td>Male 5</td>
<td>98.22%</td>
<td>92.68%</td>
<td>93.12%</td>
<td>92.53%</td>
</tr>
<tr>
<td>Average</td>
<td>94.13%</td>
<td>89.60%</td>
<td>86.09%</td>
<td>86.97%</td>
</tr>
</tbody>
</table>

Table 16. Displayed here are the offline results for all healthy subjects comparing an adjacent windowing scheme of 250ms with a majority voting scheme of 50ms with a 250ms window. The percentages are the average across all classes using the scheme specified at the top of each column.

4.1.2 Real Time Pattern Recognition Myoelectric Control Scheme for Healthy Subjects

As seen from the previous section in Tables 15 and 16, the AR features using a regular adjacent windowing scheme had much higher classification accuracy in comparison to using the time domain features. It also showed that the majority voting scheme did not offer any advantages either. Therefore, the AR features using a 250ms adjacent windowing scheme with no majority voting was used in the real time myoelectric control scheme. The number of AR coefficients used from each channel in the real time system was two. Although Table 15 shows that using three and four AR coefficients slightly improves the classification accuracy, testing of the real time system resulted in very poor results until the number of
AR coefficients from each EMG channel was decreased to two. This choice will be discussed further in the following chapter. The results of the real time testing for each subject was compiled and displayed as seen in Table 17 for the first female subject. Tables 18 through 26 display the rest of the real time results for the remaining subjects. For Tables 17 through 26, the incorrect output of the classifier was shown when classes were misclassified. The subjects used for the real time analysis were the same subjects that were used in the offline tests shown in the previous section. A summary of all the real time results is shown in Table 27. In order to gain a general understanding of what classes were misclassified; one is directed to Figure 35. Figure 35 shows the percentage of each class that was misclassified out of the total number of misclassifications for all subjects combined. It was also insightful to see what the classifier output was for each command when misclassifications were occurring. This allowed one to see what motions had a similar EMG contraction and could easily be misclassified for one another. Figure 36 shows the output percentage of each class during misclassifications of rest over all subjects. It should be noted that the figure could appear confusing because rest is listed as a percentage of the misclassified output even though the command was for rest. This confusion is explained by looking at female subject five’s data in Table 21. In the third set of her data during a rest command, the classifier output was rest and pronation. The misclassification in this table shows that the classifier output alternated between rest and pronation while the rest command was given. This was the only case among all subjects where a misclassification like this occurred during a rest command. The percentage of rest in Figure 36 was therefore calculated as one half divided by three which was the number of alternating rest outputs during misclassification for the rest command divided by the number of times rest was misclassified. The same analysis was performed for all other classes and is shown in Figure 37 to Figure 41 for grasping, releasing, pronation, supination and extension respectively. There is no figure for flexion because flexion had a 100% classification. In cases where the output was misclassified and outputs alternated between three different outputs during the misclassification, a value of one third was given to the occurrences of each class which was later used to find the percentages for each class.
### Table 17. Displayed here are the real time pattern recognition myoelectric control scheme results for female subject 1.

<table>
<thead>
<tr>
<th>Command</th>
<th>Set 1</th>
<th></th>
<th></th>
<th>Command</th>
<th>Set 2</th>
<th></th>
<th></th>
<th>Command</th>
<th>Set 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class</td>
<td>Classified</td>
<td>Misclassified</td>
<td></td>
<td>Class</td>
<td>Classified</td>
<td>Misclassified</td>
<td></td>
<td>Class</td>
</tr>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>4</td>
<td>Extension</td>
<td>Sup/Rel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>Releasing</td>
<td></td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>Ex/Flex</td>
<td></td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Classification Accuracy** 90.48%  95.24%  90.48%

**Total Accuracy** 92.06%

### Table 18. Displayed here are the real time pattern recognition myoelectric control scheme results for female subject 2.

<table>
<thead>
<tr>
<th>Command</th>
<th>Set 1</th>
<th></th>
<th></th>
<th>Command</th>
<th>Set 2</th>
<th></th>
<th></th>
<th>Command</th>
<th>Set 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class</td>
<td>Classified</td>
<td>Misclassified</td>
<td></td>
<td>Class</td>
<td>Classified</td>
<td>Misclassified</td>
<td></td>
<td>Class</td>
</tr>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>Ext</td>
<td></td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>Pro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>4</td>
<td>Extension</td>
<td>Pro/Ext</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>Pro/Rel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>Pro</td>
<td></td>
<td></td>
<td>12</td>
<td>Releasing</td>
<td>Rel/Pro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td>13</td>
<td>Pronation</td>
<td>Pro/Sup/Ext</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>Rel/Ext</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td>16</td>
<td>Supination</td>
<td>Sup/Ext</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Classification Accuracy** 85.71%  66.67%  71.43%

**Total Accuracy** 74.50%
### Female Subject 3

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>Ext/Sup</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Sup/Ext</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Table 19. Displayed here are the real time pattern recognition myoelectric control scheme results for female subject 3.

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Table 20. Displayed here are the real time pattern recognition myoelectric control scheme results for female subject 4.
<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>Gra/Pro</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>Pro/Sup/Ext</td>
<td></td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>Gra/Pro</td>
<td></td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Releasing</td>
<td>Rel/Pro/Sup</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>Gra/Pro</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>Rest/Pro</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>Pro/Rel</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy: 80.95% 95.24% 90.48%
Total Accuracy: 88.89%

Table 21. Displayed here are the real time pattern recognition myoelectric control scheme results for female subject 5.

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>Ext</td>
<td></td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>Pro</td>
<td></td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>Pro</td>
<td></td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Pronation</td>
<td>Gra</td>
<td></td>
<td>13</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>Pro</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>Sup</td>
<td></td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Releasing</td>
<td>Sup/Pro</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy: 76.19% 90.48% 95.24%
Total Accuracy: 87.30%

Table 22. Displayed here are the real time pattern recognition myoelectric control scheme results for male subject 1.
## Male Subject 2

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td>3</td>
<td>Releasing</td>
<td>Ext</td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td>5</td>
<td>Releasing</td>
<td>Sup/Gra</td>
<td>5</td>
<td>Releasing</td>
<td>Gra/Sup/Rel</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td>9</td>
<td>Grasping</td>
<td>Sup</td>
<td>9</td>
<td>Supination</td>
<td>Ext/Pro/Sup</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>Sup/Ext</td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>Sup</td>
<td>12</td>
<td>Releasing</td>
<td>Sup/Gra</td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td>13</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>Ext/Pro</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td>19</td>
<td>Releasing</td>
<td>Sup/Gra/Rel</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Classification Accuracy:** 85.71%  
**Total Accuracy:** 84.13%

### Table 23
Displayed here are the real time pattern recognition myoelectric control scheme results for male subject 2.

## Male Subject 3

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td>13</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td>16</td>
<td>Supination</td>
<td>Sup/Ext</td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td>19</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td>20</td>
<td>Pronation</td>
<td>Pro/Rel/Gra</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Classification Accuracy:** 95.24%  
**Total Accuracy:** 95.24%

### Table 24
Displayed here are the real time pattern recognition myoelectric control scheme results for male subject 3.
### Male Subject 4

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td>Rel/Sup</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td>13</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Grasping</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td>Sup/Pro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td>19</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Classification Accuracy** 100.00% 100.00% 90.48%
**Total Accuracy** 96.83%

Table 25. Displayed here are the real time pattern recognition myoelectric control scheme results for male subject 4.

### Male Subject 5

<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td>4</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>x</td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td>Sup/Pro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td>10</td>
<td>Flexion</td>
<td>x</td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td>Ext/Pro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td>12</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>x</td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td>13</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Releasing</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td>17</td>
<td>Flexion</td>
<td>x</td>
<td>17</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td>19</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td>21</td>
<td>Flexion</td>
<td>x</td>
<td>21</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Classification Accuracy** 90.48% 100.00% 90.48%
**Total Accuracy** 93.65%

Table 26. Displayed here are the real time pattern recognition myoelectric control scheme results for male subject 5.
<table>
<thead>
<tr>
<th>Subject</th>
<th>Set 1</th>
<th>Set 2</th>
<th>Set 3</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female 1</td>
<td>90.48%</td>
<td>95.24%</td>
<td>90.48%</td>
<td>92.06%</td>
</tr>
<tr>
<td>Female 2</td>
<td>85.71%</td>
<td>66.67%</td>
<td>71.43%</td>
<td>74.60%</td>
</tr>
<tr>
<td>Female 3</td>
<td>100.00%</td>
<td>90.48%</td>
<td>95.24%</td>
<td>95.24%</td>
</tr>
<tr>
<td>Female 4</td>
<td>80.95%</td>
<td>80.95%</td>
<td>100.00%</td>
<td>87.30%</td>
</tr>
<tr>
<td>Female 5</td>
<td>80.95%</td>
<td>95.24%</td>
<td>90.48%</td>
<td>88.89%</td>
</tr>
<tr>
<td>Male 1</td>
<td>76.19%</td>
<td>90.48%</td>
<td>95.24%</td>
<td>87.30%</td>
</tr>
<tr>
<td>Male 2</td>
<td>85.71%</td>
<td>80.95%</td>
<td>85.71%</td>
<td>84.13%</td>
</tr>
<tr>
<td>Male 3</td>
<td>95.24%</td>
<td>95.24%</td>
<td>95.24%</td>
<td>95.24%</td>
</tr>
<tr>
<td>Male 4</td>
<td>100.00%</td>
<td>100.00%</td>
<td>90.48%</td>
<td>96.83%</td>
</tr>
<tr>
<td>Male 5</td>
<td>90.48%</td>
<td>100.00%</td>
<td>90.48%</td>
<td>93.65%</td>
</tr>
<tr>
<td>Average</td>
<td>88.57%</td>
<td>89.52%</td>
<td>90.48%</td>
<td>89.52%</td>
</tr>
</tbody>
</table>

Table 27. Displayed here is a summary of the real time results for all of the healthy subjects.

Figure 35. Displayed here is the percentage of each class that was misclassified out of the total number of misclassifications for the healthy subjects.

Figure 36. Displayed here is the percentage of each class that was output from the classifier during misclassifications of the rest class.
Figure 37. Displayed here is the percentage of each class that was output from the classifier during misclassifications of the grasping class.

Figure 38. Displayed here is the percentage of each class that was output from the classifier during misclassifications of the releasing class.
Figure 39. Displayed here is the percentage of each class that was output from the classifier during misclassifications of the pronation class.

Figure 40. Displayed here is the percentage of each class that was output from the classifier during misclassifications of the supination class.
4.1.3 Real Time Pattern Recognition Myoelectric Control Scheme for a CCS Subject

The testing for the subject with CCS proceeded a bit differently than it did for the healthy subjects. One DOF controllers were built to test each DOF separately before testing the three DOF controller the healthy subjects were tested with. As explained in the previous chapter, each single DOF controller had three classes which included a class for rest. The single DOF controllers were tested with three sets of randomized commands. Table 28 displays the test results for each of the single DOF controllers. In this table, the incorrect output of the classifier is shown when classes were misclassified. Once the tests were completed for the single DOF controllers, the full three DOF controller was tested and those results are shown in Table 29. This table also shows the incorrect output of the classifier when classes were misclassified. In order to gain a general understanding of what classes were misclassified with the full three DOF controller; one is directed to Figure 42. Figure 42 shows the percentage of each class that was misclassified out of the total number of misclassifications.
### Elbow Controller

<table>
<thead>
<tr>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Flexion</td>
<td>x</td>
<td></td>
<td>1 Flexion</td>
<td>x</td>
<td></td>
<td>1 Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2 Extension</td>
<td>x</td>
<td></td>
<td>2 Extension</td>
<td>x</td>
<td></td>
<td>2 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3 Rest</td>
<td>x</td>
<td></td>
<td>3 Flexion</td>
<td>x</td>
<td></td>
<td>3 Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4 Extension</td>
<td>x</td>
<td></td>
<td>4 Extension</td>
<td>x</td>
<td></td>
<td>4 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5 Extension</td>
<td>x</td>
<td></td>
<td>5 Rest</td>
<td>x</td>
<td></td>
<td>5 Flexion</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>6 Extension</td>
<td>x</td>
<td></td>
<td>6 Extension</td>
<td>x</td>
<td></td>
<td>6 Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7 Flexion</td>
<td>x</td>
<td></td>
<td>7 Rest</td>
<td>x</td>
<td></td>
<td>7 Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8 Flexion</td>
<td>x</td>
<td></td>
<td>8 Flexion</td>
<td>x</td>
<td></td>
<td>8 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>9 Rest</td>
<td>x</td>
<td></td>
<td>9 Flexion</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy: 100.00%
Total Accuracy: 100.00%

### Wrist Controller

<table>
<thead>
<tr>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Pronation</td>
<td>Pro/Sup</td>
<td></td>
<td>1 Supination</td>
<td>x</td>
<td></td>
<td>1 Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2 Supination</td>
<td>x</td>
<td></td>
<td>2 Supination</td>
<td>x</td>
<td></td>
<td>2 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3 Rest</td>
<td>x</td>
<td></td>
<td>3 Rest</td>
<td>x</td>
<td></td>
<td>3 Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4 Supination</td>
<td>x</td>
<td></td>
<td>4 Pronation</td>
<td>x</td>
<td></td>
<td>4 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5 Pronation</td>
<td>x</td>
<td></td>
<td>5 Rest</td>
<td>x</td>
<td></td>
<td>5 Supination</td>
<td>Pro</td>
<td></td>
</tr>
<tr>
<td>6 Rest</td>
<td>x</td>
<td></td>
<td>6 Pronation</td>
<td>x</td>
<td></td>
<td>6 Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7 Pronation</td>
<td>x</td>
<td></td>
<td>7 Rest</td>
<td>x</td>
<td></td>
<td>7 Rest</td>
<td>x</td>
<td>Rest/Sup</td>
</tr>
<tr>
<td>8 Supination</td>
<td>x</td>
<td></td>
<td>8 Supination</td>
<td>x</td>
<td></td>
<td>8 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>9 Rest</td>
<td>x</td>
<td></td>
<td>9 Pronation</td>
<td>x</td>
<td></td>
<td>9 Pronation</td>
<td>Pro/Sup</td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy: 88.89%
Total Accuracy: 80.95%

### Hand Controller

<table>
<thead>
<tr>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Grasping</td>
<td>x</td>
<td></td>
<td>1 Grasping</td>
<td>x</td>
<td></td>
<td>1 Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2 Releasing</td>
<td>x</td>
<td></td>
<td>2 Grasping</td>
<td>x</td>
<td></td>
<td>2 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>3 Rest</td>
<td>x</td>
<td></td>
<td>3 Rest</td>
<td>x</td>
<td></td>
<td>3 Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4 Grasping</td>
<td>x</td>
<td></td>
<td>4 Releasing</td>
<td>x</td>
<td></td>
<td>4 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>5 Releasing</td>
<td>x</td>
<td></td>
<td>5 Rest</td>
<td>x</td>
<td></td>
<td>5 Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>6 Rest</td>
<td>x</td>
<td></td>
<td>6 Releasing</td>
<td>x</td>
<td></td>
<td>6 Grasping</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7 Releasing</td>
<td>x</td>
<td></td>
<td>7 Rest</td>
<td>x</td>
<td></td>
<td>7 Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8 Grasping</td>
<td>x</td>
<td></td>
<td>8 Grasping</td>
<td>x</td>
<td></td>
<td>8 Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>9 Rest</td>
<td>Rel/Rest</td>
<td></td>
<td>9 Releasing</td>
<td>x</td>
<td></td>
<td>9 Grasping</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy: 88.89%
Total Accuracy: 95.24%

Table 28. Displayed in this table are the results of the single DOF controllers used for the CCS subject.
<table>
<thead>
<tr>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
<th>Command</th>
<th>Class</th>
<th>Classified</th>
<th>Misclassified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>1</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>2</td>
<td>Flexion</td>
<td>Flex/Sup</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Grasping</td>
<td>Pro/Rel</td>
<td></td>
<td>3</td>
<td>Releasing</td>
<td>Pro</td>
<td></td>
<td>3</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>4</td>
<td>Grasping</td>
<td>Gra/Rel</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Rest</td>
<td>Rest/Rel</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>5</td>
<td>Releasing</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>6</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Extension</td>
<td>Gra/Rel</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>Rest/Gra</td>
<td></td>
<td>7</td>
<td>Rest</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Flexion</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>8</td>
<td>Flexion</td>
<td>Sup</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>9</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>10</td>
<td>Flexion</td>
<td>Sup</td>
<td></td>
<td>10</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>11</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Releasing</td>
<td>Sup</td>
<td></td>
<td>12</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>12</td>
<td>Flexion</td>
<td>Flex/Sup</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Flexion</td>
<td>Flex/Sup</td>
<td></td>
<td>13</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>13</td>
<td>Releasing</td>
<td>Ext/Flex/Sup</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Releasing</td>
<td>Rel/Sup</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>14</td>
<td>Supination</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>15</td>
<td>Rest</td>
<td>Rest/Gra/Rel</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>16</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>17</td>
<td>Flexion</td>
<td>Fle/Sup</td>
<td></td>
<td>17</td>
<td>Grasping</td>
<td>Rel</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Releasing</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
<td>18</td>
<td>Extension</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Pronation</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>19</td>
<td>Releasing</td>
<td>Re/Ext</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Rest</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Grasping</td>
<td>x</td>
<td></td>
<td>20</td>
<td>Pronation</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Supination</td>
<td>x</td>
<td></td>
<td>21</td>
<td>Flexion</td>
<td>Sup</td>
<td></td>
<td>21</td>
<td>Grasping</td>
<td>Rel</td>
<td></td>
</tr>
</tbody>
</table>

Classification Accuracy: 71.43%, 76.19%, 57.14%
Total Accuracy: 68.25%

Table 29. Displayed in this table are the results for the three DOF controller for the CCS subject.

![Class Misclassifications](image)

Figure 42. Displayed here is the percentage of each class that was misclassified out of the total number of misclassifications for the CCS subject.

4.2 Myoelectric Control Scheme Based on a Time Delayed Neural Network Results

By varying all of the parameters for the TDNN which included total delay, delay interval, and hidden layer neurons, thirty-six different neural networks were created for the first subject. Each neural
network was trained using the six designated training trials of data. Once trained, the neural networks were then tested with the six trials of test data. A comparison of the results while varying all of the parameters of the TDNNs for subject one is shown in Table 30. The main measurement for the TDNNs was the average error of the TDNN in degrees. The average error was calculated for both the elbow and shoulder DOF. One can note from Table 30 that changing the number of neurons in the hidden layer did not have an effect on the accuracy of the TDNNs and therefore the rest of the results will only consider TDNNs having ten neurons in the hidden layer. To summarize the results for the first subject, Table 31 was created. Table 31 displays the average error for the shoulder and elbow DOF while varying the total delay and delay interval for subject one. This table shows that for subject one, the total time delay of 300ms and a delay interval of 100ms and 150ms produced the most accurate results. In the methodology it was explained that the first subject was used to find the optimal parameters for the TDNN. Those optimal parameters would then be tested with the data from the remaining subjects. The two TDNNs using 300ms and a delay interval of 100ms and 150ms produced the least amount of error for the first subject. It was assumed that using 100ms delay intervals would provide more information from the EMG signals because it would include one more data point to be input to the TDNN from each EMG channel than using an interval of 150ms. Therefore, the TDNN parameters used for the rest of the subjects were a total delay of 300ms, delay intervals of 100ms, and 10 neurons in the hidden layer. Table 32 displays the average error of the shoulder and elbow combined for all subjects using the optimal TDNN parameters. Table 33 displays an overall average error for each type of motion and both DOF for each subject. Also included in the last two tables are standard deviations which show the variability of the errors. All of the results presented thus far consisted of a quantitative analysis for the TDNNs. A qualitative analysis is also helpful. Figures 43 through 48 display the output of the TDNN using the optimal TDNN parameters for each of the testing motions of subject one. The same figures were created for subjects two through five and correspond to Figures 49 to 54, Figures 55 to 60, Figures 61 to 66, and Figures 67 to 72 respectively.
Although the subject with CCS was not formally included in the TDNN methodology, a TDNN was designed which output his elbow position using inputs from the biceps and triceps. The results are pictured in Figure 73.

<table>
<thead>
<tr>
<th>Time Delay (ms)</th>
<th>Time Interval (ms)</th>
<th>Number of Neurons</th>
<th>Slow Elbow Movement</th>
<th>Fast Elbow Movement</th>
<th>Slow Shoulder Movement</th>
<th>Fast Shoulder Movement</th>
<th>Slow Reaching Movement</th>
<th>Fast Reaching Movement</th>
<th>Average Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>50</td>
<td>10</td>
<td>6.3</td>
<td>26.3</td>
<td>7.5</td>
<td>22.7</td>
<td>14.9</td>
<td>21.7</td>
<td>13.0</td>
</tr>
<tr>
<td>300</td>
<td>50</td>
<td>20</td>
<td>6.6</td>
<td>26.0</td>
<td>7.6</td>
<td>20.9</td>
<td>14.9</td>
<td>20.7</td>
<td>14.0</td>
</tr>
<tr>
<td>300</td>
<td>50</td>
<td>30</td>
<td>5.8</td>
<td>24.4</td>
<td>8.0</td>
<td>22.6</td>
<td>14.5</td>
<td>21.7</td>
<td>13.2</td>
</tr>
<tr>
<td>300</td>
<td>50</td>
<td>40</td>
<td>5.7</td>
<td>25.1</td>
<td>7.5</td>
<td>22.8</td>
<td>14.7</td>
<td>20.9</td>
<td>13.5</td>
</tr>
<tr>
<td>300</td>
<td>100</td>
<td>10</td>
<td>6.0</td>
<td>25.9</td>
<td>7.5</td>
<td>21.7</td>
<td>16.4</td>
<td>20.4</td>
<td>13.2</td>
</tr>
<tr>
<td>300</td>
<td>100</td>
<td>20</td>
<td>6.6</td>
<td>26.0</td>
<td>7.6</td>
<td>20.9</td>
<td>14.9</td>
<td>20.7</td>
<td>14.0</td>
</tr>
<tr>
<td>300</td>
<td>100</td>
<td>30</td>
<td>5.5</td>
<td>24.9</td>
<td>7.0</td>
<td>20.3</td>
<td>14.1</td>
<td>20.5</td>
<td>13.6</td>
</tr>
<tr>
<td>300</td>
<td>100</td>
<td>40</td>
<td>6.6</td>
<td>24.6</td>
<td>8.0</td>
<td>21.5</td>
<td>14.5</td>
<td>22.2</td>
<td>12.8</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>10</td>
<td>6.2</td>
<td>25.4</td>
<td>10.1</td>
<td>20.4</td>
<td>14.8</td>
<td>20.6</td>
<td>13.3</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>20</td>
<td>6.1</td>
<td>25.2</td>
<td>7.5</td>
<td>20.1</td>
<td>14.9</td>
<td>21.6</td>
<td>12.7</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>30</td>
<td>6.0</td>
<td>24.6</td>
<td>8.1</td>
<td>20.6</td>
<td>14.3</td>
<td>21.0</td>
<td>12.8</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>40</td>
<td>6.5</td>
<td>25.4</td>
<td>7.7</td>
<td>21.0</td>
<td>14.2</td>
<td>20.7</td>
<td>12.5</td>
</tr>
<tr>
<td>600</td>
<td>50</td>
<td>10</td>
<td>6.4</td>
<td>27.4</td>
<td>7.4</td>
<td>24.0</td>
<td>14.8</td>
<td>24.0</td>
<td>13.6</td>
</tr>
<tr>
<td>600</td>
<td>50</td>
<td>20</td>
<td>7.1</td>
<td>24.9</td>
<td>9.8</td>
<td>21.1</td>
<td>15.2</td>
<td>20.3</td>
<td>12.7</td>
</tr>
<tr>
<td>600</td>
<td>50</td>
<td>30</td>
<td>6.3</td>
<td>23.7</td>
<td>8.3</td>
<td>21.0</td>
<td>13.3</td>
<td>21.6</td>
<td>13.3</td>
</tr>
<tr>
<td>600</td>
<td>50</td>
<td>40</td>
<td>7.3</td>
<td>23.7</td>
<td>9.4</td>
<td>20.7</td>
<td>13.3</td>
<td>21.8</td>
<td>13.9</td>
</tr>
<tr>
<td>600</td>
<td>100</td>
<td>10</td>
<td>7.0</td>
<td>25.3</td>
<td>9.0</td>
<td>22.0</td>
<td>15.2</td>
<td>20.7</td>
<td>15.3</td>
</tr>
<tr>
<td>600</td>
<td>100</td>
<td>20</td>
<td>6.5</td>
<td>23.9</td>
<td>8.0</td>
<td>20.8</td>
<td>14.5</td>
<td>21.5</td>
<td>12.5</td>
</tr>
<tr>
<td>600</td>
<td>100</td>
<td>30</td>
<td>7.9</td>
<td>25.8</td>
<td>8.7</td>
<td>22.1</td>
<td>13.5</td>
<td>18.8</td>
<td>13.5</td>
</tr>
<tr>
<td>600</td>
<td>100</td>
<td>40</td>
<td>7.1</td>
<td>24.2</td>
<td>10.0</td>
<td>21.6</td>
<td>12.8</td>
<td>21.0</td>
<td>12.2</td>
</tr>
<tr>
<td>600</td>
<td>150</td>
<td>10</td>
<td>6.6</td>
<td>25.1</td>
<td>8.8</td>
<td>20.6</td>
<td>14.8</td>
<td>22.2</td>
<td>15.7</td>
</tr>
<tr>
<td>600</td>
<td>150</td>
<td>20</td>
<td>7.7</td>
<td>23.9</td>
<td>9.0</td>
<td>22.4</td>
<td>14.3</td>
<td>21.4</td>
<td>12.4</td>
</tr>
<tr>
<td>600</td>
<td>150</td>
<td>30</td>
<td>7.2</td>
<td>23.7</td>
<td>9.2</td>
<td>20.8</td>
<td>13.0</td>
<td>20.7</td>
<td>14.6</td>
</tr>
<tr>
<td>600</td>
<td>150</td>
<td>40</td>
<td>7.6</td>
<td>23.9</td>
<td>9.1</td>
<td>21.9</td>
<td>14.1</td>
<td>22.9</td>
<td>13.3</td>
</tr>
<tr>
<td>900</td>
<td>50</td>
<td>10</td>
<td>7.2</td>
<td>29.2</td>
<td>7.7</td>
<td>24.8</td>
<td>15.5</td>
<td>18.4</td>
<td>15.3</td>
</tr>
<tr>
<td>900</td>
<td>50</td>
<td>20</td>
<td>6.8</td>
<td>24.5</td>
<td>9.2</td>
<td>21.6</td>
<td>13.9</td>
<td>22.1</td>
<td>13.3</td>
</tr>
<tr>
<td>900</td>
<td>50</td>
<td>30</td>
<td>8.1</td>
<td>23.9</td>
<td>10.1</td>
<td>19.9</td>
<td>14.2</td>
<td>22.5</td>
<td>14.7</td>
</tr>
<tr>
<td>900</td>
<td>50</td>
<td>40</td>
<td>9.0</td>
<td>24.0</td>
<td>11.0</td>
<td>23.7</td>
<td>15.3</td>
<td>21.5</td>
<td>16.0</td>
</tr>
<tr>
<td>900</td>
<td>100</td>
<td>10</td>
<td>6.0</td>
<td>29.7</td>
<td>9.8</td>
<td>24.3</td>
<td>14.9</td>
<td>17.3</td>
<td>13.2</td>
</tr>
<tr>
<td>900</td>
<td>100</td>
<td>20</td>
<td>6.7</td>
<td>23.5</td>
<td>8.6</td>
<td>18.4</td>
<td>13.8</td>
<td>23.9</td>
<td>13.6</td>
</tr>
<tr>
<td>900</td>
<td>100</td>
<td>30</td>
<td>8.6</td>
<td>25.5</td>
<td>10.1</td>
<td>20.7</td>
<td>14.5</td>
<td>19.1</td>
<td>13.6</td>
</tr>
<tr>
<td>900</td>
<td>100</td>
<td>40</td>
<td>7.0</td>
<td>23.1</td>
<td>8.7</td>
<td>20.1</td>
<td>15.2</td>
<td>23.4</td>
<td>15.1</td>
</tr>
<tr>
<td>900</td>
<td>150</td>
<td>10</td>
<td>7.6</td>
<td>28.0</td>
<td>12.0</td>
<td>30.0</td>
<td>18.1</td>
<td>33.0</td>
<td>20.7</td>
</tr>
<tr>
<td>900</td>
<td>150</td>
<td>20</td>
<td>26.0</td>
<td>34.9</td>
<td>44.2</td>
<td>34.5</td>
<td>102.2</td>
<td>55.2</td>
<td>105.4</td>
</tr>
<tr>
<td>900</td>
<td>150</td>
<td>30</td>
<td>15.1</td>
<td>29.7</td>
<td>24.0</td>
<td>30.5</td>
<td>62.3</td>
<td>31.7</td>
<td>64.0</td>
</tr>
<tr>
<td>900</td>
<td>150</td>
<td>40</td>
<td>12.4</td>
<td>37.7</td>
<td>16.2</td>
<td>33.3</td>
<td>28.0</td>
<td>41.7</td>
<td>27.7</td>
</tr>
</tbody>
</table>

Table 30. Displayed in this table are the results for all thirty-six TDNNs created for subject 1. The top of the column displays the type of movement for the given trial and the two corresponding columns underneath the type of movement are the errors occurring for each DOF. Errors are listed in degrees.
Table 31. Displayed here is a summarized table of results for the first subject while varying the time delay and time interval. Errors are listed in degrees.

<table>
<thead>
<tr>
<th>Time Delay (ms)</th>
<th>Time Interval (ms)</th>
<th>Slow Elbow Movement</th>
<th>Fast Elbow Movement</th>
<th>Slow Shoulder Movement</th>
<th>Fast Shoulder Movement</th>
<th>Slow Reaching Movement</th>
<th>Fast Reaching Movement</th>
<th>Average Error</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Shoulder Error</td>
<td>Elbow Error</td>
<td>Shoulder Error</td>
<td>Elbow Error</td>
<td>Shoulder Error</td>
<td>Elbow Error</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>50</td>
<td>6.3</td>
<td>26.3</td>
<td>7.5</td>
<td>22.7</td>
<td>14.9</td>
<td>21.7</td>
<td>13.0</td>
</tr>
<tr>
<td>300</td>
<td>100</td>
<td>6.0</td>
<td>25.9</td>
<td>7.5</td>
<td>21.7</td>
<td>16.4</td>
<td>20.4</td>
<td>15.2</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>6.2</td>
<td>25.4</td>
<td>10.1</td>
<td>20.4</td>
<td>14.8</td>
<td>20.6</td>
<td>13.3</td>
</tr>
<tr>
<td>600</td>
<td>50</td>
<td>6.4</td>
<td>27.4</td>
<td>7.4</td>
<td>24.0</td>
<td>14.8</td>
<td>24.0</td>
<td>13.6</td>
</tr>
<tr>
<td>600</td>
<td>100</td>
<td>7.0</td>
<td>25.3</td>
<td>9.0</td>
<td>22.0</td>
<td>15.2</td>
<td>20.7</td>
<td>15.3</td>
</tr>
<tr>
<td>600</td>
<td>150</td>
<td>6.6</td>
<td>25.1</td>
<td>8.8</td>
<td>20.6</td>
<td>14.8</td>
<td>22.2</td>
<td>15.7</td>
</tr>
<tr>
<td>900</td>
<td>50</td>
<td>7.2</td>
<td>29.2</td>
<td>7.7</td>
<td>24.8</td>
<td>15.5</td>
<td>18.4</td>
<td>15.3</td>
</tr>
<tr>
<td>900</td>
<td>100</td>
<td>6.0</td>
<td>29.7</td>
<td>9.8</td>
<td>24.3</td>
<td>14.9</td>
<td>17.3</td>
<td>13.2</td>
</tr>
<tr>
<td>900</td>
<td>150</td>
<td>7.6</td>
<td>28.0</td>
<td>11.2</td>
<td>30.0</td>
<td>18.1</td>
<td>33.0</td>
<td>20.7</td>
</tr>
</tbody>
</table>

Table 32. Displayed here is the average error for the shoulder and elbow DOF for each type of movement using the optimal TDNN parameters which were a total delay of 300ms, delay interval of 100ms, and 10 neurons in the hidden layer. Errors are in degrees.

<table>
<thead>
<tr>
<th>Shoulder Error (Degrees)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Subject</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elbow Error (Degrees)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Subject</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

Table 33. Displayed here is the average error of the shoulder and elbow position combined for all movements using the optimal TDNN parameters which were a total delay of 300ms, delay interval of 100ms, and 10 neurons in the hidden layer. Errors are in degrees.

<table>
<thead>
<tr>
<th>Test Subject</th>
<th>Average Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15.7±22.9</td>
</tr>
<tr>
<td>2</td>
<td>17.0±23.2</td>
</tr>
<tr>
<td>3</td>
<td>23.6±30.6</td>
</tr>
<tr>
<td>4</td>
<td>21.5±31.4</td>
</tr>
<tr>
<td>5</td>
<td>17.4±25.3</td>
</tr>
</tbody>
</table>
Figure 43. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 1 performing slow elbow movement.

Figure 44. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 1 performing fast elbow movement.
Figure 45. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 1 performing slow shoulder movement.

Figure 46. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 1 performing fast shoulder movement.
Figure 47. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 1 performing slow reaching movement.

Figure 48. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 1 performing fast reaching movement.
Figure 49. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 2 performing slow elbow movement.

Figure 50. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 2 performing fast elbow movement.
Figure 51. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 2 performing slow shoulder movement.

Figure 52. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 2 performing fast shoulder movement.
Figure 53. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 2 performing slow reaching movement.

Figure 54. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 2 performing fast reaching movement.
Figure 55. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 3 performing slow elbow movement.

Figure 56. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 3 performing fast elbow movement.
Figure 57. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 3 performing slow shoulder movement.

Figure 58. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 3 performing fast shoulder movement.
Figure 59. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 3 performing slow reaching movement.

Figure 60. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 3 performing fast reaching movement.
Figure 61. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 4 performing slow elbow movement.

Figure 62. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 4 performing fast elbow movement.
Figure 63. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 4 performing slow shoulder movement.

Figure 64. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 4 performing fast shoulder movement.
Figure 65. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 4 performing slow reaching movement.

Figure 66. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 4 performing fast reaching movement.
Figure 67. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 5 performing slow elbow movement.

Figure 68. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 5 performing fast elbow movement.
Figure 69. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 5 performing slow shoulder movement.

Figure 70. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 5 performing fast shoulder movement.
Figure 71. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 5 performing slow reaching movement.

Figure 72. Displayed here is the output of the TDNN having a 300ms delay, 100ms delay intervals, and 10 neurons in the hidden layer for subject 5 performing fast reaching movement.
Figure 73. Displayed here are the results of using a TDNN to predict the elbow position of a subject with CCS while using the biceps and the triceps.
Chapter 5: Discussion

5.1 Real Time Pattern Recognition Myoelectric Control Scheme Discussion

The analysis of the real time pattern recognition myoelectric control scheme is presented in three different sections. The first section discusses the offline window and feature analysis. Although this first section does not discuss real time results, this analysis was involved in the design and development of the real time system. The second section discusses the results of the real time myoelectric control scheme with healthy subjects while the third section discusses the real time myoelectric control scheme with the subject that had CCS.

5.1.1 Offline Window and Feature Analysis

The first part of this work focused on identifying what EMG feature and windowing scheme to use in the development of the real time myoelectric control scheme. This was accomplished by completing a thorough analysis of both features and windowing schemes. Time domain features of MAV, MAVS, SSC, ZC and WL were tested as well as AR features which are considered to be frequency domain features. The time domain features were tested individually and as a set. Adjacent windowing schemes utilizing window lengths of 50ms, 100ms, and 250ms were tested in addition to a majority voting scheme using 50ms windows with a total window length of 250ms. The goal of testing all of the possible features and windows was to find the optimal windowing and feature scheme which would be implemented in real time.

The first observation made from the offline analysis results compiled in the previous chapter, was that an increase in classification accuracy occurred when the window size increased for the adjacent windowing schemes. A look at the overall classification accuracies for each test subject showed an increase in accuracy when the window increased from 50ms to 100ms and 100ms to 250ms independent of the feature used except for the MAVS feature. Specifically for all ten subjects, increasing the window size from 50ms to 100ms and from 100ms to 250ms resulted in an increase of classification accuracy in
every instance when using SSC, WL, one AR feature, three AR features, and four AR features. All ten
subjects also had an increase in accuracy when increasing from 50ms to 100ms when two AR coefficients
were used as features. For nine out of the ten test subjects, the classification accuracy improved when
increasing the window size from 50ms to 100ms for MAV and the TD set as well as when increasing the
window size from 100ms to 250ms for MAV and two AR features. When increasing the window size
from 100ms to 250ms, the classification accuracy only increased for five subjects when using the TD set.
The smaller amount of improved classification accuracies for the TD set could possibly be explained by
the poor performance of the MAVS feature. Using the MAVS feature actually resulted in a decrease in
classification accuracy for 9 out of 10 subjects when increasing the classification accuracy from 50ms to
100ms and 100ms to 250ms. Because of the terrible performance of the MAVS feature, the TD set could
have been affected by the trend of the MAVS feature. The MAVS feature is included in the TD set which
is why this possibility exists. The MAVS feature was by far the worst feature and will be discussed
further in the following paragraphs. Other than the MAVS feature, there was an overwhelming amount of
evidence that increasing the window size resulted in a higher classification accuracy and that the 250ms
window was the best choice when using an adjacent windowing scheme. This is the most apparent trend
from the offline data analysis and is also supported in the literature [17], [18]. One may think that one
should explore window sizes beyond 250ms, but an increase beyond 250ms would result in window sizes
that are not feasible to use in a real time myoelectric control scheme [17]. Only windows that would result
in an acceptable delay in a real time myoelectric control scheme were considered in this study.

The next analysis completed was whether or not the majority voting scheme could add any
improvements in classification performance. Table 16 shows the comparison of the adjacent windowing
scheme using 250ms with the majority voting scheme of 250ms and a smaller window of 50ms for both
the TD features and using all four AR features. It can be seen from Table 16 that the average
classification accuracy using AR features without majority voting was 94.13% and is 89.60% when using
it. Only two out of ten subjects had improved classification accuracy when using the majority voting
scheme with AR features. Although the overall accuracy decreased when using the majority voting
scheme for AR features, there was a slight increase in accuracy for the TD feature set of 86.09% to 86.97%. Six out of ten subjects had improved classification accuracy when using the TD feature set with majority voting. The AR features performed significantly better than the TD features as seen in Table 15 and Table 16 and the majority voting did not offer any improvements in classification accuracy so the adjacent windowing scheme with a 250ms window was chosen to be used in the real time myoelectric control scheme. If one were to implement a real time myoelectric control scheme using TD features, then the majority voting scheme should be considered. This work chose not to use TD features in the real time control scheme because they did not outperform the AR features.

The analysis of specific features should start with Table 15. This table only included classification accuracies using 250ms windows in an adjacent windowing scheme. It showed the average classification accuracies across all classes for each of the feature schemes implemented. The best classification accuracy for nine of the ten subjects occurred when using three AR features and the remaining subject had the best classification accuracy when using four AR features. The second best classification accuracy came when using two or four AR features for the nine subjects who had the best performance when using three AR features while the second best classification accuracy for the remaining subject was using three AR features. The highest average classification accuracies across all subjects was 94.92%, 94.13%, 93.46%, and 89.75% when using three AR features, four AR features, two AR features, and one AR feature, respectively. The gain of using two AR features rather than one gives an average increase in accuracy of 3.71% whereas the gain of adding a third AR feature is only 1.46%. Adding the fourth and last possible AR coefficient (fourth order model) actually gives a slight decrease in the average classification accuracy for all subjects. Using the AR features provides the best classification accuracy relative to any of the TD features without a doubt. And whether one chooses to use all or some of the AR coefficients as features, the AR features still outperform the TD features. As a result the real time myoelectric scheme used the 250ms adjacent windowing scheme with AR features.

The best performing TD feature was the WL feature. Comparing only the TD features amongst each other, the best performing feature for all ten subjects was WL. The average classification accuracy
across all subjects using WL was 89.74% which is one hundredth of a percent under the average classification accuracy of using one AR coefficient as a feature. The large success of the WL feature among possible time domain features was also seen by Oskoei [47]. The rest of the individual TD features performed much worse than WL. The rest of the individual TD features had 73.14%, 58.83%, 52.31%, and 28.57% accuracy for ZC, MAV, SSC, and MAVS respectively. After wavelength, the best TD feature was MAV and SSC which were second or third most accurate for all ten subjects. For all ten subjects the worst classification accuracy occurred when using the MAVS feature. The poor performance of the MAVS feature could be the reason that Englehart did not use it while he adopted the four other TD features from Hudgin’s earlier work [18]. Nine out of the ten subjects achieved a better classification accuracy using the WL feature alone in comparison to the TD set which includes the WL. This could be a result of the extremely poor performance of the MAVS feature and its effect on the TD set. It would seem logical that the TD set with the MAVS feature removed would perform with higher classification accuracy than the current results. Although the work of the real time development proceeded with the AR features, it would be recommended to use the WL feature if one were to limit their use to only time domain features.

It is valuable to investigate what classes were able to be classified accurately and what classes had poor classification accuracy. In order to investigate this, each individual subject’s classification accuracies were analyzed for the 250ms adjacent windowing scheme using all four AR coefficients. Flexion had the highest classification accuracy for five out of the ten subjects, while rest, grasping, releasing, pronation, and extension each had the highest classification accuracy for one test subject. Flexion having the majority of highest classification accuracies for five subjects could be the result of the biceps brachii and brachialis contributing to only one class of movement as the prime mover. In addition, the only muscles that are recorded on the biceps electrode are the biceps and brachialis. This results in a significant difference of features from the biceps EMG channel when performing flexion versus any of the other classes. One might expect to find the same result for the extension class because the triceps is the prime mover for extension and is the only muscle picked up by the triceps electrode. But this does not
occur, and it is unknown why. In fact, the extension class had the worst classification accuracy of all
classes for four subjects and had the second to worst classification accuracy for another subject. Extension
had the highest number of times that it had the most inaccurate classification for four subjects and the
releasing class had the second highest with three subjects.

5.1.2 Real Time Pattern Recognition Myoelectric Control Scheme for Healthy Subjects

The majority of myoelectric pattern classification work in literature has extremely tight
constraints on their data collection. Many researchers use a device to hold the arm completely stationary
while recording isometric EMG data. Although these constraints allow for a higher rate of classification,
they cannot be recreated in a real time system for a prosthetic or rehabilitation robot application. Much of
the work in the literature only analyzes EMG features or a classifier in an offline analysis and do not
extend their analysis beyond offline studies. An essential piece of evaluation for control schemes is left
out when only an offline analysis is performed. Offline studies do not incorporate any visual feedback for
the user. Feedback for control systems is essential, especially those that interface with humans. In
noninvasive myoelectric control systems the only feedback available is visual feedback from the device
being controlled. Without this feedback an analysis of a designed myoelectric control scheme is
incomplete. The work presented here incorporated the data collection environment used for offline tests
into the real time system which also incorporated necessary user feedback.

The offline analysis discussed in the previous section led to the design and development of the
real time myoelectric control scheme that used an adjacent windowing scheme of 250ms and two AR
coefficients from each EMG channel as features. One might note that in the offline results using three AR
features resulted in a slight increase in classification accuracy compared to using two AR features. In the
development stages of the real time myoelectric control system when more than two AR coefficients were
used from each channel, the classification accuracy was significantly reduced to a point where the control
scheme was not even feasible to use. It is known in the literature (and was seen in preliminary work
performed prior to the work presented here) that as the dimensionality increases, the accuracy of a LDA
decreases. It is also possible that the accuracy of the probabilities that came from the PDFs was decreased in Labview due to using higher dimensions when more than two AR coefficients from each channel were used. From the offline results, the overall classification accuracy difference between using two or three AR coefficients was only 1.46%, which is not very significant relative to the drop in accuracy if any other featuring scheme was chosen to be implemented in real time.

After training the system with EMG data for each class, the subjects were asked to practice controlling the robotic arm. This allowed the subjects to see what classes were easily able to be classified in addition to which classes or movements they struggled at controlling. This feedback allowed users to identify the classes that they needed to practice controlling. The visual feedback allowed them to adjust the way that they were contracting in order to adjust the output of the LDA classifier and controller. Several times the subject realized that they struggled with one or two classes that were consistently classified incorrectly. In these cases, more training data was collected and the LDA was retrained with the new data. Once the subjects felt comfortable controlling the robotic arm, they were tested with the three sets of randomized commands.

The average classification rate for all of the healthy subjects was 89.52% as seen in Table 27. Male subject four had the highest classification accuracy with 96.83%. This corresponded to two misclassifications out of the sixty-three total commands. He performed the first two sets of commands at 100% accuracy while the last set of commands had the two misclassifications as seen in Table 25. The worst classification occurred for female subject two who had a classification accuracy of 74.60%. This subject noted that she became fatigued which is seen in the drop of her classification accuracy after the first set. Her results for the three sets progressed as 85.71%, 66.67%, and 71.43% respectively and can be seen in Table 18. A look at her misclassifications for the last two sets showed that the majority of misclassifications were occurring for the pronation, supination, and releasing classes. The majority of the outputs from the controller during these misclassifications were extension which meant that her fatigue was causing the triceps to contract. This resulted in extension being output from the LDA classifier. Looking at the average classification accuracy across all subjects for each set showed that the fatigue
experienced by female subject two was an anomaly. If fatigue would have had an effect on all of the subjects, there would be a drop in accuracy from set one to set two and from set two to set three. This drop in classification does not exist because the average classification accuracies for each set were 88.57%, 89.52%, and 90.48% respectively. If one were to exclude female subject two from the results because of her fatigue, the average classification accuracy for the real time system would be 91.18%.

Beyond the specific classification accuracies of the individual subjects, valuable information is gained by looking at the general trends of misclassifications that occurred across all of the subjects. Figure 35 shows the percentage of each class that was misclassified out of the total number of misclassifications that occurred for all subjects. This figure enables one to see the number of misclassifications that occurred for one class relative to the other classes. This shows that releasing was the class most often misclassified at 36% of the total misclassifications while pronation is second most at 23%. These two classes contribute to a near 60% of all misclassifications that occurred. It is logical that the two highest classes of misclassification are derived from the contraction of forearm muscles because the majority of classes in this analysis occur from muscle contractions in the forearm. It also shows that flexion had zero misclassifications which corresponded to 100% accuracy. The rest of the classes made up 14%, 12%, 11%, and 4% of the total misclassifications which corresponded to grasping, extension, supination, and rest respectively.

It is also helpful to look at the controller output during the misclassifications of each command for all subjects. Figures 36 to 41 show this analysis. Figure 38 shows the breakdown of misclassifications for releasing which was the class misclassified the most. When subjects were asked to perform the releasing class, pronation made up 47% of the incorrect outputs when misclassification occurred. This shows that the pronation class had a similar EMG pattern to the releasing class, at least from the EMG electrodes point of view. The high percentage of pronation is not surprising because pronation had the second highest occurrence of misclassifications discussed in the paragraph above. This again gives evidence to the idea that the EMG when contracting for releasing and pronation was similar. One could attempt to reduce the number of releasing and pronation misclassifications by asking test subjects to make
a clear distinction between the way they contract for releasing and pronation. Another way of attempting to reduce the number of releasing and pronation misclassifications would be to position the electrodes at specific muscles responsible for releasing and pronation. This could allow for the features extracted from each electrode channel to be more distinguishable for both classes. Before making a permanent change to the electrode locations, one would have to test and see if the movement of electrodes would have a negative effect on the performance of other classes.

Another observation one should note from Figures 36 to 41 is the minimal occurrence of flexion as an output during the misclassification of other classes. This only occurred once which was during the rest command. This piece of information along with the fact that flexion was never misclassified shows that the flexion class was the most distinguishable class out of all the classes. A general occurrence among misclassifications that occurred for most of the classes was an output that changed between the correct class and another class. This occurred for a large portion of the misclassifications for grasping, pronation, supination, and extension. Rest was only misclassified a total of three times which makes it hard to make any conclusions about misclassifications of the rest class. Supination and extension only had a total of seven and eight misclassifications respectively which also makes it difficult to make conclusions about these classes. A larger set of subjects or more data from the subjects used would need to be obtained in order to make more conclusive statements about the misclassifications occurring while attempting to perform the rest, supination, and extension classes.

5.1.3 Real Time Pattern Recognition Myoelectric Control Scheme for a CCS Subject

A common drawback to most of the work published in the area of myoelectric pattern recognition is the lack of incorporating an end user into the work. The work presented in the offline study and the real time myoelectric control scheme with healthy subjects formed a baseline myoelectric control scheme which could be used for a person with CCS. To the author’s knowledge no attempts in the literature have been made to incorporate a myoelectric control scheme for a subject with CCS. Due to the lack of prior work incorporating CCS subjects into a myoelectric control scheme, there were many unknowns heading
into the work for the CCS subject. The testing of the subject with CCS took place over several visitations to the BLL at RIT on account of necessary adaptations that needed to be made to the developed myoelectric control scheme.

There was much uncertainty surrounding the CCS subject’s ability to control all three DOF using the developed myoelectric control scheme for the healthy subjects. Because of this uncertainty, three less complex controllers that controlled one DOF each were created. The three single DOF controllers corresponded to the same DOF from the three DOF controller. The single DOF controllers also used the same theoretical analysis methods and algorithm that was used for the full three DOF controller. The goal was to first establish a series of successful trials for the subject using the simpler controllers. This would allow the subject to gain confidence in the control schemes and his ability to control the robotic arm. Testing of the single DOF controllers also gave information to the author about the ability of the CCS subject to use the myoelectric control scheme.

Each of the single DOF myoelectric controllers were tested separately. Two of the single DOF controllers completed testing with little to no complications. The first controller that was tested with ease was the controller for the elbow joint. This controller used the biceps and triceps EMG as inputs to the myoelectric control scheme. With the first set of training data, the CCS subject was able to control all three classes, flexion, extension, and rest. The CCS subject was able to control this controller with 100% accuracy as shown in Table 28. The second controller that was tested with no complications was the hand controller. This controller used the four forearm EMG channel locations that were used for the healthy subjects. The subject was able to control the three classes, grasping, releasing, and rest with the initial set of training data. The subject achieved 96.30% classification accuracy as seen in Table 28.

The third single DOF controller tested was the wrist controller. It was initially planned to use four EMG channels in the same forearm locations that were used for the healthy subjects. Unfortunately, the CCS subject was unable to have any success controlling the wrist DOF. Several attempts were made at recollecting data to train with, but the subject still had no success. The subject was then analyzed while he was pronating and supinating to see what muscles were contracting the most. It was noticed after
analyzing his upper limb during pronation and supination that he was using other muscles in addition to
his forearm muscles for pronation and supination. It was found that his biceps was contracting during
both pronation and supination. As a result, one of the forearm channels was moved to the biceps which
left only three electrode channels surrounding the circumference of the forearm. By making this change
the subject was able to successfully use the controller and he achieved an 85.19% classification accuracy
which can be seen in Table 28.

After testing the single DOF controllers and being successful at controlling each one, the full
three DOF controller was tested with the CCS subject. Similar to the difficulty experienced when trying
to control the wrist controller, the CCS subject had great difficulty controlling all of the classes. After
several attempts were made at retraining the controller with new data, no level of success was attained.
Just like the process that was followed to adapt the wrist controller, the subject was analyzed to see
exactly what muscles were contributing to the different classes. After reanalyzing the way that the subject
was contracting his muscles during the different motion classes, it showed again that he was using
compensatory muscles from the shoulder. It was then decided to use only two of the four forearm
channels. The two forearm channels were placed on the flexors and extensors and the remaining two
channels were moved to the shoulder on the deltoid and the trapezius. The final choice of EMG channel
locations can be seen in Figure 74. Once these locations were found, the subject was tested using the three
sets of randomized commands.

![Image of EMG locations](image)

**Figure 74.** Displayed here are the final EMG locations chosen for the three DOF controller for the CCS subject. There are electrode channels on the deltoid, trapezius, triceps, biceps, wrist extensors, and wrist flexors.
As seen from Table 29, the CCS subject had an overall classification accuracy of 68.25% for the three DOF controller which is significantly lower than the classification rates for the single DOF controllers as well as the results for the healthy subjects. This can be contributed to a number of factors. The first factor is the higher number of commands in each test set compared to the individual controllers. The individual controllers only had to test for three classes where the three DOF controller had to test for seven classes. Since each class had three commands in each test set, the testing of the three DOF controller took significantly longer. This resulted in spasticity setting into the subject’s arm muscles as mentioned in the background. This caused involuntary contractions of his muscles which resulted in the subject not being able to control his muscles and EMG. The second factor is due to the use of compensatory muscles for some of the classes. For instance the biceps was used to compensate for a lack of wrist strength during wrist pronation and supination. This was shown to be the case when testing for the wrist single DOF controller. As seen from Figure 42, there was a large amount of misclassifications for the flexion class. The large amount of misclassifications occurred because of the fact that the subject contracted his biceps during wrist supination. Because his biceps contracted during supination, the flexion class was often misclassified as supination. This is in strict contrast to the results for the healthy subjects where all ten subjects had 100% classification accuracy with flexion. This difference shows the importance of needing to test actual subjects for which the end myoelectric control scheme will be applied to.

Important to consider is the actual application of a rehabilitation robotic device, orthotic, or exoskeleton using this control scheme for a subject with CCS. Much of this discussion stems from a qualitative analysis of the experiences occurred during testing of the CCS subject. It would seem that it is possible to use this pattern recognition scheme in single DOF applications for rehabilitation exercises such as those used in physical therapy for a single joint. The high classification results that occurred using the single DOF controllers for the CCS subject point to the feasibility of using an end application that focuses on one DOF. For instance, the 100% accuracy achieved for the elbow shows that the control scheme could be used in a rehabilitation scenario where an assistive device is used to extend the range
and mobility of the elbow. Although the other two single DOF controllers did not have perfect accuracy, classification accuracies were high enough that one could imagine a similar rehabilitation device that could enhance the mobility and range of the wrist and hand. For instance, the CCS subject compensated wrist pronation and supination by using his biceps. It would be feasible to use a pattern recognition scheme to recognize the subject’s intent for pronation and supination and use this information to assist the subject in his motion. This would increase his ability to perform pronation or supination without using compensatory muscles. It should be noted that such devices would need to consider the safety and health of the subject at all costs. This is especially true if the subject is directly interfaced to a robotic device. Because of the low classification using the three DOF controller, it does not appear that this myoelectric control scheme could work for a robotic device that uses multiple DOF. In addition, it does not appear that a permanent assistive device designed for daily use could be used because the myoelectric control scheme does not account for any load. Another reason a device for daily use which uses this myoelectric control scheme would not be recommended for a CCS subject is due to the spasticity that set into the subject’s muscles. This caused some classifications to be made without the intent of the subject. If this were to happen in daily life, an exoskeleton could implement movements without the intent of the subject. Also, additional tests would have to be performed on a population of CCS patients for a more thorough assessment.

5.2 Myoelectric Control Scheme Based on a Time Delayed Neural Network

One of the main goals of this work was to study the applicability of work completed earlier by Au and Kirsch as a possible myoelectric control scheme for a rehabilitation robot application. The feasibility of this was investigated by primarily analyzing the effect of decreasing the total delay of the TDNN as well as by varying the delay interval and number of neurons in the hidden layer. Au and Kirsch had optimal values of 875ms and 125ms for their total delay and delay interval respectively. That amount of delay is not possible for a real time myoelectric control scheme. A thorough analysis of data obtained
from the first subject was used to determine the optimal parameters for the TDNN and then those optimal parameters were tested on the data from the four remaining subjects.

From subject one it was found that the number of neurons in the hidden layer did not have an effect on the accuracy of the TDNN which can be seen in Table 30. As a result it was decided that a hidden layer of ten should be chosen as the optimal number of neurons because the other larger number of neurons tested would take up more resources in the end application. In order to view the results in a manner easy to analyze, Table 31 was created. This simplified version of the results for subject one shows only the results of the TDNNs which had ten neurons in the hidden layer. It allows one to easily view the accuracy while the total delay and the delay interval were varied. This table shows that the most accurate TDNNs occurred for a TDNN of 300ms followed by 600ms and 900ms respectively. When using a 600ms total delay the error in the TDNN increased about 1° independent of what delay interval was used. For the 900ms total delay, the error of the TDNN increased by 1°, 2°, and 7° from the 300ms total delay for the 50ms, 100ms, and 150ms delay intervals respectively. The error of the TDNNs was only slightly different for the total delays of 300ms and 600ms but much worse for the 900ms total delay. This was a little surprising because the optimal delay for the previous work was 875ms which is very close to 900ms which had the worst results in this work. Because there was a slight gain in accuracy when using the 300ms total delay relative to the 600ms total delay and the obvious advantage of using a smaller delay, the optimal delay chosen for this work was 300ms. The accuracy of the TDNN when using a 300ms delay with a 100ms delay interval or a 150ms delay interval resulted in an average error of 15.7°, while the 50ms interval had an average error of 16.0°. In order to move forward with the data of the remaining subjects, a delay interval of 100ms was chosen because using the 100ms interval would allow for one more input to the TDNN relative to the 150ms interval. Therefore, the optimal TDNN parameters as given by the results from subject one were a total delay of 300ms, a time delay interval of 100ms, and a hidden layer of ten neurons. These optimal TDNN parameters were used to test the remaining four subjects’ data. An overall average error of the TDNN for each subject was shown in Table 33. While subject one had the least amount of error with 15.7°, the other subjects had similar results. Subject two and subject five had
similar errors of 17.0° and 17.4° respectively. Subject four and subject three were slightly worse with average TDNN errors of 21.5° and 23.6° respectively. These errors are similar to the errors reported by Au and Kirsch which were around 20°. A more detailed result for each subject is shown in Table 32 which shows the average error in each DOF for each trial of movement.

It is difficult to gain a full understanding of the TDNN’s ability to predict joint positions from only viewing quantitative data. The TDNN output of each trial of data for each subject was graphed versus the actual joint position in order to view the results qualitatively. Several observations can be taken away from the graphed results. The TDNN has the ability to predict joint position for multiple DOF at once which is shown in Figure 47. This figure shows the results during slow reaching for subject one. A myoelectric control scheme that could control multiple DOF at once is highly desirable and yet is not prevalent in the literature and previous work. This would be a benefit that is not offered by the typical myoelectric control schemes which are based on pattern recognition. Another observation seen from the data is the poor accuracy of the TDNN for an immobile DOF while the other DOF is moving. For instance Figure 52 shows the TDNN output for subject two during fast shoulder movement. While the TDNN was able to track the shoulder position with great accuracy, the ability of the TDNN to track the stationary elbow joint was extremely poor. This was a common occurrence in which the TDNN struggled to maintain stability and accuracy of the shoulder during elbow movement and vice versa. Some trials seemed to show that the TDNN had a great ability to predict joint positions from EMG and yet other trials had bad performances. As an example, Figure 55 displays the results for subject three during slow elbow movement. This TDNN shows absolutely no ability to track the subject’s shoulder position. This probably resulted from a great disparity between the training data of slow elbow position versus the testing data of slow elbow position. Retraining the TDNN with a new set of slow elbow movement data could possibly change these poor results. Another important observation to make from the TDNNs is the fact that they map dynamic and complex reaching motions. Current myoelectric control schemes in the literature use pattern recognition approaches from isometric EMG data which is not as desirable in an end application for a myoelectric control scheme.
The quantitative results showed that decreasing the total delay while maintaining the same level of joint position accuracy was possible. One question that must be answered now is what the actual myoelectric control scheme would look like using a TDNN. Although the TDNN has a 300ms delay associated with it, the actual myoelectric control scheme would have a slightly higher delay due to the processing time needed to calculate the TDNN output when given the EMG inputs. The average time needed for the TDNN in the MATLAB environment was 14ms. Because of this processing delay, a continuous output from the TDNN would not be possible. If a continuous stream of output was attempted, a continuous increase in delay would occur because the TDNN could never process the inputs fast enough. It would therefore be recommended to give inputs to the TDNN every 15 to 20ms which would adjust for the amount of time needed for the TDNN to process its output. Something else to consider is filtering the output of the TDNN with a low pass filter. Looking at the qualitative results shows that the TDNN output jumps erratically instead of maintaining a smooth output which is desired. A low pass filter would smooth the output and result in a more user friendly control scheme. Another note to make for an actual application is the overall precision that the TDNNs have. Across all subjects the average error was 19.0° which could cause problems if there was a need to be very accurate. All of the training data here used EMG data that did not have any load. An everyday assistive device would involve additional loading to the arm. This could significantly change the EMG. In order to account for this, training data with different loads would need to be applied to the TDNN during training. In addition to the TDNNs not accounting for load, one would need to consider the change in EMG signals that could occur for an assistive device taking over the movement of the arm. If a device was assisting a subject’s arm directly, the need for the muscles to contract to move the arm decreases which would change the EMG signal. An investigation into how the TDNN could adapt to account for these changes would need to be completed.
Chapter 6: Conclusion

This work consisted of primarily two different projects or studies of myoelectric control schemes. The first involved the design and development of a real time myoelectric control scheme using a pattern recognition approach. A myoelectric control scheme using pattern recognition from isometric EMG data is the typical technique found in the literature. The design started with a thorough analysis of windowing schemes and features often used in the literature. Ten healthy subjects were used in the offline analysis. Three different DOF were used in this work which included the elbow, wrist, and hand. Seven classes were used as part of the pattern recognition scheme which included elbow flexion and extension, wrist pronation and supination, hand grasping and releasing, and rest. Nowhere in the literature was it found that a myoelectric control scheme combined these three DOF. The offline analysis was used in order to suggest what techniques should be used in the real time myoelectric control scheme. The highest classification accuracy achieved was 94.92% when using a 250ms window and three AR coefficients from a fourth order model. A real time myoelectric control scheme was then developed using the 250ms window and AR features in Labview. The same ten healthy subjects that were used for the offline analysis were tested with the real time myoelectric control scheme. The average classification accuracy across all ten subjects was 89.52%. Once the real time system was tested with healthy subjects, the real time myoelectric control scheme was then extended to a subject with CCS. Rather than immediately testing the full three DOF controller on the CCS subject, three separate single DOF controllers were built and tested. The CCS subject was able to achieve 100.00%, 96.30%, and 85.19% accuracy for the elbow, hand, and wrist controllers respectively. When the CCS subject was tested using the full three DOF controller he was able to achieve an accuracy of 68.25%. Several adaptations needed to be made for the CCS subject because he used compensatory muscles which resulted in changing some of the EMG electrode locations. To the author’s knowledge no attempts have ever been made to design a myoelectric control scheme for a subject with CCS. The work for this myoelectric control scheme project is awaiting acceptance to the 2010 International Conference on Robotics and Automation [74].
The second project included in this work was the development of a myoelectric control scheme based on a TDNN. The goal of this project was to explore a myoelectric control scheme that would overcome two of the limitations of typical pattern recognition myoelectric control schemes. The first being the control of only one DOF at once while the second is using EMG from dynamic and complex reaching motions. Previous work had shown that a TDNN was capable of predicting joint position using a total delay of 875ms and a 125ms interval of delay, but this is not suitable for a real time scheme because of the large amount of delay. Data was collected using an upper extremity motion capture system which recorded joint position from the elbow and the shoulder in the sagittal plane as well as EMG from the triceps, biceps, deltoid, and pectoralis. The primary goal was to investigate the feasibility of a TDNN based myoelectric control scheme by decreasing the total delay. Delays of 300ms, 600ms, and 900ms were tested. Time delay intervals of 50ms, 100ms, and 150ms were also tested as well as the number of hidden layer neurons which were 10, 20, 30 and 40 neurons. TDNNs were created with each of the possible combinations of total delay, delay intervals, and number of hidden layer neurons for the first subject. They were trained to output the joint positions of the elbow and shoulder using inputs from the lowpass filtered EMG signals. Several types of complex motion were used to train the TDNNs which also included different speeds of motion. The most accurate results were achieved when using a total delay of 300ms, a delay interval of 100ms, and 10 neurons in the hidden layer. These parameters were used to create TDNNs for the remaining subjects and the average error across all five subjects was 19.0° when using those parameters. This was similar to the previous work which reported errors around 20°. Although the total delay was able to be decreased, work would need to be implemented on the TDNN to try and stabilize the outputs prior to using them as inputs to an actual control scheme. The work for the TDNN project has been published and presented at the 2009 Engineering in Medicine and Biology Conference [75].
Chapter 7: Future Work

There are several directions that can be pursued for the future work of the real time myoelectric control scheme. Using the developed pattern recognition myoelectric control scheme, one could develop robotic devices for rehabilitation exercises. These rehabilitation robotic devices would use the myoelectric control scheme to assist subjects in movement in order to help strengthen their muscles. It could also be used to extend the range of motion for someone like the CCS subject who did not have full mobility of several motions of the arm. Ultimately it would be beneficial to develop rehabilitation robotic devices that could assist in everyday life for those needing assistance at all times. It would be desirable to extend the real time myoelectric control scheme to subjects with other conditions and disabilities besides CCS. CCS is only one condition of the many neuromuscular disabilities that might benefit from applications involving myoelectric control schemes. Other changes could certainly be attempted in order to make the classification accuracy better. This would include testing of other features, classifiers, and windowing schemes. Another approach one could use to increase the accuracy is adding other electrode locations or trying to target different muscles. One could also try to extend the three DOF to four or five DOF by adding motions of the shoulder. In preliminary work, experiments were completed which attempted to run two classifiers in parallel. If this implementation would work, it would allow for movement in multiple DOF. There was no success at accomplishing this, but in the future the parallel classifier concept could be reattempted.

The most logical next step for the myoelectric control scheme based on the TDNN would be to implement the control scheme in real time on the robotic arm in the BLL. This would take the control scheme beyond a theoretical framework and to an actual evaluation of the control scheme’s implementation. Other options would include making attempts at decreasing the error in the output of the TDNN. This could possibly be achieved by increasing the amount of EMG channels. Another choice would be to use a TDNN for each DOF rather than using one TDNN that has multiple outputs. It is also possible that incorporating other kinematic data such as velocity, acceleration, or force as inputs to the
TDNN would increase its accuracy. Using kinematic data could help the TDNN more accurately map the joint positions. A desirable implementation that would make the TDNN better would be a more stable output. The TDNN outputs seemed to jump erratically and anything that could be done to make the output more stable would make the scheme more user friendly in an actual application. Extending the shoulder beyond the DOF in the sagittal plane to its other two DOF would be another future application.
Bibliography


Appendix A: Subject Physiological Data

<table>
<thead>
<tr>
<th>Test Subject</th>
<th>Female</th>
<th></th>
<th></th>
<th></th>
<th>Male</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>Age</td>
<td>22</td>
<td>21</td>
<td>26</td>
<td>18</td>
<td>23</td>
<td>22</td>
<td>26</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>Weight (lb)</td>
<td>126</td>
<td>150</td>
<td>145</td>
<td>145</td>
<td>120</td>
<td>255</td>
<td>220</td>
<td>290</td>
<td>135</td>
</tr>
<tr>
<td>Height (ft,in)</td>
<td>5'8&quot;</td>
<td>6'0&quot;</td>
<td>5'7&quot;</td>
<td>5'5&quot;</td>
<td>5'5&quot;</td>
<td>5'10&quot;</td>
<td>6'0&quot;</td>
<td>5'11&quot;</td>
<td>5'4&quot;</td>
</tr>
<tr>
<td>Upper Arm Length (in)</td>
<td>13</td>
<td>14</td>
<td>14.5</td>
<td>12</td>
<td>11</td>
<td>14.5</td>
<td>12.5</td>
<td>14</td>
<td>12</td>
</tr>
<tr>
<td>Lower Arm Length (in)</td>
<td>10</td>
<td>11</td>
<td>10</td>
<td>10.5</td>
<td>10</td>
<td>12</td>
<td>12</td>
<td>11</td>
<td>10.5</td>
</tr>
<tr>
<td>Total Arm Length (in)</td>
<td>23</td>
<td>25</td>
<td>24.3</td>
<td>22.5</td>
<td>21</td>
<td>26.5</td>
<td>24.5</td>
<td>25</td>
<td>22.5</td>
</tr>
<tr>
<td>Upper Arm Circumference at rest (in)</td>
<td>10.5</td>
<td>10.75</td>
<td>11</td>
<td>11</td>
<td>9.5</td>
<td>15.5</td>
<td>14</td>
<td>17</td>
<td>10</td>
</tr>
<tr>
<td>Upper Arm Circumference when flexed (in)</td>
<td>11</td>
<td>11.25</td>
<td>11.5</td>
<td>12</td>
<td>11</td>
<td>16.5</td>
<td>17</td>
<td>18.5</td>
<td>10.5</td>
</tr>
<tr>
<td>Exercise Frequency (Times/Week)</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Lift Weights (Y/N)</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Dominant Arm</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
</tr>
<tr>
<td>Arm Used for Experiments</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
<td>Right</td>
</tr>
</tbody>
</table>
Appendix B: Matlab Code for Offline Analysis of Data Segmentation

% This m-file segmented the data into windows of the specified length.
% All of the raw data from each class was visually inspected and saved in
% one large data file named "raw_data.mat". If the BioRadio lost any data
% and resulted in blank portions of data then those portions of data were
% removed.

load raw_data.mat

% Depending on what size of window was being created, the window size was
% uncommented. This file was run three times to create each of the
% different window sizes among the data.
%s = 48; % 50ms windows
%s = 96; % 100ms
%s = 240; % 250ms

% c represented the amount of windows that could be extracted from the
% trial of data being segmented into windows.

% c = floor(length(grasping1)/s);
% The following lines of code break the above trial of data, "grasping1"
% which includes 6 channels of EMG into individual channels of EMG data.
g1_1 = zeros(c,s);
g1_2 = zeros(c,s);
g1_3 = zeros(c,s);
g1_4 = zeros(c,s);
g1_5 = zeros(c,s);
g1_6 = zeros(c,s);
% The following lines of code is a loop which creates the actual windows
% and saves the data into windows of individual EMG channels. The code then
% completes the same segmenting of data for all of 10 of the trials for
% grasping.
for a = 1:c
    for b = 1:s
        g1_1(a,b) = grasping1((a-1)*s+b,1);
g1_2(a,b) = grasping1((a-1)*s+b,2);
g1_3(a,b) = grasping1((a-1)*s+b,3);
g1_4(a,b) = grasping1((a-1)*s+b,4);
g1_5(a,b) = grasping1((a-1)*s+b,5);
g1_6(a,b) = grasping1((a-1)*s+b,6);
    end
end
c = floor(length(grasping2)/s);
g2_1 = zeros(c,s);
g2_2 = zeros(c,s);
g2_3 = zeros(c,s);
g2_4 = zeros(c,s);
g2_5 = zeros(c,s);
g2_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        g2_1(a,b) = grasping2((a-1)*s+b,1);
g2_2(a,b) = grasping2((a-1)*s+b,2);
g2_3(a,b) = grasping2((a-1)*s+b,3);
g2_4(a,b) = grasping2((a-1)*s+b,4);
g2_5(a,b) = grasping2((a-1)*s+b,5);
g2_6(a,b) = grasping2((a-1)*s+b,6);
    end
end
c = floor(length(grasping3)/s);
g3_1 = zeros(c,s);
g3_2 = zeros(c,s);
g3_3 = zeros(c,s);
g3_4 = zeros(c,s);
g3_5 = zeros(c,s);
g3_6 = zeros(c,s);
for a = 1:c
  for b = 1:s
    g3_1(a,b) = grasping3((a-1)*s+b,1);
    g3_2(a,b) = grasping3((a-1)*s+b,2);
    g3_3(a,b) = grasping3((a-1)*s+b,3);
    g3_4(a,b) = grasping3((a-1)*s+b,4);
    g3_5(a,b) = grasping3((a-1)*s+b,5);
    g3_6(a,b) = grasping3((a-1)*s+b,6);
  end
end

c = floor(length(grasping4)/s);
g4_1 = zeros(c,s);
g4_2 = zeros(c,s);
g4_3 = zeros(c,s);
g4_4 = zeros(c,s);
g4_5 = zeros(c,s);
g4_6 = zeros(c,s);
for a = 1:c
  for b = 1:s
    g4_1(a,b) = grasping4((a-1)*s+b,1);
    g4_2(a,b) = grasping4((a-1)*s+b,2);
    g4_3(a,b) = grasping4((a-1)*s+b,3);
    g4_4(a,b) = grasping4((a-1)*s+b,4);
    g4_5(a,b) = grasping4((a-1)*s+b,5);
    g4_6(a,b) = grasping4((a-1)*s+b,6);
  end
end

c = floor(length(grasping5)/s);
g5_1 = zeros(c,s);
g5_2 = zeros(c,s);
g5_3 = zeros(c,s);
g5_4 = zeros(c,s);
g5_5 = zeros(c,s);
g5_6 = zeros(c,s);
for a = 1:c
  for b = 1:s
    g5_1(a,b) = grasping5((a-1)*s+b,1);
    g5_2(a,b) = grasping5((a-1)*s+b,2);
    g5_3(a,b) = grasping5((a-1)*s+b,3);
    g5_4(a,b) = grasping5((a-1)*s+b,4);
    g5_5(a,b) = grasping5((a-1)*s+b,5);
    g5_6(a,b) = grasping5((a-1)*s+b,6);
  end
end

c = floor(length(grasping6)/s);
g6_1 = zeros(c,s);
g6_2 = zeros(c,s);
g6_3 = zeros(c,s);
g6_4 = zeros(c,s);
g6_5 = zeros(c,s);
g6_6 = zeros(c,s);
for a = 1:c
  for b = 1:s
    g6_1(a,b) = grasping6((a-1)*s+b,1);
    g6_2(a,b) = grasping6((a-1)*s+b,2);
    g6_3(a,b) = grasping6((a-1)*s+b,3);
\begin{verbatim}
g6_4(a,b) = grasping6((a-1)*s+b,4);
g6_5(a,b) = grasping6((a-1)*s+b,5);
g6_6(a,b) = grasping6((a-1)*s+b,6);
end
end
c = floor(length(grasping7)/s);
g7_1 = zeros(c,s);
g7_2 = zeros(c,s);
g7_3 = zeros(c,s);
g7_4 = zeros(c,s);
g7_5 = zeros(c,s);
g7_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        g7_1(a,b) = grasping7((a-1)*s+b,1);
g7_2(a,b) = grasping7((a-1)*s+b,2);
g7_3(a,b) = grasping7((a-1)*s+b,3);
g7_4(a,b) = grasping7((a-1)*s+b,4);
g7_5(a,b) = grasping7((a-1)*s+b,5);
g7_6(a,b) = grasping7((a-1)*s+b,6);
    end
end
c = floor(length(grasping8)/s);
g8_1 = zeros(c,s);
g8_2 = zeros(c,s);
g8_3 = zeros(c,s);
g8_4 = zeros(c,s);
g8_5 = zeros(c,s);
g8_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        g8_1(a,b) = grasping8((a-1)*s+b,1);
g8_2(a,b) = grasping8((a-1)*s+b,2);
g8_3(a,b) = grasping8((a-1)*s+b,3);
g8_4(a,b) = grasping8((a-1)*s+b,4);
g8_5(a,b) = grasping8((a-1)*s+b,5);
g8_6(a,b) = grasping8((a-1)*s+b,6);
    end
end
c = floor(length(grasping9)/s);
g9_1 = zeros(c,s);
g9_2 = zeros(c,s);
g9_3 = zeros(c,s);
g9_4 = zeros(c,s);
g9_5 = zeros(c,s);
g9_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        g9_1(a,b) = grasping9((a-1)*s+b,1);
g9_2(a,b) = grasping9((a-1)*s+b,2);
g9_3(a,b) = grasping9((a-1)*s+b,3);
g9_4(a,b) = grasping9((a-1)*s+b,4);
g9_5(a,b) = grasping9((a-1)*s+b,5);
g9_6(a,b) = grasping9((a-1)*s+b,6);
    end
end
c = floor(length(grasping10)/s);
g10_1 = zeros(c,s);
g10_2 = zeros(c,s);
g10_3 = zeros(c,s);
g10_4 = zeros(c,s);
g10_5 = zeros(c,s);
\end{verbatim}
g10_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        g10_1(a,b) = grasping10((a-1)*s+b,1);
        g10_2(a,b) = grasping10((a-1)*s+b,2);
        g10_3(a,b) = grasping10((a-1)*s+b,3);
        g10_4(a,b) = grasping10((a-1)*s+b,4);
        g10_5(a,b) = grasping10((a-1)*s+b,5);
        g10_6(a,b) = grasping10((a-1)*s+b,6);
    end
end

% The code below groups all of the windows for each EMG channel from the
% first 5 trials into one group for training and the last 5 trials were
% grouped for testing.
g_train_1 = [g1_1;g2_1;g3_1;g4_1;g5_1];
g_test_1 = [g6_1;g7_1;g8_1;g9_1;g10_1];
g_train_2 = [g1_2;g2_2;g3_2;g4_2;g5_2];
g_test_2 = [g6_2;g7_2;g8_2;g9_2;g10_2];
g_train_3 = [g1_3;g2_3;g3_3;g4_3;g5_3];
g_test_3 = [g6_3;g7_3;g8_3;g9_3;g10_3];
g_train_4 = [g1_4;g2_4;g3_4;g4_4;g5_4];
g_test_4 = [g6_4;g7_4;g8_4;g9_4;g10_4];
g_train_5 = [g1_5;g2_5;g3_5;g4_5;g5_5];
g_test_5 = [g6_5;g7_5;g8_5;g9_5;g10_5];
g_train_6 = [g1_6;g2_6;g3_6;g4_6;g5_6];
g_test_6 = [g6_6;g7_6;g8_6;g9_6;g10_6];

% The code below is the same code used above except for the releasing class
r1_1 = zeros(c,s);
r1_2 = zeros(c,s);
r1_3 = zeros(c,s);
r1_4 = zeros(c,s);
r1_5 = zeros(c,s);
r1_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        r1_1(a,b) = releasing1((a-1)*s+b,1);
        r1_2(a,b) = releasing1((a-1)*s+b,2);
        r1_3(a,b) = releasing1((a-1)*s+b,3);
        r1_4(a,b) = releasing1((a-1)*s+b,4);
        r1_5(a,b) = releasing1((a-1)*s+b,5);
        r1_6(a,b) = releasing1((a-1)*s+b,6);
    end
end
c = floor(length(releasing2)/s);
r2_1 = zeros(c,s);
r2_2 = zeros(c,s);
r2_3 = zeros(c,s);
r2_4 = zeros(c,s);
r2_5 = zeros(c,s);
r2_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        r2_1(a,b) = releasing2((a-1)*s+b,1);
        r2_2(a,b) = releasing2((a-1)*s+b,2);
        r2_3(a,b) = releasing2((a-1)*s+b,3);
        r2_4(a,b) = releasing2((a-1)*s+b,4);
        r2_5(a,b) = releasing2((a-1)*s+b,5);
        r2_6(a,b) = releasing2((a-1)*s+b,6);
    end
end
c = floor(length(releasing3)/s);
r3_1 = zeros(c,s);
r3_2 = zeros(c,s);
r3_3 = zeros(c,s);
r3_4 = zeros(c,s);
r3_5 = zeros(c,s);
r3_6 = zeros(c,s);
for a = 1:c:
    for b = 1:s:
        r3_1(a,b) = releasing3((a-1)*s+b,1);
        r3_2(a,b) = releasing3((a-1)*s+b,2);
        r3_3(a,b) = releasing3((a-1)*s+b,3);
        r3_4(a,b) = releasing3((a-1)*s+b,4);
        r3_5(a,b) = releasing3((a-1)*s+b,5);
        r3_6(a,b) = releasing3((a-1)*s+b,6);
    end
end

c = floor(length(releasing4)/s);
r4_1 = zeros(c,s);
r4_2 = zeros(c,s);
r4_3 = zeros(c,s);
r4_4 = zeros(c,s);
r4_5 = zeros(c,s);
r4_6 = zeros(c,s);
for a = 1:c:
    for b = 1:s:
        r4_1(a,b) = releasing4((a-1)*s+b,1);
        r4_2(a,b) = releasing4((a-1)*s+b,2);
        r4_3(a,b) = releasing4((a-1)*s+b,3);
        r4_4(a,b) = releasing4((a-1)*s+b,4);
        r4_5(a,b) = releasing4((a-1)*s+b,5);
        r4_6(a,b) = releasing4((a-1)*s+b,6);
    end
end

c = floor(length(releasing5)/s);
r5_1 = zeros(c,s);
r5_2 = zeros(c,s);
r5_3 = zeros(c,s);
r5_4 = zeros(c,s);
r5_5 = zeros(c,s);
r5_6 = zeros(c,s);
for a = 1:c:
    for b = 1:s:
        r5_1(a,b) = releasing5((a-1)*s+b,1);
        r5_2(a,b) = releasing5((a-1)*s+b,2);
        r5_3(a,b) = releasing5((a-1)*s+b,3);
        r5_4(a,b) = releasing5((a-1)*s+b,4);
        r5_5(a,b) = releasing5((a-1)*s+b,5);
        r5_6(a,b) = releasing5((a-1)*s+b,6);
    end
end

c = floor(length(releasing6)/s);
r6_1 = zeros(c,s);
r6_2 = zeros(c,s);
r6_3 = zeros(c,s);
r6_4 = zeros(c,s);
r6_5 = zeros(c,s);
r6_6 = zeros(c,s);
for a = 1:c:
    for b = 1:s:
        r6_1(a,b) = releasing6((a-1)*s+b,1);
        r6_2(a,b) = releasing6((a-1)*s+b,2);
        r6_3(a,b) = releasing6((a-1)*s+b,3);
\begin{verbatim}
 r6_4(a,b) = releasing6((a-1)*s+b,4);
 r6_5(a,b) = releasing6((a-1)*s+b,5);
 r6_6(a,b) = releasing6((a-1)*s+b,6);
 end
 end

 c = floor(length(releasing7)/s);
 r7_1 = zeros(c,s);
 r7_2 = zeros(c,s);
 r7_3 = zeros(c,s);
 r7_4 = zeros(c,s);
 r7_5 = zeros(c,s);
 r7_6 = zeros(c,s);
 for a = 1:c
   for b = 1:s
     r7_1(a,b) = releasing7((a-1)*s+b,1);
     r7_2(a,b) = releasing7((a-1)*s+b,2);
     r7_3(a,b) = releasing7((a-1)*s+b,3);
     r7_4(a,b) = releasing7((a-1)*s+b,4);
     r7_5(a,b) = releasing7((a-1)*s+b,5);
     r7_6(a,b) = releasing7((a-1)*s+b,6);
   end
 end

 c = floor(length(releasing8)/s);
 r8_1 = zeros(c,s);
 r8_2 = zeros(c,s);
 r8_3 = zeros(c,s);
 r8_4 = zeros(c,s);
 r8_5 = zeros(c,s);
 r8_6 = zeros(c,s);
 for a = 1:c
   for b = 1:s
     r8_1(a,b) = releasing8((a-1)*s+b,1);
     r8_2(a,b) = releasing8((a-1)*s+b,2);
     r8_3(a,b) = releasing8((a-1)*s+b,3);
     r8_4(a,b) = releasing8((a-1)*s+b,4);
     r8_5(a,b) = releasing8((a-1)*s+b,5);
     r8_6(a,b) = releasing8((a-1)*s+b,6);
   end
 end

 c = floor(length(releasing9)/s);
 r9_1 = zeros(c,s);
 r9_2 = zeros(c,s);
 r9_3 = zeros(c,s);
 r9_4 = zeros(c,s);
 r9_5 = zeros(c,s);
 r9_6 = zeros(c,s);
 for a = 1:c
   for b = 1:s
     r9_1(a,b) = releasing9((a-1)*s+b,1);
     r9_2(a,b) = releasing9((a-1)*s+b,2);
     r9_3(a,b) = releasing9((a-1)*s+b,3);
     r9_4(a,b) = releasing9((a-1)*s+b,4);
     r9_5(a,b) = releasing9((a-1)*s+b,5);
     r9_6(a,b) = releasing9((a-1)*s+b,6);
   end
 end

 c = floor(length(releasing10)/s);
 r10_1 = zeros(c,s);
 r10_2 = zeros(c,s);
 r10_3 = zeros(c,s);
 r10_4 = zeros(c,s);
 r10_5 = zeros(c,s);
\end{verbatim}
r10_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        r10_1(a,b) = releasing10((a-1)*s+b,1);
        r10_2(a,b) = releasing10((a-1)*s+b,2);
        r10_3(a,b) = releasing10((a-1)*s+b,3);
        r10_4(a,b) = releasing10((a-1)*s+b,4);
        r10_5(a,b) = releasing10((a-1)*s+b,5);
        r10_6(a,b) = releasing10((a-1)*s+b,6);
    end
end
r_train_1 = [r1_1;r2_1;r3_1;r4_1;r5_1];
r_test_1 = [r6_1;r7_1;r8_1;r9_1;r10_1];
r_train_2 = [r1_2;r2_2;r3_2;r4_2;r5_2];
r_test_2 = [r6_2;r7_2;r8_2;r9_2;r10_2];
r_train_3 = [r1_3;r2_3;r3_3;r4_3;r5_3];
r_test_3 = [r6_3;r7_3;r8_3;r9_3;r10_3];
r_train_4 = [r1_4;r2_4;r3_4;r4_4;r5_4];
r_test_4 = [r6_4;r7_4;r8_4;r9_4;r10_4];
r_train_5 = [r1_5;r2_5;r3_5;r4_5;r5_5];
r_test_5 = [r6_5;r7_5;r8_5;r9_5;r10_5];
r_train_6 = [r1_6;r2_6;r3_6;r4_6;r5_6];
r_test_6 = [r6_6;r7_6;r8_6;r9_6;r10_6];

% The code below is the same code used above except for the pronation class

for a = 1:c
    for b = 1:s
        p1_1(a,b) = pronation1((a-1)*s+b,1);
        p1_2(a,b) = pronation1((a-1)*s+b,2);
        p1_3(a,b) = pronation1((a-1)*s+b,3);
        p1_4(a,b) = pronation1((a-1)*s+b,4);
        p1_5(a,b) = pronation1((a-1)*s+b,5);
        p1_6(a,b) = pronation1((a-1)*s+b,6);
    end
end

for a = 1:c
    for b = 1:s
        p2_1(a,b) = pronation2((a-1)*s+b,1);
        p2_2(a,b) = pronation2((a-1)*s+b,2);
        p2_3(a,b) = pronation2((a-1)*s+b,3);
        p2_4(a,b) = pronation2((a-1)*s+b,4);
        p2_5(a,b) = pronation2((a-1)*s+b,5);
        p2_6(a,b) = pronation2((a-1)*s+b,6);
    end
end

for a = 1:c
    for b = 1:s
        p3_1(a,b) = pronation3((a-1)*s+b,1);
        p3_2(a,b) = pronation3((a-1)*s+b,2);
        p3_3(a,b) = pronation3((a-1)*s+b,3);
        p3_4(a,b) = pronation3((a-1)*s+b,4);
        p3_5(a,b) = pronation3((a-1)*s+b,5);
        p3_6(a,b) = pronation3((a-1)*s+b,6);
    end
end
p3_3 = zeros(c,s);
p3_4 = zeros(c,s);
p3_5 = zeros(c,s);
p3_6 = zeros(c,s);
for a = 1:c:
  for b = 1:s:
    p3_1(a,b) = pronation3((a-1)*s+b,1);
    p3_2(a,b) = pronation3((a-1)*s+b,2);
    p3_3(a,b) = pronation3((a-1)*s+b,3);
    p3_4(a,b) = pronation3((a-1)*s+b,4);
    p3_5(a,b) = pronation3((a-1)*s+b,5);
    p3_6(a,b) = pronation3((a-1)*s+b,6);
  end
end

c = floor(length(pronation4)/s);
p4_1 = zeros(c,s);
p4_2 = zeros(c,s);
p4_3 = zeros(c,s);
p4_4 = zeros(c,s);
p4_5 = zeros(c,s);
p4_6 = zeros(c,s);
for a = 1:c:
  for b = 1:s:
    p4_1(a,b) = pronation4((a-1)*s+b,1);
    p4_2(a,b) = pronation4((a-1)*s+b,2);
    p4_3(a,b) = pronation4((a-1)*s+b,3);
    p4_4(a,b) = pronation4((a-1)*s+b,4);
    p4_5(a,b) = pronation4((a-1)*s+b,5);
    p4_6(a,b) = pronation4((a-1)*s+b,6);
  end
end

c = floor(length(pronation5)/s);
p5_1 = zeros(c,s);
p5_2 = zeros(c,s);
p5_3 = zeros(c,s);
p5_4 = zeros(c,s);
p5_5 = zeros(c,s);
p5_6 = zeros(c,s);
for a = 1:c:
  for b = 1:s:
    p5_1(a,b) = pronation5((a-1)*s+b,1);
    p5_2(a,b) = pronation5((a-1)*s+b,2);
    p5_3(a,b) = pronation5((a-1)*s+b,3);
    p5_4(a,b) = pronation5((a-1)*s+b,4);
    p5_5(a,b) = pronation5((a-1)*s+b,5);
    p5_6(a,b) = pronation5((a-1)*s+b,6);
  end
end

c = floor(length(pronation6)/s);
p6_1 = zeros(c,s);
p6_2 = zeros(c,s);
p6_3 = zeros(c,s);
p6_4 = zeros(c,s);
p6_5 = zeros(c,s);
p6_6 = zeros(c,s);
for a = 1:c:
  for b = 1:s:
    p6_1(a,b) = pronation6((a-1)*s+b,1);
    p6_2(a,b) = pronation6((a-1)*s+b,2);
    p6_3(a,b) = pronation6((a-1)*s+b,3);
    p6_4(a,b) = pronation6((a-1)*s+b,4);
    p6_5(a,b) = pronation6((a-1)*s+b,5);

p6_6(a,b) = pronation6((a-1)*s+b,6);
end

end
c = floor(length(pronation7)/s);
p7_1 = zeros(c,s);
p7_2 = zeros(c,s);
p7_3 = zeros(c,s);
p7_4 = zeros(c,s);
p7_5 = zeros(c,s);
p7_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        p7_1(a,b) = pronation7((a-1)*s+b,1);
p7_2(a,b) = pronation7((a-1)*s+b,2);
p7_3(a,b) = pronation7((a-1)*s+b,3);
p7_4(a,b) = pronation7((a-1)*s+b,4);
p7_5(a,b) = pronation7((a-1)*s+b,5);
p7_6(a,b) = pronation7((a-1)*s+b,6);
    end
end
c = floor(length(pronation8)/s);
p8_1 = zeros(c,s);
p8_2 = zeros(c,s);
p8_3 = zeros(c,s);
p8_4 = zeros(c,s);
p8_5 = zeros(c,s);
p8_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        p8_1(a,b) = pronation8((a-1)*s+b,1);
p8_2(a,b) = pronation8((a-1)*s+b,2);
p8_3(a,b) = pronation8((a-1)*s+b,3);
p8_4(a,b) = pronation8((a-1)*s+b,4);
p8_5(a,b) = pronation8((a-1)*s+b,5);
p8_6(a,b) = pronation8((a-1)*s+b,6);
    end
end
c = floor(length(pronation9)/s);
p9_1 = zeros(c,s);
p9_2 = zeros(c,s);
p9_3 = zeros(c,s);
p9_4 = zeros(c,s);
p9_5 = zeros(c,s);
p9_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        p9_1(a,b) = pronation9((a-1)*s+b,1);
p9_2(a,b) = pronation9((a-1)*s+b,2);
p9_3(a,b) = pronation9((a-1)*s+b,3);
p9_4(a,b) = pronation9((a-1)*s+b,4);
p9_5(a,b) = pronation9((a-1)*s+b,5);
p9_6(a,b) = pronation9((a-1)*s+b,6);
    end
end
c = floor(length(pronation10)/s);
p10_1 = zeros(c,s);
p10_2 = zeros(c,s);
p10_3 = zeros(c,s);
p10_4 = zeros(c,s);
p10_5 = zeros(c,s);
p10_6 = zeros(c,s);
for a = 1:c
for b = 1:s
    p10_1(a,b) = pronation10((a-1)*s+b,1);
    p10_2(a,b) = pronation10((a-1)*s+b,2);
    p10_3(a,b) = pronation10((a-1)*s+b,3);
    p10_4(a,b) = pronation10((a-1)*s+b,4);
    p10_5(a,b) = pronation10((a-1)*s+b,5);
    p10_6(a,b) = pronation10((a-1)*s+b,6);
end
end
p_train_1 = [p1_1;p2_1;p3_1;p4_1;p5_1];
p_test_1 = [p6_1;p7_1;p8_1;p9_1;p10_1];
p_train_2 = [p1_2;p2_2;p3_2;p4_2;p5_2];
p_test_2 = [p6_2;p7_2;p8_2;p9_2;p10_2];
p_train_3 = [p1_3;p2_3;p3_3;p4_3;p5_3];
p_test_3 = [p6_3;p7_3;p8_3;p9_3;p10_3];
p_train_4 = [p1_4;p2_4;p3_4;p4_4;p5_4];
p_test_4 = [p6_4;p7_4;p8_4;p9_4;p10_4];
p_train_5 = [p1_5;p2_5;p3_5;p4_5;p5_5];
p_test_5 = [p6_5;p7_5;p8_5;p9_5;p10_5];
p_train_6 = [p1_6;p2_6;p3_6;p4_6;p5_6];
p_test_6 = [p6_6;p7_6;p8_6;p9_6;p10_6];

% The code below is the same code used above except for the supination % class
c = floor(length(supination1)/s);
s1_1 = zeros(c,s);
s1_2 = zeros(c,s);
s1_3 = zeros(c,s);
s1_4 = zeros(c,s);
s1_5 = zeros(c,s);
s1_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        s1_1(a,b) = supination1((a-1)*s+b,1);
        s1_2(a,b) = supination1((a-1)*s+b,2);
        s1_3(a,b) = supination1((a-1)*s+b,3);
        s1_4(a,b) = supination1((a-1)*s+b,4);
        s1_5(a,b) = supination1((a-1)*s+b,5);
        s1_6(a,b) = supination1((a-1)*s+b,6);
    end
end
c = floor(length(supination2)/s);
s2_1 = zeros(c,s);
s2_2 = zeros(c,s);
s2_3 = zeros(c,s);
s2_4 = zeros(c,s);
s2_5 = zeros(c,s);
s2_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        s2_1(a,b) = supination2((a-1)*s+b,1);
        s2_2(a,b) = supination2((a-1)*s+b,2);
        s2_3(a,b) = supination2((a-1)*s+b,3);
        s2_4(a,b) = supination2((a-1)*s+b,4);
        s2_5(a,b) = supination2((a-1)*s+b,5);
        s2_6(a,b) = supination2((a-1)*s+b,6);
    end
end
c = floor(length(supination3)/s);
s3_1 = zeros(c,s);
s3_2 = zeros(c,s);
s3_3 = zeros(c,s);
s3_4 = zeros(c,s);
s3_5 = zeros(c,s);
s3_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        s3_1(a,b) = supination3((a-1)*s+b,1);
s3_2(a,b) = supination3((a-1)*s+b,2);
s3_3(a,b) = supination3((a-1)*s+b,3);
s3_4(a,b) = supination3((a-1)*s+b,4);
s3_5(a,b) = supination3((a-1)*s+b,5);
s3_6(a,b) = supination3((a-1)*s+b,6);
    end
end
c = floor(length(supination4)/s);
s4_1 = zeros(c,s);
s4_2 = zeros(c,s);
s4_3 = zeros(c,s);
s4_4 = zeros(c,s);
s4_5 = zeros(c,s);
s4_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        s4_1(a,b) = supination4((a-1)*s+b,1);
s4_2(a,b) = supination4((a-1)*s+b,2);
s4_3(a,b) = supination4((a-1)*s+b,3);
s4_4(a,b) = supination4((a-1)*s+b,4);
s4_5(a,b) = supination4((a-1)*s+b,5);
s4_6(a,b) = supination4((a-1)*s+b,6);
    end
end
c = floor(length(supination5)/s);
s5_1 = zeros(c,s);
s5_2 = zeros(c,s);
s5_3 = zeros(c,s);
s5_4 = zeros(c,s);
s5_5 = zeros(c,s);
s5_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        s5_1(a,b) = supination5((a-1)*s+b,1);
s5_2(a,b) = supination5((a-1)*s+b,2);
s5_3(a,b) = supination5((a-1)*s+b,3);
s5_4(a,b) = supination5((a-1)*s+b,4);
s5_5(a,b) = supination5((a-1)*s+b,5);
s5_6(a,b) = supination5((a-1)*s+b,6);
    end
end
c = floor(length(supination6)/s);
s6_1 = zeros(c,s);
s6_2 = zeros(c,s);
s6_3 = zeros(c,s);
s6_4 = zeros(c,s);
s6_5 = zeros(c,s);
s6_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        s6_1(a,b) = supination6((a-1)*s+b,1);
s6_2(a,b) = supination6((a-1)*s+b,2);
s6_3(a,b) = supination6((a-1)*s+b,3);
s6_4(a,b) = supination6((a-1)*s+b,4);
s6_5(a,b) = supination6((a-1)*s+b,5);
s6_6(a,b) = supination6((a-1)*s+b,6);
\[ c = \text{floor}(\text{length}(\text{supination7})/s); \]
\[ s7_1 = \text{zeros}(c,s); \]
\[ s7_2 = \text{zeros}(c,s); \]
\[ s7_3 = \text{zeros}(c,s); \]
\[ s7_4 = \text{zeros}(c,s); \]
\[ s7_5 = \text{zeros}(c,s); \]
\[ s7_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \text{for } b = 1:s \]
\[ s7_1(a,b) = \text{supination7}((a-1)*s+b,1); \]
\[ s7_2(a,b) = \text{supination7}((a-1)*s+b,2); \]
\[ s7_3(a,b) = \text{supination7}((a-1)*s+b,3); \]
\[ s7_4(a,b) = \text{supination7}((a-1)*s+b,4); \]
\[ s7_5(a,b) = \text{supination7}((a-1)*s+b,5); \]
\[ s7_6(a,b) = \text{supination7}((a-1)*s+b,6); \]
\[ \text{end} \]
\[ \text{end} \]
\[ c = \text{floor}(\text{length}(\text{supination8})/s); \]
\[ s8_1 = \text{zeros}(c,s); \]
\[ s8_2 = \text{zeros}(c,s); \]
\[ s8_3 = \text{zeros}(c,s); \]
\[ s8_4 = \text{zeros}(c,s); \]
\[ s8_5 = \text{zeros}(c,s); \]
\[ s8_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \text{for } b = 1:s \]
\[ s8_1(a,b) = \text{supination8}((a-1)*s+b,1); \]
\[ s8_2(a,b) = \text{supination8}((a-1)*s+b,2); \]
\[ s8_3(a,b) = \text{supination8}((a-1)*s+b,3); \]
\[ s8_4(a,b) = \text{supination8}((a-1)*s+b,4); \]
\[ s8_5(a,b) = \text{supination8}((a-1)*s+b,5); \]
\[ s8_6(a,b) = \text{supination8}((a-1)*s+b,6); \]
\[ \text{end} \]
\[ \text{end} \]
\[ c = \text{floor}(\text{length}(\text{supination9})/s); \]
\[ s9_1 = \text{zeros}(c,s); \]
\[ s9_2 = \text{zeros}(c,s); \]
\[ s9_3 = \text{zeros}(c,s); \]
\[ s9_4 = \text{zeros}(c,s); \]
\[ s9_5 = \text{zeros}(c,s); \]
\[ s9_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \text{for } b = 1:s \]
\[ s9_1(a,b) = \text{supination9}((a-1)*s+b,1); \]
\[ s9_2(a,b) = \text{supination9}((a-1)*s+b,2); \]
\[ s9_3(a,b) = \text{supination9}((a-1)*s+b,3); \]
\[ s9_4(a,b) = \text{supination9}((a-1)*s+b,4); \]
\[ s9_5(a,b) = \text{supination9}((a-1)*s+b,5); \]
\[ s9_6(a,b) = \text{supination9}((a-1)*s+b,6); \]
\[ \text{end} \]
\[ \text{end} \]
\[ c = \text{floor}(\text{length}(\text{supination10})/s); \]
\[ s10_1 = \text{zeros}(c,s); \]
\[ s10_2 = \text{zeros}(c,s); \]
\[ s10_3 = \text{zeros}(c,s); \]
\[ s10_4 = \text{zeros}(c,s); \]
\[ s10_5 = \text{zeros}(c,s); \]
\[ s10_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \text{for } b = 1:s \]
s10_1(a,b) = supination10((a-1)*s+b,1);
s10_2(a,b) = supination10((a-1)*s+b,2);
s10_3(a,b) = supination10((a-1)*s+b,3);
s10_4(a,b) = supination10((a-1)*s+b,4);
s10_5(a,b) = supination10((a-1)*s+b,5);
s10_6(a,b) = supination10((a-1)*s+b,6);
end
end
s_train_1 = [s1_1;s2_1;s3_1;s4_1;s5_1];
s_test_1 = [s6_1;s7_1;s8_1;s9_1;s10_1];
s_train_2 = [s1_2;s2_2;s3_2;s4_2;s5_2];
s_test_2 = [s6_2;s7_2;s8_2;s9_2;s10_2];
s_train_3 = [s1_3;s2_3;s3_3;s4_3;s5_3];
s_test_3 = [s6_3;s7_3;s8_3;s9_3;s10_3];
s_train_4 = [s1_4;s2_4;s3_4;s4_4;s5_4];
s_test_4 = [s6_4;s7_4;s8_4;s9_4;s10_4];
s_train_5 = [s1_5;s2_5;s3_5;s4_5;s5_5];
s_test_5 = [s6_5;s7_5;s8_5;s9_5;s10_5];
s_train_6 = [s1_6;s2_6;s3_6;s4_6;s5_6];
s_test_6 = [s6_6;s7_6;s8_6;s9_6;s10_6];

% The code below is the same code used above except for the extension class
c = floor(length(extension1)/s);
e1_1 = zeros(c,s);
e1_2 = zeros(c,s);
e1_3 = zeros(c,s);
e1_4 = zeros(c,s);
e1_5 = zeros(c,s);
e1_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e1_1(a,b) = extension1((a-1)*s+b,1);
e1_2(a,b) = extension1((a-1)*s+b,2);
e1_3(a,b) = extension1((a-1)*s+b,3);
e1_4(a,b) = extension1((a-1)*s+b,4);
e1_5(a,b) = extension1((a-1)*s+b,5);
e1_6(a,b) = extension1((a-1)*s+b,6);
    end
end
c = floor(length(extension2)/s);
e2_1 = zeros(c,s);
e2_2 = zeros(c,s);
e2_3 = zeros(c,s);
e2_4 = zeros(c,s);
e2_5 = zeros(c,s);
e2_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e2_1(a,b) = extension2((a-1)*s+b,1);
e2_2(a,b) = extension2((a-1)*s+b,2);
e2_3(a,b) = extension2((a-1)*s+b,3);
e2_4(a,b) = extension2((a-1)*s+b,4);
e2_5(a,b) = extension2((a-1)*s+b,5);
e2_6(a,b) = extension2((a-1)*s+b,6);
    end
end
c = floor(length(extension3)/s);
e3_1 = zeros(c,s);
e3_2 = zeros(c,s);
e3_3 = zeros(c,s);
e3_4 = zeros(c,s);
e3_5 = zeros(c,s);
e3_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e3_1(a,b) = extension3((a-1)*s+b,1);
        e3_2(a,b) = extension3((a-1)*s+b,2);
        e3_3(a,b) = extension3((a-1)*s+b,3);
        e3_4(a,b) = extension3((a-1)*s+b,4);
        e3_5(a,b) = extension3((a-1)*s+b,5);
        e3_6(a,b) = extension3((a-1)*s+b,6);
    end
end
c = floor(length(extension4)/s);
e4_1 = zeros(c,s);
e4_2 = zeros(c,s);
e4_3 = zeros(c,s);
e4_4 = zeros(c,s);
e4_5 = zeros(c,s);
e4_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e4_1(a,b) = extension4((a-1)*s+b,1);
        e4_2(a,b) = extension4((a-1)*s+b,2);
        e4_3(a,b) = extension4((a-1)*s+b,3);
        e4_4(a,b) = extension4((a-1)*s+b,4);
        e4_5(a,b) = extension4((a-1)*s+b,5);
        e4_6(a,b) = extension4((a-1)*s+b,6);
    end
end
c = floor(length(extension5)/s);
e5_1 = zeros(c,s);
e5_2 = zeros(c,s);
e5_3 = zeros(c,s);
e5_4 = zeros(c,s);
e5_5 = zeros(c,s);
e5_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e5_1(a,b) = extension5((a-1)*s+b,1);
        e5_2(a,b) = extension5((a-1)*s+b,2);
        e5_3(a,b) = extension5((a-1)*s+b,3);
        e5_4(a,b) = extension5((a-1)*s+b,4);
        e5_5(a,b) = extension5((a-1)*s+b,5);
        e5_6(a,b) = extension5((a-1)*s+b,6);
    end
end
c = floor(length(extension6)/s);
e6_1 = zeros(c,s);
e6_2 = zeros(c,s);
e6_3 = zeros(c,s);
e6_4 = zeros(c,s);
e6_5 = zeros(c,s);
e6_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e6_1(a,b) = extension6((a-1)*s+b,1);
        e6_2(a,b) = extension6((a-1)*s+b,2);
        e6_3(a,b) = extension6((a-1)*s+b,3);
        e6_4(a,b) = extension6((a-1)*s+b,4);
        e6_5(a,b) = extension6((a-1)*s+b,5);
        e6_6(a,b) = extension6((a-1)*s+b,6);
    end
end
c = floor(length(extension7)/s);
e7_1 = zeros(c,s);
e7_2 = zeros(c,s);
e7_3 = zeros(c,s);
e7_4 = zeros(c,s);
e7_5 = zeros(c,s);
e7_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e7_1(a,b) = extension7((a-1)*s+b,1);
e7_2(a,b) = extension7((a-1)*s+b,2);
e7_3(a,b) = extension7((a-1)*s+b,3);
e7_4(a,b) = extension7((a-1)*s+b,4);
e7_5(a,b) = extension7((a-1)*s+b,5);
e7_6(a,b) = extension7((a-1)*s+b,6);
    end
end
c = floor(length(extension8)/s);
e8_1 = zeros(c,s);
e8_2 = zeros(c,s);
e8_3 = zeros(c,s);
e8_4 = zeros(c,s);
e8_5 = zeros(c,s);
e8_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e8_1(a,b) = extension8((a-1)*s+b,1);
e8_2(a,b) = extension8((a-1)*s+b,2);
e8_3(a,b) = extension8((a-1)*s+b,3);
e8_4(a,b) = extension8((a-1)*s+b,4);
e8_5(a,b) = extension8((a-1)*s+b,5);
e8_6(a,b) = extension8((a-1)*s+b,6);
    end
end
c = floor(length(extension9)/s);
e9_1 = zeros(c,s);
e9_2 = zeros(c,s);
e9_3 = zeros(c,s);
e9_4 = zeros(c,s);
e9_5 = zeros(c,s);
e9_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e9_1(a,b) = extension9((a-1)*s+b,1);
e9_2(a,b) = extension9((a-1)*s+b,2);
e9_3(a,b) = extension9((a-1)*s+b,3);
e9_4(a,b) = extension9((a-1)*s+b,4);
e9_5(a,b) = extension9((a-1)*s+b,5);
e9_6(a,b) = extension9((a-1)*s+b,6);
    end
end
c = floor(length(extension10)/s);
e10_1 = zeros(c,s);
e10_2 = zeros(c,s);
e10_3 = zeros(c,s);
e10_4 = zeros(c,s);
e10_5 = zeros(c,s);
e10_6 = zeros(c,s);
for a = 1:c
    for b = 1:s
        e10_1(a,b) = extension10((a-1)*s+b,1);
e10_2(a,b) = extension10((a-1)*s+b,2);
\[
e_{10,3}(a,b) = \text{extension10}((a-1)*s+b,3);
\]
\[
e_{10,4}(a,b) = \text{extension10}((a-1)*s+b,4);
\]
\[
e_{10,5}(a,b) = \text{extension10}((a-1)*s+b,5);
\]
\[
e_{10,6}(a,b) = \text{extension10}((a-1)*s+b,6);
\]
\[
\text{end}
\]
\[
\text{end}
\]
\[
e_{\text{train},1} = [e_{1,1};e_{2,1};e_{3,1};e_{4,1};e_{5,1}];
\]
\[
e_{\text{test},1} = [e_{6,1};e_{7,1};e_{8,1};e_{9,1};e_{10,1}];
\]
\[
e_{\text{train},2} = [e_{1,2};e_{2,2};e_{3,2};e_{4,2};e_{5,2}];
\]
\[
e_{\text{test},2} = [e_{6,2};e_{7,2};e_{8,2};e_{9,2};e_{10,2}];
\]
\[
e_{\text{train},3} = [e_{1,3};e_{2,3};e_{3,3};e_{4,3};e_{5,3}];
\]
\[
e_{\text{test},3} = [e_{6,3};e_{7,3};e_{8,3};e_{9,3};e_{10,3}];
\]
\[
e_{\text{train},4} = [e_{1,4};e_{2,4};e_{3,4};e_{4,4};e_{5,4}];
\]
\[
e_{\text{test},4} = [e_{6,4};e_{7,4};e_{8,4};e_{9,4};e_{10,4}];
\]
\[
e_{\text{train},5} = [e_{1,5};e_{2,5};e_{3,5};e_{4,5};e_{5,5}];
\]
\[
e_{\text{test},5} = [e_{6,5};e_{7,5};e_{8,5};e_{9,5};e_{10,5}];
\]
\[
e_{\text{train},6} = [e_{1,6};e_{2,6};e_{3,6};e_{4,6};e_{5,6}];
\]
\[
e_{\text{test},6} = [e_{6,6};e_{7,6};e_{8,6};e_{9,6};e_{10,6}];
\]

\% The code below is the same code used above except for the flexion class
\]
\[
c = \text{floor}(\text{length}(\text{flexion1}))/s);
\]
\[
f_{1,1} = \text{zeros}(c,s);
\]
\[
f_{1,2} = \text{zeros}(c,s);
\]
\[
f_{1,3} = \text{zeros}(c,s);
\]
\[
f_{1,4} = \text{zeros}(c,s);
\]
\[
f_{1,5} = \text{zeros}(c,s);
\]
\[
f_{1,6} = \text{zeros}(c,s);
\]
\[
\text{for } a = 1:c
\]
\[
\quad \text{for } b = 1:s
\]
\[
\quad \quad f_{1,1}(a,b) = \text{flexion1}((a-1)*s+b,1);
\]
\[
\quad \quad f_{1,2}(a,b) = \text{flexion1}((a-1)*s+b,2);
\]
\[
\quad \quad f_{1,3}(a,b) = \text{flexion1}((a-1)*s+b,3);
\]
\[
\quad \quad f_{1,4}(a,b) = \text{flexion1}((a-1)*s+b,4);
\]
\[
\quad \quad f_{1,5}(a,b) = \text{flexion1}((a-1)*s+b,5);
\]
\[
\quad \quad f_{1,6}(a,b) = \text{flexion1}((a-1)*s+b,6);
\]
\[
\quad \text{end}
\]
\[
\text{end}
\]
\[
c = \text{floor}(\text{length}(\text{flexion2}))/s);
\]
\[
f_{2,1} = \text{zeros}(c,s);
\]
\[
f_{2,2} = \text{zeros}(c,s);
\]
\[
f_{2,3} = \text{zeros}(c,s);
\]
\[
f_{2,4} = \text{zeros}(c,s);
\]
\[
f_{2,5} = \text{zeros}(c,s);
\]
\[
f_{2,6} = \text{zeros}(c,s);
\]
\[
\text{for } a = 1:c
\]
\[
\quad \text{for } b = 1:s
\]
\[
\quad \quad f_{2,1}(a,b) = \text{flexion2}((a-1)*s+b,1);
\]
\[
\quad \quad f_{2,2}(a,b) = \text{flexion2}((a-1)*s+b,2);
\]
\[
\quad \quad f_{2,3}(a,b) = \text{flexion2}((a-1)*s+b,3);
\]
\[
\quad \quad f_{2,4}(a,b) = \text{flexion2}((a-1)*s+b,4);
\]
\[
\quad \quad f_{2,5}(a,b) = \text{flexion2}((a-1)*s+b,5);
\]
\[
\quad \quad f_{2,6}(a,b) = \text{flexion2}((a-1)*s+b,6);
\]
\[
\quad \text{end}
\]
\[
\text{end}
\]
\[
c = \text{floor}(\text{length}(\text{flexion3}))/s);
\]
\[
f_{3,1} = \text{zeros}(c,s);
\]
\[
f_{3,2} = \text{zeros}(c,s);
\]
\[
f_{3,3} = \text{zeros}(c,s);
\]
\[
f_{3,4} = \text{zeros}(c,s);
\]
\[
f_{3,5} = \text{zeros}(c,s);
\]
\[
f_{3,6} = \text{zeros}(c,s);
\]
\[
\text{for } a = 1:c
for b = 1:s
    f3_1(a,b) = flexion3((a-1)*s+b,1);
    f3_2(a,b) = flexion3((a-1)*s+b,2);
    f3_3(a,b) = flexion3((a-1)*s+b,3);
    f3_4(a,b) = flexion3((a-1)*s+b,4);
    f3_5(a,b) = flexion3((a-1)*s+b,5);
    f3_6(a,b) = flexion3((a-1)*s+b,6);
end

c = floor(length(flexion4)/s);
for a = 1:c
    for b = 1:s
        f4_1(a,b) = flexion4((a-1)*s+b,1);
        f4_2(a,b) = flexion4((a-1)*s+b,2);
        f4_3(a,b) = flexion4((a-1)*s+b,3);
        f4_4(a,b) = flexion4((a-1)*s+b,4);
        f4_5(a,b) = flexion4((a-1)*s+b,5);
        f4_6(a,b) = flexion4((a-1)*s+b,6);
    end
end

c = floor(length(flexion5)/s);
for a = 1:c
    for b = 1:s
        f5_1(a,b) = flexion5((a-1)*s+b,1);
        f5_2(a,b) = flexion5((a-1)*s+b,2);
        f5_3(a,b) = flexion5((a-1)*s+b,3);
        f5_4(a,b) = flexion5((a-1)*s+b,4);
        f5_5(a,b) = flexion5((a-1)*s+b,5);
        f5_6(a,b) = flexion5((a-1)*s+b,6);
    end
end

c = floor(length(flexion6)/s);
for a = 1:c
    for b = 1:s
        f6_1(a,b) = flexion6((a-1)*s+b,1);
        f6_2(a,b) = flexion6((a-1)*s+b,2);
        f6_3(a,b) = flexion6((a-1)*s+b,3);
        f6_4(a,b) = flexion6((a-1)*s+b,4);
        f6_5(a,b) = flexion6((a-1)*s+b,5);
        f6_6(a,b) = flexion6((a-1)*s+b,6);
    end
end

c = floor(length(flexion7)/s);
for a = 1:c
    for b = 1:s
        f7_1(a,b) = flexion7((a-1)*s+b,1);
        f7_2(a,b) = flexion7((a-1)*s+b,2);
        f7_3(a,b) = flexion7((a-1)*s+b,3);
        f7_4(a,b) = flexion7((a-1)*s+b,4);
        f7_5(a,b) = flexion7((a-1)*s+b,5);
        f7_6(a,b) = flexion7((a-1)*s+b,6);
    end
end

\[ f_7_2 = \text{zeros}(c,s); \]
\[ f_7_3 = \text{zeros}(c,s); \]
\[ f_7_4 = \text{zeros}(c,s); \]
\[ f_7_5 = \text{zeros}(c,s); \]
\[ f_7_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \quad \text{for } b = 1:s \]
\[ \quad \quad f_7_1(a,b) = \text{flexion7}((a-1)*s+b,1); \]
\[ \quad \quad f_7_2(a,b) = \text{flexion7}((a-1)*s+b,2); \]
\[ \quad \quad f_7_3(a,b) = \text{flexion7}((a-1)*s+b,3); \]
\[ \quad \quad f_7_4(a,b) = \text{flexion7}((a-1)*s+b,4); \]
\[ \quad \quad f_7_5(a,b) = \text{flexion7}((a-1)*s+b,5); \]
\[ \quad \quad f_7_6(a,b) = \text{flexion7}((a-1)*s+b,6); \]
\[ \quad \text{end} \]
\[ \text{end} \]
\[ c = \text{floor}(\text{length}(\text{flexion8})/s); \]
\[ f_8_1 = \text{zeros}(c,s); \]
\[ f_8_2 = \text{zeros}(c,s); \]
\[ f_8_3 = \text{zeros}(c,s); \]
\[ f_8_4 = \text{zeros}(c,s); \]
\[ f_8_5 = \text{zeros}(c,s); \]
\[ f_8_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \quad \text{for } b = 1:s \]
\[ \quad \quad f_8_1(a,b) = \text{flexion8}((a-1)*s+b,1); \]
\[ \quad \quad f_8_2(a,b) = \text{flexion8}((a-1)*s+b,2); \]
\[ \quad \quad f_8_3(a,b) = \text{flexion8}((a-1)*s+b,3); \]
\[ \quad \quad f_8_4(a,b) = \text{flexion8}((a-1)*s+b,4); \]
\[ \quad \quad f_8_5(a,b) = \text{flexion8}((a-1)*s+b,5); \]
\[ \quad \quad f_8_6(a,b) = \text{flexion8}((a-1)*s+b,6); \]
\[ \quad \text{end} \]
\[ \text{end} \]
\[ c = \text{floor}(\text{length}(\text{flexion9})/s); \]
\[ f_9_1 = \text{zeros}(c,s); \]
\[ f_9_2 = \text{zeros}(c,s); \]
\[ f_9_3 = \text{zeros}(c,s); \]
\[ f_9_4 = \text{zeros}(c,s); \]
\[ f_9_5 = \text{zeros}(c,s); \]
\[ f_9_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \quad \text{for } b = 1:s \]
\[ \quad \quad f_9_1(a,b) = \text{flexion9}((a-1)*s+b,1); \]
\[ \quad \quad f_9_2(a,b) = \text{flexion9}((a-1)*s+b,2); \]
\[ \quad \quad f_9_3(a,b) = \text{flexion9}((a-1)*s+b,3); \]
\[ \quad \quad f_9_4(a,b) = \text{flexion9}((a-1)*s+b,4); \]
\[ \quad \quad f_9_5(a,b) = \text{flexion9}((a-1)*s+b,5); \]
\[ \quad \quad f_9_6(a,b) = \text{flexion9}((a-1)*s+b,6); \]
\[ \quad \text{end} \]
\[ \text{end} \]
\[ c = \text{floor}(\text{length}(\text{flexion10})/s); \]
\[ f_{10}_1 = \text{zeros}(c,s); \]
\[ f_{10}_2 = \text{zeros}(c,s); \]
\[ f_{10}_3 = \text{zeros}(c,s); \]
\[ f_{10}_4 = \text{zeros}(c,s); \]
\[ f_{10}_5 = \text{zeros}(c,s); \]
\[ f_{10}_6 = \text{zeros}(c,s); \]
\[ \text{for } a = 1:c \]
\[ \quad \text{for } b = 1:s \]
\[ \quad \quad f_{10}_1(a,b) = \text{flexion10}((a-1)*s+b,1); \]
\[ \quad \quad f_{10}_2(a,b) = \text{flexion10}((a-1)*s+b,2); \]
\[ \quad \quad f_{10}_3(a,b) = \text{flexion10}((a-1)*s+b,3); \]
\[ \quad \quad f_{10}_4(a,b) = \text{flexion10}((a-1)*s+b,4); \]
\[
f_{10\_5}(a,b) = \text{flexion10}((a-1)\cdot s+b,5); \\
f_{10\_6}(a,b) = \text{flexion10}((a-1)\cdot s+b,6); \\
\end{array}
\]

\[
\text{c} = \text{floor}(\text{length}(\text{rest})/s); \\
\text{rest}_{-1} = \text{zeros}(<c>,s); \\
\text{rest}_2 = \text{zeros}(<c>,s); \\
\text{rest}_3 = \text{zeros}(<c>,s); \\
\text{rest}_4 = \text{zeros}(<c>,s); \\
\text{rest}_5 = \text{zeros}(<c>,s); \\
\text{rest}_6 = \text{zeros}(<c>,s); \\
\text{for} \ a = 1:<c> \\
\quad \text{for} \ b = 1:s \\
\quad\quad \text{rest}_{1}(a,b) = \text{rest}((a-1)\cdot s+b,1); \\
\quad\quad \text{rest}_{2}(a,b) = \text{rest}((a-1)\cdot s+b,2); \\
\quad\quad \text{rest}_{3}(a,b) = \text{rest}((a-1)\cdot s+b,3); \\
\quad\quad \text{rest}_{4}(a,b) = \text{rest}((a-1)\cdot s+b,4); \\
\quad\quad \text{rest}_{5}(a,b) = \text{rest}((a-1)\cdot s+b,5); \\
\quad\quad \text{rest}_{6}(a,b) = \text{rest}((a-1)\cdot s+b,6); \\
\quad \text{end} \\
\text{end} \\
\%
\text{Unlike, the 10 trials used for all of the other classes, the rest class} \\
\%
\text{had only 1 long trial of data because a subject will not fatigue while} \\
\%
\text{resting like they would while performing contractions for other classes.} \\
\%
\text{As a result, the windows for each channel were split in two groups, one} \\
\%
\text{for training and one for testing.} \\
\text{c} = \text{floor}(\text{length}(\text{rest}_{-1})/2); \\
\text{rest}_{\text{train}}_1 = [\text{rest}_{-1}(1:<c>):]; \\
\text{rest}_{\text{test}}_1 = [\text{rest}_{-1}(c+1:end):]; \\
\text{rest}_{\text{train}}_2 = [\text{rest}_{-2}(1:<c>):]; \\
\text{rest}_{\text{test}}_2 = [\text{rest}_{-2}(c+1:end):]; \\
\text{rest}_{\text{train}}_3 = [\text{rest}_{-3}(1:<c>):]; \\
\text{rest}_{\text{test}}_3 = [\text{rest}_{-3}(c+1:end):]; \\
\text{rest}_{\text{train}}_4 = [\text{rest}_{-4}(1:<c>):]; \\
\text{rest}_{\text{test}}_4 = [\text{rest}_{-4}(c+1:end):]; \\
\text{rest}_{\text{train}}_5 = [\text{rest}_{-5}(1:<c>):]; \\
\text{rest}_{\text{test}}_5 = [\text{rest}_{-5}(c+1:end):]; \\
\text{rest}_{\text{train}}_6 = [\text{rest}_{-6}(1:<c>):]; \\
\text{rest}_{\text{test}}_6 = [\text{rest}_{-6}(c+1:end):];
Appendix C: Matlab Code for Testing AR Features

% This m-file extracts the AR coefficients from the EMG data and performs
% classification using LDA.

% Prior to using this m-file, the EMG data was segmented up into windows of
% the corresponding lengths below. Depending on which window was being
% tested, one of the mat files below would be uncommented to load the EMG
% data prior to performing any signal processing.
% load raw_data_50ms.mat
% load raw_data_100ms.mat
% load raw_data_250ms.mat

% This section of code extracts all of the autoregressive features from
% each EMG channel for each class of data. A fourth order model was used.

% Memory Allocation
grasp_features_train = zeros(size(g_train_1,1),24);
grasp_features_test = zeros(size(g_test_1,1),24);
releasing_features_train = zeros(size(r_train_1,1),24);
releasing_features_test = zeros(size(r_test_1,1),24);
pronation_features_train = zeros(size(p_train_1,1),24);
pronation_features_test = zeros(size(p_test_1,1),24);
supination_features_train = zeros(size(s_train_1,1),24);
supination_features_test = zeros(size(s_test_1,1),24);
extension_features_train = zeros(size(e_train_1,1),24);
extension_features_test = zeros(size(e_test_1,1),24);
flexion_features_train = zeros(size(f_train_1,1),24);
flexion_features_test = zeros(size(f_test_1,1),24);
rest_features_train = zeros(size(rest_train_1,1),24);
rest_features_test = zeros(size(rest_test_1,1),24);

% Extracting the features for grasping
for a = 1:size(g_train_1,1)
    temp = aryule(g_train_1(a,:),4);
grasp_features_train(a,1:4) = temp(2:5);
end
for a = 1:size(g_train_2,1)
    temp = aryule(g_train_2(a,:),4);
grasp_features_train(a,5:8) = temp(2:5);
end
for a = 1:size(g_train_3,1)
    temp = aryule(g_train_3(a,:),4);
grasp_features_train(a,9:12) = temp(2:5);
end
for a = 1:size(g_train_4,1)
    temp = aryule(g_train_4(a,:),4);
grasp_features_train(a,13:16) = temp(2:5);
end
for a = 1:size(g_train_5,1)
    temp = aryule(g_train_5(a,:),4);
grasp_features_train(a,17:20) = temp(2:5);
end
for a = 1:size(g_train_6,1)
    temp = aryule(g_train_6(a,:),4);
grasp_features_train(a,21:24) = temp(2:5);
end
for a = 1:size(g_test_1,1)
    temp = aryule(g_test_1(a,:),4);
grasp_features_test(a,1:4) = temp(2:5);
for a = 1:size(g_test_2,1)
    temp = aryule(g_test_2(a,:),4);
    grasp_features_test(a,5:8) = temp(2:5);
end
for a = 1:size(g_test_3,1)
    temp = aryule(g_test_3(a,:),4);
    grasp_features_test(a,9:12) = temp(2:5);
end
for a = 1:size(g_test_4,1)
    temp = aryule(g_test_4(a,:),4);
    grasp_features_test(a,13:16) = temp(2:5);
end
for a = 1:size(g_test_5,1)
    temp = aryule(g_test_5(a,:),4);
    grasp_features_test(a,17:20) = temp(2:5);
end
for a = 1:size(g_test_6,1)
    temp = aryule(g_test_6(a,:),4);
    grasp_features_test(a,21:24) = temp(2:5);
end

% Extracting the features for releasing
for a = 1:size(r_train_1,1)
    temp = aryule(r_train_1(a,:),4);
    releasing_features_train(a,1:4) = temp(2:5);
end
for a = 1:size(r_train_2,1)
    temp = aryule(r_train_2(a,:),4);
    releasing_features_train(a,5:8) = temp(2:5);
end
for a = 1:size(r_train_3,1)
    temp = aryule(r_train_3(a,:),4);
    releasing_features_train(a,9:12) = temp(2:5);
end
for a = 1:size(r_train_4,1)
    temp = aryule(r_train_4(a,:),4);
    releasing_features_train(a,13:16) = temp(2:5);
end
for a = 1:size(r_train_5,1)
    temp = aryule(r_train_5(a,:),4);
    releasing_features_train(a,17:20) = temp(2:5);
end
for a = 1:size(r_train_6,1)
    temp = aryule(r_train_6(a,:),4);
    releasing_features_train(a,21:24) = temp(2:5);
end
for a = 1:size(r_test_1,1)
    temp = aryule(r_test_1(a,:),4);
    releasing_features_test(a,1:4) = temp(2:5);
end
for a = 1:size(r_test_2,1)
    temp = aryule(r_test_2(a,:),4);
    releasing_features_test(a,5:8) = temp(2:5);
end
for a = 1:size(r_test_3,1)
    temp = aryule(r_test_3(a,:),4);
    releasing_features_test(a,9:12) = temp(2:5);
end
for a = 1:size(r_test_4,1)
    temp = aryule(r_test_4(a,:),4);
    releasing_features_test(a,13:16) = temp(2:5);
end
for a = 1:size(r_test_5,1)
    temp = aryule(r_test_5(a,:),4);
    releasing_features_test(a,17:20) = temp(2:5);
end
for a = 1:size(r_test_6,1)
    temp = aryule(r_test_6(a,:),4);
    releasing_features_test(a,21:24) = temp(2:5);
end

% Extracting the features for pronation
for a = 1:size(p_train_1,1)
    temp = aryule(p_train_1(a,:),4);
    pronation_features_train(a,1:4) = temp(2:5);
end
for a = 1:size(p_train_2,1)
    temp = aryule(p_train_2(a,:),4);
    pronation_features_train(a,5:8) = temp(2:5);
end
for a = 1:size(p_train_3,1)
    temp = aryule(p_train_3(a,:),4);
    pronation_features_train(a,9:12) = temp(2:5);
end
for a = 1:size(p_train_4,1)
    temp = aryule(p_train_4(a,:),4);
    pronation_features_train(a,13:16) = temp(2:5);
end
for a = 1:size(p_train_5,1)
    temp = aryule(p_train_5(a,:),4);
    pronation_features_train(a,17:20) = temp(2:5);
end
for a = 1:size(p_train_6,1)
    temp = aryule(p_train_6(a,:),4);
    pronation_features_train(a,21:24) = temp(2:5);
end
for a = 1:size(p_test_1,1)
    temp = aryule(p_test_1(a,:),4);
    pronation_features_test(a,1:4) = temp(2:5);
end
for a = 1:size(p_test_2,1)
    temp = aryule(p_test_2(a,:),4);
    pronation_features_test(a,5:8) = temp(2:5);
end
for a = 1:size(p_test_3,1)
    temp = aryule(p_test_3(a,:),4);
    pronation_features_test(a,9:12) = temp(2:5);
end
for a = 1:size(p_test_4,1)
    temp = aryule(p_test_4(a,:),4);
    pronation_features_test(a,13:16) = temp(2:5);
end
for a = 1:size(p_test_5,1)
    temp = aryule(p_test_5(a,:),4);
    pronation_features_test(a,17:20) = temp(2:5);
end
for a = 1:size(p_test_6,1)
    temp = aryule(p_test_6(a,:),4);
    pronation_features_test(a,21:24) = temp(2:5);
end

% Extracting the features for supination
for a = 1:size(s_train_1,1)
temp = aryule(s_train_1(a,:),4);
supination_features_train(a,1:4) = temp(2:5);
end
for a = 1:size(s_train_2,1)
temp = aryule(s_train_2(a,:),4);
supination_features_train(a,5:8) = temp(2:5);
end
for a = 1:size(s_train_3,1)
temp = aryule(s_train_3(a,:),4);
supination_features_train(a,9:12) = temp(2:5);
end
for a = 1:size(s_train_4,1)
temp = aryule(s_train_4(a,:),4);
supination_features_train(a,13:16) = temp(2:5);
end
for a = 1:size(s_train_5,1)
temp = aryule(s_train_5(a,:),4);
supination_features_train(a,17:20) = temp(2:5);
end
for a = 1:size(s_train_6,1)
temp = aryule(s_train_6(a,:),4);
supination_features_train(a,21:24) = temp(2:5);
end
for a = 1:size(s_test_1,1)
temp = aryule(s_test_1(a,:),4);
supination_features_test(a,1:4) = temp(2:5);
end
for a = 1:size(s_test_2,1)
temp = aryule(s_test_2(a,:),4);
supination_features_test(a,5:8) = temp(2:5);
end
for a = 1:size(s_test_3,1)
temp = aryule(s_test_3(a,:),4);
supination_features_test(a,9:12) = temp(2:5);
end
for a = 1:size(s_test_4,1)
temp = aryule(s_test_4(a,:),4);
supination_features_test(a,13:16) = temp(2:5);
end
for a = 1:size(s_test_5,1)
temp = aryule(s_test_5(a,:),4);
supination_features_test(a,17:20) = temp(2:5);
end
for a = 1:size(s_test_6,1)
temp = aryule(s_test_6(a,:),4);
supination_features_test(a,21:24) = temp(2:5);
end

% Extracting the features for rest
for a = 1:size(rest_train_1,1)
temp = aryule(rest_train_1(a,:),4);
rest_features_train(a,1:4) = temp(2:5);
end
for a = 1:size(rest_train_2,1)
temp = aryule(rest_train_2(a,:),4);
rest_features_train(a,5:8) = temp(2:5);
end
for a = 1:size(rest_train_3,1)
temp = aryule(rest_train_3(a,:),4);
rest_features_train(a,9:12) = temp(2:5);
end
for a = 1:size(rest_train_4,1)

temp = aryule(rest_train_4(a,:),4);
rest_features_train(a,13:16) = temp(2:5);
end
for a = 1:size(rest_train_5,1)
  temp = aryule(rest_train_5(a,:),4);
  rest_features_train(a,17:20) = temp(2:5);
end
for a = 1:size(rest_train_6,1)
  temp = aryule(rest_train_6(a,:),4);
  rest_features_train(a,21:24) = temp(2:5);
end
for a = 1:size(rest_test_1,1)
  temp = aryule(rest_test_1(a,:),4);
  rest_features_test(a,1:4) = temp(2:5);
end
for a = 1:size(rest_test_2,1)
  temp = aryule(rest_test_2(a,:),4);
  rest_features_test(a,5:8) = temp(2:5);
end
for a = 1:size(rest_test_3,1)
  temp = aryule(rest_test_3(a,:),4);
  rest_features_test(a,9:12) = temp(2:5);
end
for a = 1:size(rest_test_4,1)
  temp = aryule(rest_test_4(a,:),4);
  rest_features_test(a,13:16) = temp(2:5);
end
for a = 1:size(rest_test_5,1)
  temp = aryule(rest_test_5(a,:),4);
  rest_features_test(a,17:20) = temp(2:5);
end
for a = 1:size(rest_test_6,1)
  temp = aryule(rest_test_6(a,:),4);
  rest_features_test(a,21:24) = temp(2:5);
end
% Extracting the features for extension
for a = 1:size(e_train_1,1)
  temp = aryule(e_train_1(a,:),4);
  extension_features_train(a,1:4) = temp(2:5);
end
for a = 1:size(e_train_2,1)
  temp = aryule(e_train_2(a,:),4);
  extension_features_train(a,5:8) = temp(2:5);
end
for a = 1:size(e_train_3,1)
  temp = aryule(e_train_3(a,:),4);
  extension_features_train(a,9:12) = temp(2:5);
end
for a = 1:size(e_train_4,1)
  temp = aryule(e_train_4(a,:),4);
  extension_features_train(a,13:16) = temp(2:5);
end
for a = 1:size(e_train_5,1)
  temp = aryule(e_train_5(a,:),4);
  extension_features_train(a,17:20) = temp(2:5);
end
for a = 1:size(e_train_6,1)
  temp = aryule(e_train_6(a,:),4);
  extension_features_train(a,21:24) = temp(2:5);
end
for a = 1:size(e_test_1,1)
temp = aryule(e_test_1(a,:),4);
extension_features_test(a,1:4) = temp(2:5);
end
for a = 1:size(e_test_2,1)
    temp = aryule(e_test_2(a,:),4);
    extension_features_test(a,5:8) = temp(2:5);
end
for a = 1:size(e_test_3,1)
    temp = aryule(e_test_3(a,:),4);
    extension_features_test(a,9:12) = temp(2:5);
end
for a = 1:size(e_test_4,1)
    temp = aryule(e_test_4(a,:),4);
    extension_features_test(a,13:16) = temp(2:5);
end
for a = 1:size(e_test_5,1)
    temp = aryule(e_test_5(a,:),4);
    extension_features_test(a,17:20) = temp(2:5);
end
for a = 1:size(e_test_6,1)
    temp = aryule(e_test_6(a,:),4);
    extension_features_test(a,21:24) = temp(2:5);
end

% Extracting the features for flexion
for a = 1:size(f_train_1,1)
    temp = aryule(f_train_1(a,:),4);
    flexion_features_train(a,1:4) = temp(2:5);
end
for a = 1:size(f_train_2,1)
    temp = aryule(f_train_2(a,:),4);
    flexion_features_train(a,5:8) = temp(2:5);
end
for a = 1:size(f_train_3,1)
    temp = aryule(f_train_3(a,:),4);
    flexion_features_train(a,9:12) = temp(2:5);
end
for a = 1:size(f_train_4,1)
    temp = aryule(f_train_4(a,:),4);
    flexion_features_train(a,13:16) = temp(2:5);
end
for a = 1:size(f_train_5,1)
    temp = aryule(f_train_5(a,:),4);
    flexion_features_train(a,17:20) = temp(2:5);
end
for a = 1:size(f_train_6,1)
    temp = aryule(f_train_6(a,:),4);
    flexion_features_train(a,21:24) = temp(2:5);
end
for a = 1:size(f_test_1,1)
    temp = aryule(f_test_1(a,:),4);
    flexion_features_test(a,1:4) = temp(2:5);
end
for a = 1:size(f_test_2,1)
    temp = aryule(f_test_2(a,:),4);
    flexion_features_test(a,5:8) = temp(2:5);
end
for a = 1:size(f_test_3,1)
    temp = aryule(f_test_3(a,:),4);
    flexion_features_test(a,9:12) = temp(2:5);
end
for a = 1:size(f_test_4,1)
temp = aryule(f_test_4(a,:),4);
flexion_features_test(a,13:16) = temp(2:5);
end
for a = 1:size(f_test_5,1)
    temp = aryule(f_test_5(a,:),4);
flexion_features_test(a,17:20) = temp(2:5);
end
for a = 1:size(f_test_6,1)
    temp = aryule(f_test_6(a,:),4);
flexion_features_test(a,21:24) = temp(2:5);
end

% This section of code uses the autoregressive features calculated above
% from training data and implements the LDA on the test data. It calculates
% the accuracy using 1,2,3, and 4 autoregressive feature(s).
% "results" holds the classification accuracies of each feature set where
% row 1 uses 1 AR coefficient, row 2 uses 2 AR, etc. The last row is the
% majority voting accuracies. Each column is the separate class accuracies
% and the last column is the total accuracy.
results = zeros(5,8);

% Calculation of the mean vectors and covariance matrices for using 1
% autoregressive coefficient as features
mean_g =
    mean([grasp_features_train(:,1),grasp_features_train(:,5),grasp_features_train(:,9),grasp_features_train(:,13),grasp_features_train(:,17),grasp_features_train(:,21)]);
mean_r =
    mean([releasing_features_train(:,1),releasing_features_train(:,5),releasing_features_train(:,9),releasing_features_train(:,13),releasing_features_train(:,17),releasing_features_train(:,21)]);
mean_p =
    mean([pronation_features_train(:,1),pronation_features_train(:,5),pronation_features_train(:,9),pronation_features_train(:,13),pronation_features_train(:,17),pronation_features_train(:,21)]);
mean_s =
    mean([supination_features_train(:,1),supination_features_train(:,5),supination_features_train(:,9),supination_features_train(:,13),supination_features_train(:,17),supination_features_train(:,21)]);
mean_e =
    mean([extension_features_train(:,1),extension_features_train(:,5),extension_features_train(:,9),extension_features_train(:,13),extension_features_train(:,17),extension_features_train(:,21)]);
mean_f =
    mean([flexion_features_train(:,1),flexion_features_train(:,5),flexion_features_train(:,9),flexion_features_train(:,13),flexion_features_train(:,17),flexion_features_train(:,21)]);
mean_rest =
    mean([rest_features_train(:,1),rest_features_train(:,5),rest_features_train(:,9),rest_features_train(:,13),rest_features_train(:,17),rest_features_train(:,21)]);

mean_g =
    mean([grasp_features_train(:,1),grasp_features_train(:,5),grasp_features_train(:,9),grasp_features_train(:,13),grasp_features_train(:,17),grasp_features_train(:,21)]);
mean_r =
    mean([releasing_features_train(:,1),releasing_features_train(:,5),releasing_features_train(:,9),releasing_features_train(:,13),releasing_features_train(:,17),releasing_features_train(:,21)]);
mean_p =
    mean([pronation_features_train(:,1),pronation_features_train(:,5),pronation_features_train(:,9),pronation_features_train(:,13),pronation_features_train(:,17),pronation_features_train(:,21)]);
mean_s =
    mean([supination_features_train(:,1),supination_features_train(:,5),supination_features_train(:,9),supination_features_train(:,13),supination_features_train(:,17),supination_features_train(:,21)]);
mean_e =
    mean([extension_features_train(:,1),extension_features_train(:,5),extension_features_train(:,9),extension_features_train(:,13),extension_features_train(:,17),extension_features_train(:,21)]);
mean_f =
    mean([flexion_features_train(:,1),flexion_features_train(:,5),flexion_features_train(:,9),flexion_features_train(:,13),flexion_features_train(:,17),flexion_features_train(:,21)]);

mean_g =
    mean([grasp_features_train(:,1),grasp_features_train(:,5),grasp_features_train(:,9),grasp_features_train(:,13),grasp_features_train(:,17),grasp_features_train(:,21)]);
mean_r =
    mean([releasing_features_train(:,1),releasing_features_train(:,5),releasing_features_train(:,9),releasing_features_train(:,13),releasing_features_train(:,17),releasing_features_train(:,21)]);
mean_p =
    mean([pronation_features_train(:,1),pronation_features_train(:,5),pronation_features_train(:,9),pronation_features_train(:,13),pronation_features_train(:,17),pronation_features_train(:,21)]);
mean_s =
    mean([supination_features_train(:,1),supination_features_train(:,5),supination_features_train(:,9),supination_features_train(:,13),supination_features_train(:,17),supination_features_train(:,21)]);
mean_e =
    mean([extension_features_train(:,1),extension_features_train(:,5),extension_features_train(:,9),extension_features_train(:,13),extension_features_train(:,17),extension_features_train(:,21)]);
mean_f =
    mean([flexion_features_train(:,1),flexion_features_train(:,5),flexion_features_train(:,9),flexion_features_train(:,13),flexion_features_train(:,17),flexion_features_train(:,21)]);

mean_g =
    mean([grasp_features_train(:,1),grasp_features_train(:,5),grasp_features_train(:,9),grasp_features_train(:,13),grasp_features_train(:,17),grasp_features_train(:,21)]);
mean_r =
    mean([releasing_features_train(:,1),releasing_features_train(:,5),releasing_features_train(:,9),releasing_features_train(:,13),releasing_features_train(:,17),releasing_features_train(:,21)]);
mean_p =
    mean([pronation_features_train(:,1),pronation_features_train(:,5),pronation_features_train(:,9),pronation_features_train(:,13),pronation_features_train(:,17),pronation_features_train(:,21)]);
mean_s =
    mean([supination_features_train(:,1),supination_features_train(:,5),supination_features_train(:,9),supination_features_train(:,13),supination_features_train(:,17),supination_features_train(:,21)]);
mean_e =
    mean([extension_features_train(:,1),extension_features_train(:,5),extension_features_train(:,9),extension_features_train(:,13),extension_features_train(:,17),extension_features_train(:,21)]);
mean_f =
    mean([flexion_features_train(:,1),flexion_features_train(:,5),flexion_features_train(:,9),flexion_features_train(:,13),flexion_features_train(:,17),flexion_features_train(:,21)]);
cov_rest =
cov([rest_features_train(:,1),rest_features_train(:,5),rest_features_train(:,9),rest_features_train(:,13),rest_features_train(:,17),rest_features_train(:,21)]);

%Testing Rest

correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)

test =
[rest_features_test(a,1),rest_features_test(a,5),rest_features_test(a,9),rest_features_test(a,13),rest_features_test(a,17),rest_features_test(a,21)];

w_rest = [test-mean_rest]'*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);

w_g = [test-mean_gl]'*inv(cov_g)*[test-mean_gl];
prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);

w_r = [test-mean_rl]'*inv(cov_r)*[test-mean_rl];
prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);

w_p = [test-mean_pl]'*inv(cov_p)*[test-mean_pl];
prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);

w_s = [test-mean_sl]'*inv(cov_s)*[test-mean_sl];
prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);

w_e = [test-mean_el]'*inv(cov_e)*[test-mean_el];
prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);

w_f = [test-mean_fl]'*inv(cov_f)*[test-mean_fl];
prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)
correct_rest = correct_rest + 1;
else
wrong_rest = wrong_rest + 1;
end
end
results(1,1) = correct_rest / (correct_rest + wrong_rest);

%Testing Grasping

correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)

test =
[grasp_features_test(a,1),grasp_features_test(a,5),grasp_features_test(a,9),grasp_features_test(a,13),grasp_features_test(a,17),grasp_features_test(a,21)];

w_rest = [test-mean_rest]'*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);

w_g = [test-mean_gl]'*inv(cov_g)*[test-mean_gl];
prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);

w_r = [test-mean_rl]'*inv(cov_r)*[test-mean_rl];
prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);

w_p = [test-mean_pl]'*inv(cov_p)*[test-mean_pl];
prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);

w_s = [test-mean_sl]'*inv(cov_s)*[test-mean_sl];
prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);

w_e = [test-mean_el]'*inv(cov_e)*[test-mean_el];
prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);

w_f = [test-mean_fl]'*inv(cov_f)*[test-mean_fl];
prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
correct_g = correct_g + 1;
else
wrong_g = wrong_g + 1;
end
results(1,2) = correct_g / (correct_g + wrong_g);
end
end
results(1,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing

correct_r = 0;
wrong_r = 0;
for a = 1:size(releasing_features_test,1)
    test = [releasing_features_test(a,1),releasing_features_test(a,5),releasing_features_test(a,9),releasing_features_test(a,13),releasing_features_test(a,17),releasing_features_test(a,21)];

    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

    if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
        correct_r = correct_r + 1;
    else
        wrong_r = wrong_r + 1;
    end
end
end
results(1,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation

correct_p = 0;
wrong_p = 0;
for a = 1:size(pronation_features_test,1)
    test = [pronation_features_test(a,1),pronation_features_test(a,5),pronation_features_test(a,9),pronation_features_test(a,13),pronation_features_test(a,17),pronation_features_test(a,21)];

    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

    if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
        correct_p = correct_p + 1;
    else
        wrong_p = wrong_p + 1;
    end
end
end

155
results(1,4) = correct_p / (correct_p + wrong_p);

% Testing Supination
correct_s = 0;
wrong_s = 0;
for a = 1:size(supination_features_test,1)
    test =
        [supination_features_test(a,1),supination_features_test(a,5),supination_features_test(a,9),supination_features_test(a,13),supination_features_test(a,17),supination_features_test(a,21)];
        w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
        prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
        w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
        prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
        w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
        prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
        w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
        prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
        w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
        prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
        w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
        prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
        w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
        prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
        if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
            correct_s = correct_s + 1;
        else
            wrong_s = wrong_s + 1;
        end
    end
results(1,5) = correct_s / (correct_s + wrong_s);
% Testing Extension
correct_e = 0;
wrong_e = 0;
for a = 1:size(extension_features_test,1)
    test =
        [extension_features_test(a,1),extension_features_test(a,5),extension_features_test(a,9),extension_features_test(a,13),extension_features_test(a,17),extension_features_test(a,21)];
        w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
        prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
        w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
        prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
        w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
        prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
        w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
        prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
        w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
        prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
        w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
        prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
        w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
        prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
        if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
            correct_e = correct_e + 1;
        else
            wrong_e = wrong_e + 1;
        end
    end
results(1,6) = correct_e / (correct_e + wrong_e);
correct_f = 0;
wrong_f = 0;
for a = 1:size(flexion_features_test,1)
    test =
    [flexion_features_test(a,1),flexion_features_test(a,5),flexion_features_test(a,9),flexion_features_test(a,13),flexion_features_test(a,17),flexion_features_test(a,21)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_f>prob_rest && prob_f>prob_g && prob_f>prob_r && prob_f>prob_p && prob_f>prob_s && prob_f>prob_e)
        correct_f = correct_f + 1;
    else
        wrong_f = wrong_f + 1;
    end
end
results(1,7) = correct_f / (correct_f + wrong_f);
results(1,8) =
    (correct_g+correct_r+correct_p+correct_s+correct_rest)/(wrong_g+wrong_r+wrong_p+wrong_s+wrong_rest+correct_g+correct_r+correct_p+correct_s+correct_rest);

% Calculation of the mean vectors and covariance matrices for using 2
% autoregressive coefficients as features
mean_g =
    mean([grasp_features_train(:,1:2),grasp_features_train(:,5:6),grasp_features_train(:,9:10),grasp_features_train(:,13:14),grasp_features_train(:,17:18),grasp_features_train(:,21:22)]);
mean_r =
    mean([releasing_features_train(:,1:2),releasing_features_train(:,5:6),releasing_features_train(:,9:10),releasing_features_train(:,13:14),releasing_features_train(:,17:18),releasing_features_train(:,21:22)]);
mean_p =
    mean([pronation_features_train(:,1:2),pronation_features_train(:,5:6),pronation_features_train(:,9:10),pronation_features_train(:,13:14),pronation_features_train(:,17:18),pronation_features_train(:,21:22)]);
mean_s =
    mean([supination_features_train(:,1:2),supination_features_train(:,5:6),supination_features_train(:,9:10),supination_features_train(:,13:14),supination_features_train(:,17:18),supination_features_train(:,21:22)]);
mean_e =
    mean([extension_features_train(:,1:2),extension_features_train(:,5:6),extension_features_train(:,9:10),extension_features_train(:,13:14),extension_features_train(:,17:18),extension_features_train(:,21:22)]);
mean_f =
    mean([flexion_features_train(:,1:2),flexion_features_train(:,5:6),flexion_features_train(:,9:10),flexion_features_train(:,13:14),flexion_features_train(:,17:18),flexion_features_train(:,21:22)]);
mean_rest =
    mean([rest_features_train(:,1:2),rest_features_train(:,5:6),rest_features_train(:,9:10),rest_features_train(:,13:14),rest_features_train(:,17:18),rest_features_train(:,21:22)]);
cov_g =
    cov([grasp_features_train(:,1:2),grasp_features_train(:,5:6),grasp_features_train(:,9:10),grasp_features_train(:,13:14),grasp_features_train(:,17:18),grasp_features_train(:,21:22)]);
cov_r =
    cov([releasing_features_train(:,1:2),releasing_features_train(:,5:6),releasing_features_train(:,9:10),releasing_features_train(:,13:14),releasing_features_train(:,17:18),releasing_features_train(:,21:22)]);
cov_p =
cov(pronation_features_train(:,1:2),pronation_features_train(:,5:6),pronation_features_train(:,9:10),pronation_features_train(:,13 :14),pronation_features_train(:,17:18),pronation_features_train(:,21:22));
cov_s =
cov(supination_features_train(:,1:2),supination_features_train(:,5:6),supination_features_train(:,9:10),supination_features_train( :,13:14),supination_features_train(:,17:18),supination_features_train(:,21:22));
cov_e =
cov(extension_features_train(:,1:2),extension_features_train(:,5:6),extension_features_train(:,9:10),extension_features_train(:,13 :14),extension_features_train(:,17:18),extension_features_train(:,21:22));
cov_f =
cov(flexion_features_train(:,1:2),flexion_features_train(:,5:6),flexion_features_train(:,9:10),flexion_features_train(:,13:14),flexion_features_train(:,17:18),flexion_features_train(:,21:22));
cov_rest =
cov(rest_features_train(:,1:2),rest_features_train(:,5:6),rest_features_train(:,9:10),rest_features_train(:,13:14),rest_features_train(:,17:18),rest_features_train(:,21:22));

%Testing Rest

correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)

test =
[rest_features_test(a,1:2),rest_features_test(a,5:6),rest_features_test(a,9:10),rest_features_test(a,13:14),rest_features_test(a,17:18 ),rest_features_test(a,21:22)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);

w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);

w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);

w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);

w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);

w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);

w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)

correct_rest = correct_rest + 1;
else

wrong_rest = wrong_rest + 1;
end
end
results(2,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping

correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)

test =
grasp_features_test(a,1:2),grasp_features_test(a,5:6),grasp_features_test(a,9:10),grasp_features_test(a,13:14),grasp_features_test(a,17:18 ),grasp_features_test(a,21:22)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);

w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);

w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);

w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
\[ \text{prob}_p = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_p)|^{1/2}} \exp(-0.5 w_p); \]
\[ w_s = ([\text{test-mean}_s]'\text{inv}(\text{cov}_s)'[\text{test-mean}_s]); \]
\[ \text{prob}_s = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_s)|^{1/2}} \exp(-0.5 w_s); \]
\[ w_e = ([\text{test-mean}_e]'\text{inv}(\text{cov}_e)'[\text{test-mean}_e]); \]
\[ \text{prob}_e = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_e)|^{1/2}} \exp(-0.5 w_e); \]
\[ w_f = ([\text{test-mean}_f]'\text{inv}(\text{cov}_f)'[\text{test-mean}_f]); \]
\[ \text{prob}_f = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_f)|^{1/2}} \exp(-0.5 w_f); \]

\[
\text{if}(\text{prob}_g > \text{prob}_\text{rest} \&\& \text{prob}_g > \text{prob}_p \&\& \text{prob}_g > \text{prob}_s \&\& \text{prob}_g > \text{prob}_e \&\& \text{prob}_g > \text{prob}_f)
\text{correct}_g = \text{correct}_g + 1;
\text{else}
\text{wrong}_g = \text{wrong}_g + 1;
\text{end}
\end{cases}
\]
\[
\text{results}(2,2) = \frac{\text{correct}_g}{\text{correct}_g + \text{wrong}_g};
\]

\[
\% \text{Testing Releasing}
\]
\[
\text{correct}_r = 0;
\text{wrong}_r = 0;
\text{for } a = 1: \text{size}([\text{releasing_features_test}(a,1:2), \text{releasing_features_test}(a,5:6), \text{releasing_features_test}(a,9:10), \text{releasing_features_test}(a,13:14), \text{releasing_features_test}(a,17:18), \text{releasing_features_test}(a,21:22)]);
\text{w_rest} = ([\text{test-mean}_\text{rest}]'\text{inv}(\text{cov}_\text{rest})'[\text{test-mean}_\text{rest}]);
\text{prob_rest} = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_\text{rest})|^{1/2}} \exp(-0.5 w_{\text{rest}}); \]
\[ w_g = ([\text{test-mean}_g]'\text{inv}(\text{cov}_g)'[\text{test-mean}_g]); \]
\[ \text{prob}_g = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_g)|^{1/2}} \exp(-0.5 w_g); \]
\[ w_r = ([\text{test-mean}_r]'\text{inv}(\text{cov}_r)'[\text{test-mean}_r]); \]
\[ \text{prob}_r = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_r)|^{1/2}} \exp(-0.5 w_r); \]
\[ w_p = ([\text{test-mean}_p]'\text{inv}(\text{cov}_p)'[\text{test-mean}_p]); \]
\[ \text{prob}_p = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_p)|^{1/2}} \exp(-0.5 w_p); \]
\[ w_s = ([\text{test-mean}_s]'\text{inv}(\text{cov}_s)'[\text{test-mean}_s]); \]
\[ \text{prob}_s = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_s)|^{1/2}} \exp(-0.5 w_s); \]
\[ w_e = ([\text{test-mean}_e]'\text{inv}(\text{cov}_e)'[\text{test-mean}_e]); \]
\[ \text{prob}_e = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_e)|^{1/2}} \exp(-0.5 w_e); \]
\[ w_f = ([\text{test-mean}_f]'\text{inv}(\text{cov}_f)'[\text{test-mean}_f]); \]
\[ \text{prob}_f = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_f)|^{1/2}} \exp(-0.5 w_f); \]

\[
\text{if}(\text{prob}_r > \text{prob}_\text{rest} \&\& \text{prob}_r > \text{prob}_g \&\& \text{prob}_r > \text{prob}_p \&\& \text{prob}_r > \text{prob}_s \&\& \text{prob}_r > \text{prob}_e \&\& \text{prob}_r > \text{prob}_f)
\text{correct}_r = \text{correct}_r + 1;
\text{else}
\text{wrong}_r = \text{wrong}_r + 1;
\text{end}
\end{cases}
\]
\[
\text{results}(2,3) = \frac{\text{correct}_r}{\text{correct}_r + \text{wrong}_r};
\]

\[
\% \text{Testing Pronation}
\]
\[
\text{correct}_p = 0;
\text{wrong}_p = 0;
\text{for } a = 1: \text{size}([\text{pronation_features_test}(a,1:2), \text{pronation_features_test}(a,5:6), \text{pronation_features_test}(a,9:10), \text{pronation_features_test}(a,13:14), \text{pronation_features_test}(a,17:18), \text{pronation_features_test}(a,21:22)]);
\text{w_rest} = ([\text{test-mean}_\text{rest}]'\text{inv}(\text{cov}_\text{rest})'[\text{test-mean}_\text{rest}]);
\text{prob_rest} = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_\text{rest})|^{1/2}} \exp(-0.5 w_{\text{rest}}); \]
\[ w_g = ([\text{test-mean}_g]'\text{inv}(\text{cov}_g)'[\text{test-mean}_g]); \]
\[ \text{prob}_g = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_g)|^{1/2}} \exp(-0.5 w_g); \]
\[ w_r = ([\text{test-mean}_r]'\text{inv}(\text{cov}_r)'[\text{test-mean}_r]); \]
\[ \text{prob}_r = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_r)|^{1/2}} \exp(-0.5 w_r); \]
\[ w_p = ([\text{test-mean}_p]'\text{inv}(\text{cov}_p)'[\text{test-mean}_p]); \]
\[ \text{prob}_p = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_p)|^{1/2}} \exp(-0.5 w_p); \]
\[ w_s = ([\text{test-mean}_s]'\text{inv}(\text{cov}_s)'[\text{test-mean}_s]); \]
\[ \text{prob}_s = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_s)|^{1/2}} \exp(-0.5 w_s); \]
\[ w_e = ([\text{test-mean}_e]'\text{inv}(\text{cov}_e)'[\text{test-mean}_e]); \]
\[ \text{prob}_e = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_e)|^{1/2}} \exp(-0.5 w_e); \]
\[ w_f = ([\text{test-mean}_f]'\text{inv}(\text{cov}_f)'[\text{test-mean}_f]); \]
\[ \text{prob}_f = \frac{1}{(2\pi)^{24/2}|\text{det}(\text{cov}_f)|^{1/2}} \exp(-0.5 w_f); \]

\[
\text{if}(\text{prob}_p > \text{prob}_\text{rest} \&\& \text{prob}_p > \text{prob}_g \&\& \text{prob}_p > \text{prob}_r \&\& \text{prob}_p > \text{prob}_s \&\& \text{prob}_p > \text{prob}_e \&\& \text{prob}_p > \text{prob}_f)
\text{correct}_p = \text{correct}_p + 1;
\text{else}
\text{wrong}_p = \text{wrong}_p + 1;
\text{end}
\end{cases}
\]
\[
\text{results}(2,4) = \frac{\text{correct}_p}{\text{correct}_p + \text{wrong}_p};
\]
prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
 prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
 prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
correct_p = correct_p + 1;
else
  wrong_p = wrong_p + 1;
end
end
results(2,4) = correct_p / (correct_p + wrong_p);

% Testing Supination

% Testing Extension

for a = 1:size(extension_features_test,1)
test = 
  [extension_features_test(a,1:2),extension_features_test(a,5:6),extension_features_test(a,9:10),extension_features_test(a,13:14),extension_features_test(a,17:18),extension_features_test(a,21:22)];
\[
\text{prob}_e = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_e))^{1/2}}\exp(-0.5w_e);
\]
\[
w_f = [\text{test-mean}_f]^{\text{inv}}(\text{cov}_f)[\text{test-mean}_f];
\]
\[
\text{prob}_f = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_f))^{1/2}}\exp(-0.5w_f);
\]

```matlab
if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
correct_e = correct_e + 1;
else
    wrong_e = wrong_e + 1;
end
end
results(2,6) = correct_e / (correct_e + wrong_e);

% Testing Flexion
correct_f = 0;
wrong_f = 0;
for a = 1:size(flexion_features_test,1)
test = [flexion_features_test(a,1:2),flexion_features_test(a,5:6),flexion_features_test(a,9:10),flexion_features_test(a,13:14),flexion_features_test(a,17:18),flexion_features_test(a,21:22)];

w_rest = [test-mean_rest]^{\text{inv}}(\text{cov}_rest)[test-mean_rest];
\text{prob}_rest = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_rest))^{1/2}}\exp(-0.5w_rest);
\text{w}_g = [test-mean_g]^{\text{inv}}(\text{cov}_g)[test-mean_g];
\text{prob}_g = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_g))^{1/2}}\exp(-0.5w_g);
\text{w}_r = [test-mean_r]^{\text{inv}}(\text{cov}_r)[test-mean_r];
\text{prob}_r = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_r))^{1/2}}\exp(-0.5w_r);
\text{w}_p = [test-mean_p]^{\text{inv}}(\text{cov}_p)[test-mean_p];
\text{prob}_p = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_p))^{1/2}}\exp(-0.5w_p);
\text{w}_s = [test-mean_s]^{\text{inv}}(\text{cov}_s)[test-mean_s];
\text{prob}_s = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_s))^{1/2}}\exp(-0.5w_s);
\text{w}_e = [test-mean_e]^{\text{inv}}(\text{cov}_e)[test-mean_e];
\text{prob}_e = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_e))^{1/2}}\exp(-0.5w_e);
\text{w}_f = [test-mean_f]^{\text{inv}}(\text{cov}_f)[test-mean_f];
\text{prob}_f = \frac{1}{(2\pi)^{24/2}(\det(\text{cov}_f))^{1/2}}\exp(-0.5w_f);

if(prob_f>prob_rest && prob_f>prob_g && prob_f>prob_r && prob_f>prob_p && prob_f>prob_s && prob_f>prob_e)
correct_f = correct_f + 1;
else
    wrong_f = wrong_f + 1;
end
end
results(2,7) = correct_f / (correct_f + wrong_f);
results(2,8) =
\frac{(\text{correct}_g+\text{correct}_r+\text{correct}_p+\text{correct}_s+\text{correct}_rest)}{(\text{wrong}_g+\text{wrong}_r+\text{wrong}_p+\text{wrong}_s+\text{wrong}_rest+\text{correct}_g+\text{correct}_r+\text{correct}_p+\text{correct}_s+\text{correct}_rest)};

% Calculation of the mean vectors and covariance matrices for using 3
% autoregressive coefficients as features
mean_g = mean([\text{grasp_features_train(:,1:3),grasp_features_train(:,5:7),grasp_features_train(:,9:11),grasp_features_train(:,13:15),grasp_features_train(:,17:19),grasp_features_train(:,21:23)}]);
mean_r = mean([\text{releasing_features_train(:,1:3),releasing_features_train(:,5:7),releasing_features_train(:,9:11),releasing_features_train(:,13:15),releasing_features_train(:,17:19),releasing_features_train(:,21:23)}]);
mean_p = mean([\text{pronation_features_train(:,1:3),pronation_features_train(:,5:7),pronation_features_train(:,9:11),pronation_features_train(:,13:15),pronation_features_train(:,17:19),pronation_features_train(:,21:23)}]);
mean_s = mean([\text{supination_features_train(:,1:3),supination_features_train(:,5:7),supination_features_train(:,9:11),supination_features_train(:,13:15),supination_features_train(:,17:19),supination_features_train(:,21:23)}]);
mean_e = mean([\text{extension_features_train(:,1:3),extension_features_train(:,5:7),extension_features_train(:,9:11),extension_features_train(:,13:15),extension_features_train(:,17:19),extension_features_train(:,21:23)}]);
```
mean_f = mean([flexion_features_train(:,1:3),flexion_features_train(:,5:7),flexion_features_train(:,9:11),flexion_features_train(:,13:15),flexion_features_train(:,17:19),flexion_features_train(:,21:23)]);
mean_rest = mean([rest_features_train(:,1:3),rest_features_train(:,5:7),rest_features_train(:,9:11),rest_features_train(:,13:15),rest_features_train(:,17:19),rest_features_train(:,21:23)]);
cov_g = cov([grasp_features_train(:,1:3),grasp_features_train(:,5:7),grasp_features_train(:,9:11),grasp_features_train(:,13:15),grasp_features_train(:,17:19),grasp_features_train(:,21:23)]);
cov_r = cov([releasing_features_train(:,1:3),releasing_features_train(:,5:7),releasing_features_train(:,9:11),releasing_features_train(:,13:15),releasing_features_train(:,17:19),releasing_features_train(:,21:23)]);
cov_p = cov([pronation_features_train(:,1:3),pronation_features_train(:,5:7),pronation_features_train(:,9:11),pronation_features_train(:,13:15),pronation_features_train(:,17:19),pronation_features_train(:,21:23)]);
cov_s = cov([supination_features_train(:,1:3),supination_features_train(:,5:7),supination_features_train(:,9:11),supination_features_train(:,13:15),supination_features_train(:,17:19),supination_features_train(:,21:23)]);
cov_e = cov([extension_features_train(:,1:3),extension_features_train(:,5:7),extension_features_train(:,9:11),extension_features_train(:,13:15),extension_features_train(:,17:19),extension_features_train(:,21:23)]);
cov_f = cov([flexion_features_train(:,1:3),flexion_features_train(:,5:7),flexion_features_train(:,9:11),flexion_features_train(:,13:15),flexion_features_train(:,17:19),flexion_features_train(:,21:23)]);
cov_rest = cov([rest_features_train(:,1:3),rest_features_train(:,5:7),rest_features_train(:,9:11),rest_features_train(:,13:15),rest_features_train(:,17:19),rest_features_train(:,21:23)]);

% Testing Rest
correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)
    test = [rest_features_test(a,1:3),rest_features_test(a,5:7),rest_features_test(a,9:11),rest_features_test(a,13:15),rest_features_test(a,17:19),rest_features_test(a,21:23)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);
    if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)
        correct_rest = correct_rest + 1;
    else
        wrong_rest = wrong_rest + 1;
    end
end
results(3,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping
correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)
    test = [grasp_features_test(a,1:3),grasp_features_test(a,5:7),grasp_features_test(a,9:11),grasp_features_test(a,13:15),grasp_features_test(a,17:19),grasp_features_test(a,21:23)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
        correct_g = correct_g + 1;
    else
        wrong_g = wrong_g + 1;
    end
end
results(3,2) = correct_g / (correct_g + wrong_g);

%Testing Releasing
correct_r = 0;
wrong_r = 0;
for a = 1:size(releasing_features_test,1)
    test = [releasing_features_test(a,1:3),releasing_features_test(a,5:7),releasing_features_test(a,9:11),releasing_features_test(a,13:15),releasing_features_test(a,17:19),releasing_features_test(a,21:23)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
        correct_r = correct_r + 1;
    else
        wrong_r = wrong_r + 1;
    end
end
results(3,3) = correct_r / (correct_r + wrong_r);

%Testing Pronation
correct_p = 0;
wrong_p = 0;
for a = 1:size(pronation_features_test,1)
test =
[pronation_features_test(a,1:3),pronation_features_test(a,5:7),pronation_features_test(a,9:11),pronation_features_test(a,13:15),pronation_features_test(a,17:19),pronation_features_test(a,21:23)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
correct_p = correct_p + 1;
else
wrong_p = wrong_p + 1;
end
end
results(3,4) = correct_p / (correct_p + wrong_p);

%Testing Supination

correct_s = 0;
wrong_s = 0;
for a = 1:size(supination_features_test,1)
test =
[supination_features_test(a,1:3),supination_features_test(a,5:7),supination_features_test(a,9:11),supination_features_test(a,13:15),supination_features_test(a,17:19),supination_features_test(a,21:23)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
correct_s = correct_s + 1;
else
wrong_s = wrong_s + 1;
end
end
results(3,5) = correct_s / (correct_s + wrong_s);

%Testing Extension

correct_e = 0;
wrong_e = 0;
for a = 1:size(extension_features_test,1)
test = [extension_features_test(a,1:3),extension_features_test(a,5:7),extension_features_test(a,9:11),extension_features_test(a,13:15),extension_features_test(a,17:19),extension_features_test(a,21:23)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
correct_e = correct_e + 1;
else
wrong_e = wrong_e + 1;
end
end
results(3,6) = correct_e / (correct_e + wrong_e);

% Testing Flexion

correct_f = 0;
wrong_f = 0;
for a = 1:size(flexion_features_test,1)
test = [flexion_features_test(a,1:3),flexion_features_test(a,5:7),flexion_features_test(a,9:11),flexion_features_test(a,13:15),flexion_features_test(a,17:19),flexion_features_test(a,21:23)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

if(prob_f>prob_rest && prob_f>prob_g && prob_f>prob_r && prob_f>prob_p && prob_f>prob_s && prob_f>prob_e)
correct_f = correct_f + 1;
else
wrong_f = wrong_f + 1;
end
end
results(3,7) = correct_f / (correct_f + wrong_f);
results(3,8) = (correct_g+correct_r+correct_p+correct_s+correct_rest)/(wrong_g+wrong_r+wrong_p+wrong_s+wrong_rest+correct_g+correct_r+correct_p+correct_s+correct_rest);

% Calculation of the mean vectors and covariance matrices for using all 4 autoregressive coefficients as features
mean_g = mean([grasp_features_train]);
mean_r = mean([releasing_features_train]);
mean_p = mean([pronation_features_train]);
mean_s = mean([supination_features_train]);
mean_e = mean([extension_features_train]);
mean_f = mean([flexion_features_train]);
mean_rest = mean([rest_features_train]);
cov_g = cov([grasp_features_train]);
cov_r = cov([releasing_features_train]);
cov_p = cov([pronation_features_train]);
cov_s = cov([supination_features_train]);
cov_e = cov([extension_features_train]);
cov_f = cov([flexion_features_train]);
cov_rest = cov([rest_features_train]);

%Testing Rest
correct_rest = 0;
wrong_rest = 0;
mv_rest = zeros(size(rest_features_test,1),1);
for a = 1:size(rest_features_test,1)
    test = [rest_features_test(a,:)];
w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
prob_rest = 1/(2*pi)^(24/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);  
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
prob_g = 1/(2*pi)^(24/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);  
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
prob_r = 1/(2*pi)^(24/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);  
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
prob_p = 1/(2*pi)^(24/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);  
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
prob_s = 1/(2*pi)^(24/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);  
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
prob_e = 1/(2*pi)^(24/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);  
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
prob_f = 1/(2*pi)^(24/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);
if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
    prob_rest>prob_f)
    correct_rest = correct_rest + 1;
else
    wrong_rest = wrong_rest + 1;
end

if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
    prob_rest>prob_f)
    mv_rest(a) = 0;
elseif(prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
    mv_rest(a) = 1;
elseif(prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
    mv_rest(a) = 2;
elseif(prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
    mv_rest(a) = 3;
elseif(prob_s>prob_e && prob_s>prob_f)
    mv_rest(a) = 4;
elseif(prob_e>prob_f)
    mv_rest(a) = 5;
else
    mv_rest(a) = 6;
end
results(4,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping
correct_g = 0;
wrong_g = 0;
mv_g = zeros(size(grasp_features_test,1),1);
for a = 1:size(grasp_features_test,1)
    test = [grasp_features_test(a,:)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
        correct_g = correct_g + 1;
        end
    else
        wrong_g = wrong_g + 1;
        end
if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
    mv_g(a) = 1;
else
    mv_g(a) = 0;
end
if(prob_r>prob_p && prob_r>prob_s && prob_r>prob_rest && prob_r>prob_e && prob_r>prob_f)
    mv_g(a) = 2;
else
    mv_g(a) = 3;
end
if(prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
    mv_g(a) = 4;
else
    mv_g(a) = 5;
end
else
    mv_g(a) = 6;
end
results(4,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing
correct_r = 0;
wrong_r = 0;
mv_r = zeros(size(releasing_features_test,1),1);
for a = 1:size(releasing_features_test,1)
    test = [releasing_features_test(a,:)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
    correct_g = correct_g + 1;
else
    wrong_g = wrong_g + 1;
end
if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
    mv_r(a) = 1;
else
    mv_r(a) = 0;
end
if(prob_r>prob_p && prob_r>prob_s && prob_r>prob_rest && prob_r>prob_e && prob_r>prob_f)
    mv_r(a) = 2;
else
    mv_r(a) = 3;
end
if(prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
    mv_r(a) = 4;
else
    mv_r(a) = 5;
end
else
    mv_r(a) = 6;
end
results(4,2) = correct_r / (correct_r + wrong_r);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^((24/2))*(det(cov_s))^(1/2))*exp(-0.5*w_s);

w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^((24/2))*(det(cov_e))^(1/2))*exp(-0.5*w_e);

w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^((24/2))*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_e && prob_r>prob_s && prob_r>prob_f)
correct_r = correct_r + 1;
else
wrong_r = wrong_r + 1;
end

if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_f)
mv_r(a) = 2;
elseif(prob_rest>prob_g && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)
mv_r(a) = 0;
elseif(prob_g>prob_p && prob_g>prob_e && prob_g>prob_f)
mv_r(a) = 1;
elseif(prob_p>prob_e && prob_p>prob_f)
mv_r(a) = 3;
elseif(prob_e>prob_f)
mv_r(a) = 5;
else
mv_r(a) = 6;
end

results(4,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation

correct_p = 0;
wrong_p = 0;

mv_p = zeros(size(pronation_features_test,1),1);
for a = 1:size(pronation_features_test,1)
test = [pronation_features_test(a,:)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^((24/2))*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);

w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/((2*pi)^((24/2))*(det(cov_g))^(1/2))*exp(-0.5*w_g);

w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/((2*pi)^((24/2))*(det(cov_r))^(1/2))*exp(-0.5*w_r);

w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/((2*pi)^((24/2))*(det(cov_p))^(1/2))*exp(-0.5*w_p);

w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^((24/2))*(det(cov_s))^(1/2))*exp(-0.5*w_s);

w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^((24/2))*(det(cov_e))^(1/2))*exp(-0.5*w_e);

w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^((24/2))*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
correct_p = correct_p + 1;
else
wrong_p = wrong_p + 1;
end

if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)

mv_p(a) = 3;
elseif(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)

mv_p(a) = 0;
elseif(prob_g>prob_r && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
  mv_p(a) = 1;
elseif(prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
  mv_p(a) = 2;
elseif(prob_s>prob_e && prob_s>prob_f)
  mv_p(a) = 4;
elseif(prob_e>prob_f)
  mv_p(a) = 5;
else
  mv_p(a) = 6;
end
end
results(4,4) = correct_p / (correct_p + wrong_p);

%Testing Supination

correct_s = 0;
wrong_s = 0;

mv_s = zeros(size(supination_features_test,1),1);

for a = 1:size(supination_features_test,1)
  test = [supination_features_test(a,:)];
  w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
  prob_rest = 1/((2*pi)^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
  w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
  prob_g = 1/((2*pi)^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
  w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
  prob_r = 1/((2*pi)^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
  w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
  prob_p = 1/((2*pi)^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
  w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
  prob_s = 1/((2*pi)^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
  w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
  prob_e = 1/((2*pi)^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
  w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
  prob_f = 1/((2*pi)^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
  if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
    correct_s = correct_s + 1;
  else
    wrong_s = wrong_s + 1;
  end
end

if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
  mv_s(a) = 4;
elseif(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_e && prob_rest>prob_f)
  mv_s(a) = 0;
elseif(prob_g>prob_r && prob_g>prob_p && prob_g>prob_e && prob_g>prob_f)
  mv_s(a) = 1;
elseif(prob_r>prob_p && prob_r>prob_e && prob_r>prob_f)
  mv_s(a) = 2;
elseif(prob_p>prob_e && prob_p>prob_f)
  mv_s(a) = 3;
elseif(prob_e>prob_f)
  mv_s(a) = 5;
else
  mv_s(a) = 6;
end
end

results(4,5) = correct_s / (correct_s + wrong_s);

%Testing Extension

correct_e = 0;

wrong_e = 0;


```matlab
mv_e = zeros(size(extension_features_test,1),1);
for a = 1:size(extension_features_test,1)
    test = [extension_features_test(a,:)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if (prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
        correct_e = correct_e + 1;
    else
        wrong_e = wrong_e + 1;
    end
    mv_e(a) = 5;
elseif(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_f)
    mv_e(a) = 0;
elseif(prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_f)
    mv_e(a) = 1;
elseif(prob_r>prob_p && prob_r>prob_s && prob_r>prob_f)
    mv_e(a) = 2;
elseif(prob_p>prob_s && prob_p>prob_f)
    mv_e(a) = 3;
else
    mv_e(a) = 6;
end
results(4,6) = correct_e / (correct_e + wrong_e);
%Testing Flexion
for a = 1:size(flexion_features_test,1)
    test = [flexion_features_test(a,:)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi^(24/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi^(24/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi^(24/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi^(24/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi^(24/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi^(24/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi^(24/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
end
results(4,6) = correct_e / (correct_e + wrong_e);
end
```
\[ \text{prob}_f = \frac{1}{((2\pi)^{(24/2)} \cdot \text{det}(\text{cov}_f)^{(1/2)}) \cdot \exp(-0.5 \cdot w_f)}; \]

if(prob_f > prob_rest & & prob_f > prob_g & & prob_f > prob_r & & prob_f > prob_p & & prob_f > prob_s & & prob_f > prob_e)
    correct_f = correct_f + 1;
else
    wrong_f = wrong_f + 1;
end

if(prob_f > prob_rest & & prob_f > prob_g & & prob_f > prob_r & & prob_f > prob_p & & prob_f > prob_s & & prob_f > prob_e)
    mv_f(a) = 6;
elseif(prob_rest > prob_g & & prob_rest > prob_r & & prob_rest > prob_p & & prob_rest > prob_s & & prob_rest > prob_e)
    mv_f(a) = 1;
elseif(prob_g > prob_r & & prob_g > prob_p & & prob_g > prob_s & & prob_g > prob_e)
    mv_f(a) = 2;
elseif(prob_p > prob_s & & prob_p > prob_r)
    mv_f(a) = 3;
elseif(prob_s > prob_e)
    mv_f(a) = 4;
else
    mv_f(a) = 6;
end
end
results(4,7) = correct_f / (correct_f + wrong_f);
results(4,8) =
    (correct_g + correct_r + correct_p + correct_s + correct_rest) /
    (wrong_g + wrong_r + wrong_p + wrong_s + wrong_rest + correct_g + correct_r + correct_p + correct_s + correct_rest);

% Majority Voting
% This section performs majority voting on the classification of the 4AR features. The windows are 50ms and a total of 5 windows (250ms) is % considered in the majority vote. In the 5 windows whichever class was % classified the most times, results in that class being chosen.

% Testing Rest
correct_rest = 0;
wrong_rest = 0;
last_mv = 0;
for a = 5:length(mv_rest)
    rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;
if(mv_rest(a) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_rest(a) == 1)
    g_votes = g_votes + 1;
elseif(mv_rest(a) == 2)
    r_votes = r_votes + 1;
elseif(mv_rest(a) == 3)
    p_votes = p_votes + 1;
elseif(mv_rest(a) == 4)
    s_votes = s_votes + 1;
elseif(mv_rest(a) == 5)
    e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end
if(mv_rest(a-1) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_rest(a-1) == 1)
    g_votes = g_votes + 1;
else(mv_rest(a-1) == 2)
    r_votes = r_votes + 1;
elseif(mv_rest(a-1) == 3)
    p_votes = p_votes + 1;
elseif(mv_rest(a-1) == 4)
    s_votes = s_votes + 1;
elseif(mv_rest(a-1) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_rest(a-2) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_rest(a-2) == 1)
    g_votes = g_votes + 1;
else(mv_rest(a-2) == 2)
    r_votes = r_votes + 1;
elseif(mv_rest(a-2) == 3)
    p_votes = p_votes + 1;
elseif(mv_rest(a-2) == 4)
    s_votes = s_votes + 1;
elseif(mv_rest(a-2) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_rest(a-3) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_rest(a-3) == 1)
    g_votes = g_votes + 1;
else(mv_rest(a-3) == 2)
    r_votes = r_votes + 1;
elseif(mv_rest(a-3) == 3)
    p_votes = p_votes + 1;
elseif(mv_rest(a-3) == 4)
    s_votes = s_votes + 1;
elseif(mv_rest(a-3) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_rest(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_rest(a-4) == 1)
    g_votes = g_votes + 1;
else(mv_rest(a-4) == 2)
    r_votes = r_votes + 1;
elseif(mv_rest(a-4) == 3)
    p_votes = p_votes + 1;
elseif(mv_rest(a-4) == 4)
    s_votes = s_votes + 1;
elseif(mv_rest(a-4) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes && rest_votes>f_votes)
    correct_rest = correct_rest + 1;
    last_mv = 0;
elseif(g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes && g_votes>f_votes)
    wrong_rest = wrong_rest + 1;
    last_mv = 1;
elseif(r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes && r_votes>f_votes)
    wrong_rest = wrong_rest + 1;
    last_mv = 2;
elseif(p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes && p_votes>f_votes)
    wrong_rest = wrong_rest + 1;
    last_mv = 3;
elseif(s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes && s_votes>f_votes)
    wrong_rest = wrong_rest + 1;
    last_mv = 4;
elseif(e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes && e_votes>f_votes)
    wrong_rest = wrong_rest + 1;
    last_mv = 5;
elseif(f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>p_votes && f_votes>s_votes && f_votes>e_votes)
    wrong_rest = wrong_rest + 1;
    last_mv = 6;
elseif(last_mv == 0)
    correct_rest = correct_rest + 1;
    last_mv = 0;
else
    wrong_rest = wrong_rest + 1;
end
end
results(5,1) = correct_rest / (correct_rest + wrong_rest);
% Testing Grasping
correct_g = 0;
wrong_g = 0;
last_mv = 0;
for a = 5:length(mv_g)
    rest_votes = 0;
    g_votes = 0;
    r_votes = 0;
    p_votes = 0;
    s_votes = 0;
    e_votes = 0;
    f_votes = 0;
    if(mv_g(a) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_g(a) == 1)
        g_votes = g_votes + 1;
    elseif(mv_g(a) == 2)
        r_votes = r_votes + 1;
    elseif(mv_g(a) == 3)
        p_votes = p_votes + 1;
    elseif(mv_g(a) == 4)
        s_votes = s_votes + 1;
    elseif(mv_g(a) == 5)
e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if (mv_g(a-1) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_g(a-1) == 1)
    g_votes = g_votes + 1;
elseif (mv_g(a-1) == 2)
    r_votes = r_votes + 1;
elseif (mv_g(a-1) == 3)
    p_votes = p_votes + 1;
elseif (mv_g(a-1) == 4)
    s_votes = s_votes + 1;
elseif (mv_g(a-1) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if (mv_g(a-2) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_g(a-2) == 1)
    g_votes = g_votes + 1;
elseif (mv_g(a-2) == 2)
    r_votes = r_votes + 1;
elseif (mv_g(a-2) == 3)
    p_votes = p_votes + 1;
elseif (mv_g(a-2) == 4)
    s_votes = s_votes + 1;
elseif (mv_g(a-2) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if (mv_g(a-3) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_g(a-3) == 1)
    g_votes = g_votes + 1;
elseif (mv_g(a-3) == 2)
    r_votes = r_votes + 1;
elseif (mv_g(a-3) == 3)
    p_votes = p_votes + 1;
elseif (mv_g(a-3) == 4)
    s_votes = s_votes + 1;
elseif (mv_g(a-3) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if (mv_g(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_g(a-4) == 1)
    g_votes = g_votes + 1;
elseif (mv_g(a-4) == 2)
    r_votes = r_votes + 1;
elseif (mv_g(a-4) == 3)
    p_votes = p_votes + 1;
elseif (mv_g(a-4) == 4)
    s_votes = s_votes + 1;
elseif (mv_g(a-4) == 5)
    e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end

if(rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes) & & rest_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 0;
elseif(rest_votes>g_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes) & & g_votes>f_votes)
    correct_g = correct_g + 1;
    last_mv = 1;
elseif(rest_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes) & & r_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 2;
elseif(p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes)
    correct_g = correct_g + 1;
    last_mv = 3;
elseif(s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes)
    wrong_g = wrong_g + 1;
    last_mv = 4;
elseif(e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes)
    wrong_g = wrong_g + 1;
    last_mv = 5;
elseif(f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>s_votes && f_votes>e_votes)
    wrong_g = wrong_g + 1;
    last_mv = 6;
elseif(last_mv == 1)
    correct_g = correct_g + 1;
    last_mv = 0;
else
    wrong_g = wrong_g + 1;
end
end
results(5,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing
correct_r = 0;
wrong_r = 0;
last_mv = 0;
for a = 5:length(mv_r)
    rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;

if(mv_r(a) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_r(a) == 1)
    g_votes = g_votes + 1;
elseif(mv_r(a) == 2)
    r_votes = r_votes + 1;
elseif(mv_r(a) == 3)
    p_votes = p_votes + 1;
elseif(mv_r(a) == 4)
    s_votes = s_votes + 1;
elseif(mv_r(a) == 5)
    e_votes = e_votes + 1;
elseif(mv_r(a) == 6)
    f_votes = f_votes + 1;
end
s_votes = s_votes + 1;
elseif (mv_r(a) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if (mv_r(a-1) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_r(a-1) == 1)
    g_votes = g_votes + 1;
elseif (mv_r(a-1) == 2)
    r_votes = r_votes + 1;
elseif (mv_r(a-1) == 3)
    p_votes = p_votes + 1;
elseif (mv_r(a-1) == 4)
    s_votes = s_votes + 1;
else
    e_votes = e_votes + 1;
endif
if (mv_r(a-2) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_r(a-2) == 1)
    g_votes = g_votes + 1;
elseif (mv_r(a-2) == 2)
    r_votes = r_votes + 1;
elseif (mv_r(a-2) == 3)
    p_votes = p_votes + 1;
elseif (mv_r(a-2) == 4)
    s_votes = s_votes + 1;
else
    e_votes = e_votes + 1;
endif
if (mv_r(a-3) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_r(a-3) == 1)
    g_votes = g_votes + 1;
elseif (mv_r(a-3) == 2)
    r_votes = r_votes + 1;
elseif (mv_r(a-3) == 3)
    p_votes = p_votes + 1;
elseif (mv_r(a-3) == 4)
    s_votes = s_votes + 1;
else
    e_votes = e_votes + 1;
endif
if (mv_r(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_r(a-4) == 1)
    g_votes = g_votes + 1;
elseif (mv_r(a-4) == 2)
    r_votes = r_votes + 1;
elseif (mv_r(a-4) == 3)
    p_votes = p_votes + 1;
elseif (mv_r(a-4) == 4)
    s_votes = s_votes + 1;
else
    e_votes = e_votes + 1;
endif
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end

if(rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes)
    wrong_r = wrong_r + 1;
    last_mv = 0;
elseif(g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes)
    wrong_r = wrong_r + 1;
    last_mv = 1;
elseif(r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes)
    correct_r = correct_r + 1;
    last_mv = 2;
elseif(p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes)
    wrong_r = wrong_r + 1;
    last_mv = 3;
elseif(s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes)
    wrong_r = wrong_r + 1;
    last_mv = 4;
elseif(e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes)
    wrong_r = wrong_r + 1;
    last_mv = 5;
elseif(f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>p_votes && f_votes>s_votes)
    wrong_r = wrong_r + 1;
    last_mv = 6;
elseif(last_mv == 2)
correct_r = correct_r + 1;
last_mv = 0;
else
wrong_r = wrong_r + 1;
end
end
results(5,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation

correct_p = 0;
wrong_p = 0;
last_mv = 0;
for a = 5:length(mv_p)
rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;
if(mv_p(a) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_p(a) == 1)
g_votes = g_votes + 1;
elseif(mv_p(a) == 2)
r_votes = r_votes + 1;
elseif(mv_p(a) == 3)
p_votes = p_votes + 1;
elseif(mv_p(a) == 4)
s_votes = s_votes + 1;
elseif(mv_p(a) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end

if(mv_p(a-1) == 0)
rest_votes = rest_votes + 1;
elseif(mv_p(a-1) == 1)
g_votes = g_votes + 1;
elseif(mv_p(a-1) == 2)
r_votes = r_votes + 1;
elseif(mv_p(a-1) == 3)
p_votes = p_votes + 1;
elseif(mv_p(a-1) == 4)
s_votes = s_votes + 1;
elseif(mv_p(a-1) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
endif

if(mv_p(a-2) == 0)
rest_votes = rest_votes + 1;
elseif(mv_p(a-2) == 1)
g_votes = g_votes + 1;
elseif(mv_p(a-2) == 2)
r_votes = r_votes + 1;
elseif(mv_p(a-2) == 3)
p_votes = p_votes + 1;
elseif(mv_p(a-2) == 4)
s_votes = s_votes + 1;
elseif(mv_p(a-2) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
endif

if(mv_p(a-3) == 0)
rest_votes = rest_votes + 1;
elseif(mv_p(a-3) == 1)
g_votes = g_votes + 1;
elseif(mv_p(a-3) == 2)
r_votes = r_votes + 1;
elseif(mv_p(a-3) == 3)
p_votes = p_votes + 1;
elseif(mv_p(a-3) == 4)
s_votes = s_votes + 1;
elseif(mv_p(a-3) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
endif

if(mv_p(a-4) == 0)
rest_votes = rest_votes + 1;
elseif(mv_p(a-4) == 1)
g_votes = g_votes + 1;
elseif(mv_p(a-4) == 2)
r_votes = r_votes + 1;
elseif(mv_p(a-4) == 3)
p_votes = p_votes + 1;
elseif(mv_p(a-4) == 4)
s_votes = s_votes + 1;
elseif (mv_p(a-4) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if (rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes && rest_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 0;
elseif (g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes && g_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 1;
elseif (r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes && r_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 2;
elseif (p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes && p_votes>f_votes)
    correct_p = correct_p + 1;
    last_mv = 3;
elseif (s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes && s_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 4;
elseif (e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes && e_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 5;
elseif (f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>p_votes && f_votes>s_votes && f_votes>e_votes)
    wrong_p = wrong_p + 1;
    last_mv = 6;
elseif (last_mv == 3)
    correct_p = correct_p + 1;
    last_mv = 0;
else
    wrong_r = wrong_r + 1;
end
end
results(5,4) = correct_p / (correct_p + wrong_p);

% Testing Supination

correct_s = 0;
wrong_s = 0;
last_mv = 0;
for a = 5:length(mv_s)

    rest_votes = 0;
    g_votes = 0;
    r_votes = 0;
    p_votes = 0;
    s_votes = 0;
    e_votes = 0;
    f_votes = 0;

    if (mv_s(a) == 0)
        rest_votes = rest_votes + 1;
    elseif (mv_s(a) == 1)
        g_votes = g_votes + 1;
    elseif (mv_s(a) == 2)

179
r_votes = r_votes + 1;
elseif(mv_s(a) == 3)
p_votes = p_votes + 1;
elseif(mv_s(a) == 4)
s_votes = s_votes + 1;
elseif(mv_s(a) == 5)
e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_s(a-1) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_s(a-1) == 1)
g_votes = g_votes + 1;
elseif(mv_s(a-1) == 2)
r_votes = r_votes + 1;
elseif(mv_s(a-1) == 3)
p_votes = p_votes + 1;
elseif(mv_s(a-1) == 4)
s_votes = s_votes + 1;
elseif(mv_s(a-1) == 5)
e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_s(a-2) == 0)
    rest Votes = rest Votes + 1;
elseif(mv_s(a-2) == 1)
g_votes = g_votes + 1;
elseif(mv_s(a-2) == 2)
r_votes = r_votes + 1;
elseif(mv_s(a-2) == 3)
p_votes = p_votes + 1;
elseif(mv_s(a-2) == 4)
s_votes = s_votes + 1;
elseif(mv_s(a-2) == 5)
e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_s(a-3) == 0)
    rest Votes = rest Votes + 1;
elseif(mv_s(a-3) == 1)
g_votes = g_votes + 1;
elseif(mv_s(a-3) == 2)
r_votes = r_votes + 1;
elseif(mv_s(a-3) == 3)
p_votes = p_votes + 1;
elseif(mv_s(a-3) == 4)
s_votes = s_votes + 1;
elseif(mv_s(a-3) == 5)
e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_s(a-4) == 0)
    rest Votes = rest Votes + 1;
elseif(mv_s(a-4) == 1)
g_votes = g_votes + 1;
elseif(mv_s(a-4) == 2)
r_votes = r_votes + 1;
elseif(mv_s(a-4) == 3)
p_votes = p_votes + 1;
elseif(mv_s(a-4) == 4)
    s_votes = s_votes + 1;
elseif(mv_s(a-4) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(rest_votes > g_votes && rest_votes > r_votes && rest_votes > p_votes && rest_votes > s_votes && rest_votes > e_votes && rest_votes > f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 0;
elseif(g_votes > rest_votes && g_votes > r_votes && g_votes > p_votes && g_votes > s_votes && g_votes > e_votes && g_votes > f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 1;
elseif(r_votes > rest_votes && r_votes > g_votes && r_votes > p_votes && r_votes > s_votes && r_votes > e_votes && r_votes > f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 2;
elseif(p_votes > rest_votes && p_votes > g_votes && p_votes > r_votes && p_votes > s_votes && p_votes > e_votes && p_votes > f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 3;
elseif(s_votes > rest_votes && s_votes > g_votes && s_votes > r_votes && s_votes > p_votes && s_votes > e_votes && s_votes > f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 4;
elseif(e_votes > rest_votes && e_votes > g_votes && e_votes > r_votes && e_votes > p_votes && e_votes > s_votes && e_votes > f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 5;
elseif(f_votes > rest_votes && f_votes > g_votes && f_votes > r_votes && f_votes > p_votes && f_votes > s_votes && f_votes > e_votes)
    wrong_s = wrong_s + 1;
    last_mv = 6;
else
    wrong_r = wrong_r + 1;
end

results(5,5) = correct_s / (correct_s + wrong_s);

% Testing Extension
correct_e = 0;
wrong_e = 0;
last_mv = 0;
for a = 5:length(mv_e)
    rest_votes = 0;
    g_votes = 0;
    r_votes = 0;
    p_votes = 0;
    s_votes = 0;
    e_votes = 0;
    f_votes = 0;

    if(mv_e(a) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_e(a) == 1)
g_votes = g_votes + 1;
elseif(mv_e(a) == 2)
   r_votes = r_votes + 1;
elseif(mv_e(a) == 3)
   p_votes = p_votes + 1;
elseif(mv_e(a) == 4)
   s_votes = s_votes + 1;
elseif(mv_e(a) == 5)
   e_votes = e_votes + 1;
else
   f_votes = f_votes + 1;
end
if(mv_e(a-1) == 0)
   rest_votes = rest_votes + 1;
elseif(mv_e(a-1) == 1)
   g_votes = g_votes + 1;
elseif(mv_e(a-1) == 2)
   r_votes = r_votes + 1;
elseif(mv_e(a-1) == 3)
   p_votes = p_votes + 1;
elseif(mv_e(a-1) == 4)
   s_votes = s_votes + 1;
elseif(mv_e(a-1) == 5)
   e_votes = e_votes + 1;
else
   f_votes = f_votes + 1;
end
if(mv_e(a-2) == 0)
   rest_votes = rest_votes + 1;
elseif(mv_e(a-2) == 1)
   g_votes = g_votes + 1;
elseif(mv_e(a-2) == 2)
   r_votes = r_votes + 1;
elseif(mv_e(a-2) == 3)
   p_votes = p_votes + 1;
elseif(mv_e(a-2) == 4)
   s_votes = s_votes + 1;
elseif(mv_e(a-2) == 5)
   e_votes = e_votes + 1;
else
   f_votes = f_votes + 1;
end
if(mv_e(a-3) == 0)
   rest_votes = rest_votes + 1;
elseif(mv_e(a-3) == 1)
   g_votes = g_votes + 1;
elseif(mv_e(a-3) == 2)
   r_votes = r_votes + 1;
elseif(mv_e(a-3) == 3)
   p_votes = p_votes + 1;
elseif(mv_e(a-3) == 4)
   s_votes = s_votes + 1;
elseif(mv_e(a-3) == 5)
   e_votes = e_votes + 1;
else
   f_votes = f_votes + 1;
end
if(mv_e(a-4) == 0)
   rest_votes = rest_votes + 1;
elseif(mv_e(a-4) == 1)
   g_votes = g_votes + 1;
elseif(mv_e(a-4) == 2)
r_votes = r_votes + 1;
elseif (mv_e(a-4) == 3)
p_votes = p_votes + 1;
elseif (mv_e(a-4) == 4)
s_votes = s_votes + 1;
elseif (mv_e(a-4) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end
if (rest_votes > g_votes && rest_votes > r_votes && rest_votes > p_votes && rest_votes > s_votes && rest_votes > e_votes && rest_votes > f_votes)
wrong_e = wrong_e + 1;
last_mv = 0;
elseif (g_votes > rest_votes && g_votes > r_votes && g_votes > p_votes && g_votes > s_votes && g_votes > e_votes && g_votes > f_votes)
wrong_e = wrong_e + 1;
last_mv = 1;
elseif (r_votes > rest_votes && r_votes > g_votes && r_votes > p_votes && r_votes > s_votes && r_votes > e_votes && r_votes > f_votes)
wrong_e = wrong_e + 1;
last_mv = 2;
elseif (p_votes > rest_votes && p_votes > g_votes && p_votes > r_votes && p_votes > s_votes && p_votes > e_votes && p_votes > f_votes)
wrong_e = wrong_e + 1;
last_mv = 3;
elseif (s_votes > rest_votes && s_votes > g_votes && s_votes > r_votes && s_votes > p_votes && s_votes > e_votes && s_votes > f_votes)
wrong_e = wrong_e + 1;
last_mv = 4;
elseif (e_votes > rest_votes && e_votes > g_votes && e_votes > r_votes && e_votes > p_votes && e_votes > s_votes && e_votes > f_votes)
correct_e = correct_e + 1;
last_mv = 5;
elseif (f_votes > rest_votes && f_votes > g_votes && f_votes > r_votes && f_votes > p_votes && f_votes > s_votes && f_votes > e_votes)
wrong_e = wrong_e + 1;
last_mv = 6;
else
wrong_e = wrong_e + 1;
end
end
results(5,6) = correct_e / (correct_e + wrong_e);

% Testing Flexion

correct_f = 0;
wrong_f = 0;
last_mv = 0;
for a = 5:length(mv_f)

rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;
if (mv_f(a) == 0)
rest_votes = rest_votes + 1;

elseif(mv_f(a) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a) == 4)
    s_votes = s_votes + 1;
else(
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_f(a-1) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-1) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-1) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-1) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-1) == 4)
    s_votes = s_votes + 1;
else(mv_f(a-1) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_f(a-2) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-2) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-2) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-2) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-2) == 4)
    s_votes = s_votes + 1;
else(mv_f(a-2) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_f(a-3) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-3) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-3) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-3) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-3) == 4)
    s_votes = s_votes + 1;
else(mv_f(a-3) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_f(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-4) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-4) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-4) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-4) == 4)
    s_votes = s_votes + 1;
else(mv_f(a-4) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
g_votes = g_votes + 1;
elseif (mv_f(a-4) == 2)
  r_votes = r_votes + 1;
elseif (mv_f(a-4) == 3)
  p_votes = p_votes + 1;
elseif (mv_f(a-4) == 4)
  s_votes = s_votes + 1;
elseif (mv_f(a-4) == 5)
  e_votes = e_votes + 1;
else
  f_votes = f_votes + 1;
end

if (rest_votes > g_votes && rest_votes > r_votes && rest_votes > p_votes && rest_votes > s_votes && rest_votes > e_votes && rest_votes > f_votes)
  wrong_f = wrong_f + 1;
  last_mv = 0;
elseif (g_votes > rest_votes && g_votes > r_votes && g_votes > p_votes && g_votes > s_votes && g_votes > e_votes && g_votes > f_votes)
  wrong_f = wrong_f + 1;
  last_mv = 1;
elseif (r_votes > rest_votes && r_votes > g_votes && r_votes > p_votes && r_votes > s_votes && r_votes > e_votes && r_votes > f_votes)
  wrong_f = wrong_f + 1;
  last_mv = 2;
elseif (p_votes > rest_votes && p_votes > g_votes && p_votes > r_votes && p_votes > s_votes && p_votes > e_votes && p_votes > f_votes)
  wrong_f = wrong_f + 1;
  last_mv = 3;
elseif (s_votes > rest_votes && s_votes > g_votes && s_votes > r_votes && s_votes > p_votes && s_votes > e_votes && s_votes > f_votes)
  wrong_f = wrong_f + 1;
  last_mv = 4;
elseif (e_votes > rest_votes && e_votes > g_votes && e_votes > r_votes && e_votes > p_votes && e_votes > s_votes && e_votes > f_votes)
  wrong_f = wrong_f + 1;
  last_mv = 5;
elseif (f_votes > rest_votes && f_votes > g_votes && f_votes > r_votes && f_votes > p_votes && f_votes > s_votes && f_votes > e_votes)
  correct_f = correct_f + 1;
  last_mv = 6;
else
  correct_f = correct_f + 1;
  last_mv = 0;
end
results(5,7) = correct_f / (correct_f + wrong_f);
results(5,8) = (correct_g + correct_r + correct_p + correct_s + correct_rest) / (wrong_g + wrong_r + wrong_p + wrong_s + wrong_rest + correct_g + correct_r + correct_p + correct_s + correct_rest);
Appendix D: Matlab Code for Testing TD Features

% This m-file extracts the TD features from EMG data and runs the LDA on
% all of the features individually and as a set. It also performs a
% majority voting classification as well.

% Prior to using this m-file, the EMG data was segmented up into windows of
% the corresponding lengths below. Depending on which window was being
% tested, one of the mat files below would be uncommented to load the EMG
% data prior to performing any signal processing.
% load raw_data_50ms.mat
% load raw_data_100ms.mat
% load raw_data_250ms.mat

% Memory Allocation
rest_features_train = zeros(size(rest_train_1,1),30);
rest_features_test = zeros(size(rest_test_1,1),30);
grasp_features_train = zeros(size(g_train_1,1),30);
grasp_features_test = zeros(size(g_test_1,1),30);
releasing_features_train = zeros(size(r_train_1,1),30);
releasing_features_test = zeros(size(r_test_1,1),30);
pronation_features_train = zeros(size(p_train_1,1),30);
pronation_features_test = zeros(size(p_test_1,1),30);
supination_features_train = zeros(size(s_train_1,1),30);
supination_features_test = zeros(size(s_test_1,1),30);
extension_features_train = zeros(size(e_train_1,1),30);
extension_features_test = zeros(size(e_test_1,1),30);
flexion_features_train = zeros(size(f_train_1,1),30);
flexion_features_test = zeros(size(f_test_1,1),30);

% Mean Absolute Value
for a = 1:size(g_train_1,1)
grasp_features_train(a,1) = sum(abs(g_train_1(a,:)))/size(g_train_1,2);
end
for a = 1:size(g_train_2,1)
grasp_features_train(a,2) = sum(abs(g_train_2(a,:)))/size(g_train_2,2);
end
for a = 1:size(g_train_3,1)
grasp_features_train(a,3) = sum(abs(g_train_3(a,:)))/size(g_train_3,2);
end
for a = 1:size(g_train_4,1)
grasp_features_train(a,4) = sum(abs(g_train_4(a,:)))/size(g_train_4,2);
end
for a = 1:size(g_train_5,1)
grasp_features_train(a,5) = sum(abs(g_train_5(a,:)))/size(g_train_5,2);
end
for a = 1:size(g_train_6,1)
grasp_features_train(a,6) = sum(abs(g_train_6(a,:)))/size(g_train_6,2);
end
for a = 1:size(g_test_1,1)
grasp_features_test(a,1) = sum(abs(g_test_1(a,:)))/size(g_test_1,2);
end
for a = 1:size(g_test_2,1)
grasp_features_test(a,2) = sum(abs(g_test_2(a,:)))/size(g_test_2,2);
end
for a = 1:size(g_test_3,1)
grasp_features_test(a,3) = sum(abs(g_test_3(a,:)))/size(g_test_3,2);
end
for a = 1:size(g_test_4,1)
grasp_features_test(a,4) = sum(abs(g_test_4(a,:)))/size(g_test_4,2);
for a = 1:size(g_test_5,1)
    grasp_features_test(a,5) = sum(abs(g_test_5(a,:)))/size(g_test_5,2);
end
for a = 1:size(g_test_6,1)
    grasp_features_test(a,6) = sum(abs(g_test_6(a,:)))/size(g_test_6,2);
end
for a = 1:size(r_train_1,1)
    releasing_features_train(a,1) = sum(abs(r_train_1(a,:)))/size(r_train_1,2);
end
for a = 1:size(r_train_2,1)
    releasing_features_train(a,2) = sum(abs(r_train_2(a,:)))/size(r_train_2,2);
end
for a = 1:size(r_train_3,1)
    releasing_features_train(a,3) = sum(abs(r_train_3(a,:)))/size(r_train_3,2);
end
for a = 1:size(r_train_4,1)
    releasing_features_train(a,4) = sum(abs(r_train_4(a,:)))/size(r_train_4,2);
end
for a = 1:size(r_train_5,1)
    releasing_features_train(a,5) = sum(abs(r_train_5(a,:)))/size(r_train_5,2);
end
for a = 1:size(r_train_6,1)
    releasing_features_train(a,6) = sum(abs(r_train_6(a,:)))/size(r_train_6,2);
end
for a = 1:size(r_test_1,1)
    releasing_features_test(a,1) = sum(abs(r_test_1(a,:)))/size(r_test_1,2);
end
for a = 1:size(r_test_2,1)
    releasing_features_test(a,2) = sum(abs(r_test_2(a,:)))/size(r_test_2,2);
end
for a = 1:size(r_test_3,1)
    releasing_features_test(a,3) = sum(abs(r_test_3(a,:)))/size(r_test_3,2);
end
for a = 1:size(r_test_4,1)
    releasing_features_test(a,4) = sum(abs(r_test_4(a,:)))/size(r_test_4,2);
end
for a = 1:size(r_test_5,1)
    releasing_features_test(a,5) = sum(abs(r_test_5(a,:)))/size(r_test_5,2);
end
for a = 1:size(r_test_6,1)
    releasing_features_test(a,6) = sum(abs(r_test_6(a,:)))/size(r_test_6,2);
end
for a = 1:size(p_train_1,1)
    pronation_features_train(a,1) = sum(abs(p_train_1(a,:)))/size(p_train_1,2);
end
for a = 1:size(p_train_2,1)
    pronation_features_train(a,2) = sum(abs(p_train_2(a,:)))/size(p_train_2,2);
end
for a = 1:size(p_train_3,1)
    pronation_features_train(a,3) = sum(abs(p_train_3(a,:)))/size(p_train_3,2);
end
for a = 1:size(p_train_4,1)
    pronation_features_train(a,4) = sum(abs(p_train_4(a,:)))/size(p_train_4,2);
end
for a = 1:size(p_train_5,1)
    pronation_features_train(a,5) = sum(abs(p_train_5(a,:)))/size(p_train_5,2);
end
for a = 1:size(p_train_6,1)
    pronation_features_train(a,6) = sum(abs(p_train_6(a,:)))/size(p_train_6,2);
end
for a = 1:size(p_test_1,1)
    pronation_features_test(a,1) = sum(abs(p_test_1(a,:)))/size(p_test_1,2);
end
for a = 1:size(p_test_2,1)
    pronation_features_test(a,2) = sum(abs(p_test_2(a,:)))/size(p_test_2,2);
end
for a = 1:size(p_test_3,1)
pronation_features_test(a,1) = sum(abs(p_test_1(a,:)))/size(p_test_1,2);
end
for a = 1:size(p_test_2,1)
    pronation_features_test(a,2) = sum(abs(p_test_2(a,:)))/size(p_test_2,2);
end
for a = 1:size(p_test_3,1)
    pronation_features_test(a,3) = sum(abs(p_test_3(a,:)))/size(p_test_3,2);
end
for a = 1:size(p_test_4,1)
    pronation_features_test(a,4) = sum(abs(p_test_4(a,:)))/size(p_test_4,2);
end
for a = 1:size(p_test_5,1)
    pronation_features_test(a,5) = sum(abs(p_test_5(a,:)))/size(p_test_5,2);
end
for a = 1:size(p_test_6,1)
    pronation_features_test(a,6) = sum(abs(p_test_6(a,:)))/size(p_test_6,2);
end
for a = 1:size(s_train_1,1)
    supination_features_train(a,1) = sum(abs(s_train_1(a,:)))/size(s_train_1,2);
end
for a = 1:size(s_train_2,1)
    supination_features_train(a,2) = sum(abs(s_train_2(a,:)))/size(s_train_2,2);
end
for a = 1:size(s_train_3,1)
    supination_features_train(a,3) = sum(abs(s_train_3(a,:)))/size(s_train_3,2);
end
for a = 1:size(s_train_4,1)
    supination_features_train(a,4) = sum(abs(s_train_4(a,:)))/size(s_train_4,2);
end
for a = 1:size(s_train_5,1)
    supination_features_train(a,5) = sum(abs(s_train_5(a,:)))/size(s_train_5,2);
end
for a = 1:size(s_train_6,1)
    supination_features_train(a,6) = sum(abs(s_train_6(a,:)))/size(s_train_6,2);
end
for a = 1:size(s_test_1,1)
    supination_features_test(a,1) = sum(abs(s_test_1(a,:)))/size(s_test_1,2);
end
for a = 1:size(s_test_2,1)
    supination_features_test(a,2) = sum(abs(s_test_2(a,:)))/size(s_test_2,2);
end
for a = 1:size(s_test_3,1)
    supination_features_test(a,3) = sum(abs(s_test_3(a,:)))/size(s_test_3,2);
end
for a = 1:size(s_test_4,1)
    supination_features_test(a,4) = sum(abs(s_test_4(a,:)))/size(s_test_4,2);
end
for a = 1:size(s_test_5,1)
    supination_features_test(a,5) = sum(abs(s_test_5(a,:)))/size(s_test_5,2);
end
for a = 1:size(s_test_6,1)
    supination_features_test(a,6) = sum(abs(s_test_6(a,:)))/size(s_test_6,2);
end
for a = 1:size(rest_train_1,1)
    rest_features_train(a,1) = sum(abs(rest_train_1(a,:)))/size(rest_train_1,2);
end
for a = 1:size(rest_train_2,1)
    rest_features_train(a,2) = sum(abs(rest_train_2(a,:)))/size(rest_train_2,2);
end
for a = 1:size(rest_train_3,1)
    rest_features_train(a,3) = sum(abs(rest_train_3(a,:)))/size(rest_train_3,2);
end
for a = 1:size(rest_train_4,1)
    rest_features_train(a,4) = sum(abs(rest_train_4(a,:)))/size(rest_train_4,2);
end
for a = 1:size(rest_train_5,1)
    rest_features_train(a,5) = sum(abs(rest_train_5(a,:)))/size(rest_train_5,2);
end
for a = 1:size(rest_train_6,1)
    rest_features_train(a,6) = sum(abs(rest_train_6(a,:)))/size(rest_train_6,2);
end
for a = 1:size(rest_test_1,1)
    rest_features_test(a,1) = sum(abs(rest_test_1(a,:)))/size(rest_test_1,2);
end
for a = 1:size(rest_test_2,1)
    rest_features_test(a,2) = sum(abs(rest_test_2(a,:)))/size(rest_test_2,2);
end
for a = 1:size(rest_test_3,1)
    rest_features_test(a,3) = sum(abs(rest_test_3(a,:)))/size(rest_test_3,2);
end
for a = 1:size(rest_test_4,1)
    rest_features_test(a,4) = sum(abs(rest_test_4(a,:)))/size(rest_test_4,2);
end
for a = 1:size(rest_test_5,1)
    rest_features_test(a,5) = sum(abs(rest_test_5(a,:)))/size(rest_test_5,2);
end
for a = 1:size(rest_test_6,1)
    rest_features_test(a,6) = sum(abs(rest_test_6(a,:)))/size(rest_test_6,2);
end
for a = 1:size(e_train_1,1)
    extension_features_train(a,1) = sum(abs(e_train_1(a,:)))/size(e_train_1,2);
end
for a = 1:size(e_train_2,1)
    extension_features_train(a,2) = sum(abs(e_train_2(a,:)))/size(e_train_2,2);
end
for a = 1:size(e_train_3,1)
    extension_features_train(a,3) = sum(abs(e_train_3(a,:)))/size(e_train_3,2);
end
for a = 1:size(e_train_4,1)
    extension_features_train(a,4) = sum(abs(e_train_4(a,:)))/size(e_train_4,2);
end
for a = 1:size(e_train_5,1)
    extension_features_train(a,5) = sum(abs(e_train_5(a,:)))/size(e_train_5,2);
end
for a = 1:size(e_train_6,1)
    extension_features_train(a,6) = sum(abs(e_train_6(a,:)))/size(e_train_6,2);
end
for a = 1:size(e_test_1,1)
    extension_features_test(a,1) = sum(abs(e_test_1(a,:)))/size(e_test_1,2);
end
for a = 1:size(e_test_2,1)
    extension_features_test(a,2) = sum(abs(e_test_2(a,:)))/size(e_test_2,2);
end
for a = 1:size(e_test_3,1)
    extension_features_test(a,3) = sum(abs(e_test_3(a,:)))/size(e_test_3,2);
end
for a = 1:size(e_test_4,1)
    extension_features_test(a,4) = sum(abs(e_test_4(a,:)))/size(e_test_4,2);
end
for a = 1:size(e_test_5,1)
    extension_features_test(a,5) = sum(abs(e_test_5(a,:)))/size(e_test_5,2);
end
for a = 1:size(e_test_6,1)
    extension_features_test(a,6) = sum(abs(e_test_6(a,:)))/size(e_test_6,2);
for a = 1:size(f_train_1,1)
    flexion_features_train(a,1) = sum(abs(f_train_1(a,:)))/size(f_train_1,2);
end
for a = 1:size(f_train_2,1)
    flexion_features_train(a,2) = sum(abs(f_train_2(a,:)))/size(f_train_2,2);
end
for a = 1:size(f_train_3,1)
    flexion_features_train(a,3) = sum(abs(f_train_3(a,:)))/size(f_train_3,2);
end
for a = 1:size(f_train_4,1)
    flexion_features_train(a,4) = sum(abs(f_train_4(a,:)))/size(f_train_4,2);
end
for a = 1:size(f_train_5,1)
    flexion_features_train(a,5) = sum(abs(f_train_5(a,:)))/size(f_train_5,2);
end
for a = 1:size(f_train_6,1)
    flexion_features_train(a,6) = sum(abs(f_train_6(a,:)))/size(f_train_6,2);
end
for a = 1:size(f_test_1,1)
    flexion_features_test(a,1) = sum(abs(f_test_1(a,:)))/size(f_test_1,2);
end
for a = 1:size(f_test_2,1)
    flexion_features_test(a,2) = sum(abs(f_test_2(a,:)))/size(f_test_2,2);
end
for a = 1:size(f_test_3,1)
    flexion_features_test(a,3) = sum(abs(f_test_3(a,:)))/size(f_test_3,2);
end
for a = 1:size(f_test_4,1)
    flexion_features_test(a,4) = sum(abs(f_test_4(a,:)))/size(f_test_4,2);
end
for a = 1:size(f_test_5,1)
    flexion_features_test(a,5) = sum(abs(f_test_5(a,:)))/size(f_test_5,2);
end
for a = 1:size(f_test_6,1)
    flexion_features_test(a,6) = sum(abs(f_test_6(a,:)))/size(f_test_6,2);
end

% Mean Absolute Value Slope
for a = 1:size(g_train_1,1)-1
    grasp_features_train(a,7) = grasp_features_train(a+1,1) - grasp_features_train(a,1);
end
for a = 1:size(g_train_2,1)-1
    grasp_features_train(a,8) = grasp_features_train(a+1,2) - grasp_features_train(a,2);
end
for a = 1:size(g_train_3,1)-1
    grasp_features_train(a,9) = grasp_features_train(a+1,3) - grasp_features_train(a,3);
end
for a = 1:size(g_train_4,1)-1
    grasp_features_train(a,10) = grasp_features_train(a+1,4) - grasp_features_train(a,4);
end
for a = 1:size(g_train_5,1)-1
    grasp_features_train(a,11) = grasp_features_train(a+1,5) - grasp_features_train(a,5);
end
for a = 1:size(g_train_6,1)-1
    grasp_features_train(a,12) = grasp_features_train(a+1,6) - grasp_features_train(a,6);
end
for a = 1:size(g_test_1,1)-1
    grasp_features_test(a,7) = grasp_features_test(a+1,1) - grasp_features_test(a,1);
end
for a = 1:size(g_test_2,1)-1
    grasp_features_test(a,8) = grasp_features_test(a+1,2) - grasp_features_test(a,2);
end
for a = 1:size(g_test_3,1)-1
    grasp_features_test(a,9) = grasp_features_test(a+1,3) - grasp_features_test(a,3);
end
for a = 1:size(g_test_4,1)-1
    grasp_features_test(a,10) = grasp_features_test(a+1,4) - grasp_features_test(a,4);
end
for a = 1:size(g_test_5,1)-1
    grasp_features_test(a,11) = grasp_features_test(a+1,5) - grasp_features_test(a,5);
end
for a = 1:size(g_test_6,1)-1
    grasp_features_test(a,12) = grasp_features_test(a+1,6) - grasp_features_test(a,6);
end
for a = 1:size(r_train_1,1)-1
    releasing_features_train(a,7) = releasing_features_train(a+1,1) - releasing_features_train(a,1);
end
for a = 1:size(r_train_2,1)-1
    releasing_features_train(a,8) = releasing_features_train(a+1,2) - releasing_features_train(a,2);
end
for a = 1:size(r_train_3,1)-1
    releasing_features_train(a,9) = releasing_features_train(a+1,3) - releasing_features_train(a,3);
end
for a = 1:size(r_train_4,1)-1
    releasing_features_train(a,10) = releasing_features_train(a+1,4) - releasing_features_train(a,4);
end
for a = 1:size(r_train_5,1)-1
    releasing_features_train(a,11) = releasing_features_train(a+1,5) - releasing_features_train(a,5);
end
for a = 1:size(r_train_6,1)-1
    releasing_features_train(a,12) = releasing_features_train(a+1,6) - releasing_features_train(a,6);
end
for a = 1:size(r_test_1,1)-1
    releasing_features_test(a,7) = releasing_features_test(a+1,1) - releasing_features_test(a,1);
end
for a = 1:size(r_test_2,1)-1
    releasing_features_test(a,8) = releasing_features_test(a+1,2) - releasing_features_test(a,2);
end
for a = 1:size(r_test_3,1)-1
    releasing_features_test(a,9) = releasing_features_test(a+1,3) - releasing_features_test(a,3);
end
for a = 1:size(r_test_4,1)-1
    releasing_features_test(a,10) = releasing_features_test(a+1,4) - releasing_features_test(a,4);
end
for a = 1:size(r_test_5,1)-1
    releasing_features_test(a,11) = releasing_features_test(a+1,5) - releasing_features_test(a,5);
end
for a = 1:size(r_test_6,1)-1
    releasing_features_test(a,12) = releasing_features_test(a+1,6) - releasing_features_test(a,6);
end
for a = 1:size(p_train_1,1)-1
    pronation_features_train(a,7) = pronation_features_train(a+1,1) - pronation_features_train(a,1);
end
for a = 1:size(p_train_2,1)-1
    pronation_features_train(a,8) = pronation_features_train(a+1,2) - pronation_features_train(a,2);
end
for a = 1:size(p_train_3,1)-1
    pronation_features_train(a,9) = pronation_features_train(a+1,3) - pronation_features_train(a,3);
end
for a = 1:size(p_train_4,1)-1
    pronation_features_train(a,10) = pronation_features_train(a+1,4) - pronation_features_train(a,4);
end
for a = 1:size(p_train_5,1)-1
    pronation_features_train(a,11) = pronation_features_train(a+1,5) - pronation_features_train(a,5);
end
for a = 1:size(p_train_6,1)-1
    pronation_features_train(a,12) = pronation_features_train(a+1,6) - pronation_features_train(a,6);
end
end
pronation_features_train(a,11) = pronation_features_train(a+1,5) - pronation_features_train(a,5);
end
for a = 1:size(p_train_6,1)-1
  pronation_features_train(a,12) = pronation_features_train(a+1,6) - pronation_features_train(a,6);
end
for a = 1:size(p_test_1,1)-1
  pronation_features_test(a,7) = pronation_features_test(a+1,1) - pronation_features_test(a,1);
end
for a = 1:size(p_test_2,1)-1
  pronation_features_test(a,8) = pronation_features_test(a+1,2) - pronation_features_test(a,2);
end
for a = 1:size(p_test_3,1)-1
  pronation_features_test(a,9) = pronation_features_test(a+1,3) - pronation_features_test(a,3);
end
for a = 1:size(p_test_4,1)-1
  pronation_features_test(a,10) = pronation_features_test(a+1,4) - pronation_features_test(a,4);
end
for a = 1:size(p_test_5,1)-1
  pronation_features_test(a,11) = pronation_features_test(a+1,5) - pronation_features_test(a,5);
end
for a = 1:size(p_test_6,1)-1
  pronation_features_test(a,12) = pronation_features_test(a+1,6) - pronation_features_test(a,6);
end
for a = 1:size(s_train_1,1)-1
  supination_features_train(a,7) = supination_features_train(a+1,1) - supination_features_train(a,1);
end
for a = 1:size(s_train_2,1)-1
  supination_features_train(a,8) = supination_features_train(a+1,2) - supination_features_train(a,2);
end
for a = 1:size(s_train_3,1)-1
  supination_features_train(a,9) = supination_features_train(a+1,3) - supination_features_train(a,3);
end
for a = 1:size(s_train_4,1)-1
  supination_features_train(a,10) = supination_features_train(a+1,4) - supination_features_train(a,4);
end
for a = 1:size(s_train_5,1)-1
  supination_features_train(a,11) = supination_features_train(a+1,5) - supination_features_train(a,5);
end
for a = 1:size(s_train_6,1)-1
  supination_features_train(a,12) = supination_features_train(a+1,6) - supination_features_train(a,6);
end
for a = 1:size(s_test_1,1)-1
  supination_features_test(a,7) = supination_features_test(a+1,1) - supination_features_test(a,1);
end
for a = 1:size(s_test_2,1)-1
  supination_features_test(a,8) = supination_features_test(a+1,2) - supination_features_test(a,2);
end
for a = 1:size(s_test_3,1)-1
  supination_features_test(a,9) = supination_features_test(a+1,3) - supination_features_test(a,3);
end
for a = 1:size(s_test_4,1)-1
  supination_features_test(a,10) = supination_features_test(a+1,4) - supination_features_test(a,4);
end
for a = 1:size(s_test_5,1)-1
  supination_features_test(a,11) = supination_features_test(a+1,5) - supination_features_test(a,5);
end
for a = 1:size(s_test_6,1)-1
  supination_features_test(a,12) = supination_features_test(a+1,6) - supination_features_test(a,6);
end
for a = 1:size(e_train_1,1)-1
  extension_features_train(a,7) = extension_features_train(a+1,1) - extension_features_train(a,1);
end
for a = 1:size(e_train_2,1)-1
  extension_features_train(a,8) = extension_features_train(a+1,2) - extension_features_train(a,2);
end
for a = 1:size(e_train_3,1)-1
  extension_features_train(a,9) = extension_features_train(a+1,3) - extension_features_train(a,3);
end
for a = 1:size(e_train_4,1)-1
  extension_features_train(a,10) = extension_features_train(a+1,4) - extension_features_train(a,4);
end
for a = 1:size(e_train_5,1)-1
  extension_features_train(a,11) = extension_features_train(a+1,5) - extension_features_train(a,5);
end
for a = 1:size(e_train_6,1)-1
  extension_features_train(a,12) = extension_features_train(a+1,6) - extension_features_train(a,6);
end
for a = 1:size(e_test_1,1)-1
  extension_features_test(a,7) = extension_features_test(a+1,1) - extension_features_test(a,1);
end
for a = 1:size(e_test_2,1)-1
  extension_features_test(a,8) = extension_features_test(a+1,2) - extension_features_test(a,2);
end
for a = 1:size(e_test_3,1)-1
  extension_features_test(a,9) = extension_features_test(a+1,3) - extension_features_test(a,3);
end
for a = 1:size(e_test_4,1)-1
  extension_features_test(a,10) = extension_features_test(a+1,4) - extension_features_test(a,4);
end
for a = 1:size(e_test_5,1)-1
  extension_features_test(a,11) = extension_features_test(a+1,5) - extension_features_test(a,5);
end
for a = 1:size(e_test_6,1)-1
  extension_features_test(a,12) = extension_features_test(a+1,6) - extension_features_test(a,6);
end
for a = 1:size(f_train_1,1)-1
  flexion_features_train(a,7) = flexion_features_train(a+1,1) - flexion_features_train(a,1);
end
for a = 1:size(f_train_2,1)-1
  flexion_features_train(a,8) = flexion_features_train(a+1,2) - flexion_features_train(a,2);
end
for a = 1:size(f_train_3,1)-1
  flexion_features_train(a,9) = flexion_features_train(a+1,3) - flexion_features_train(a,3);
end
for a = 1:size(f_train_4,1)-1
  flexion_features_train(a,10) = flexion_features_train(a+1,4) - flexion_features_train(a,4);
end
for a = 1:size(f_train_5,1)-1
  flexion_features_train(a,11) = flexion_features_train(a+1,5) - flexion_features_train(a,5);
end
for a = 1:size(f_train_6,1)-1
  flexion_features_train(a,12) = flexion_features_train(a+1,6) - flexion_features_train(a,6);
end
for a = 1:size(f_test_1,1)-1
  flexion_features_test(a,7) = flexion_features_test(a+1,1) - flexion_features_test(a,1);
end
for a = 1:size(f_test_2,1)-1
  flexion_features_test(a,8) = flexion_features_test(a+1,2) - flexion_features_test(a,2);
end
for a = 1:size(f_test_3,1)-1
  flexion_features_test(a,9) = flexion_features_test(a+1,3) - flexion_features_test(a,3);
end
for a = 1:size(f_test_4,1)-1
  flexion_features_test(a,10) = flexion_features_test(a+1,4) - flexion_features_test(a,4);
end
for a = 1:size(f_test_5,1)-1
    flexion_features_test(a,11) = flexion_features_test(a+1,5) - flexion_features_test(a,5);
end
for a = 1:size(f_test_6,1)-1
    flexion_features_test(a,12) = flexion_features_test(a+1,6) - flexion_features_test(a,6);
end
for a = 1:size(rest_train_1,1)-1
    rest_features_train(a,7) = rest_features_train(a+1,1) - rest_features_train(a,1);
end
for a = 1:size(rest_train_2,1)-1
    rest_features_train(a,8) = rest_features_train(a+1,2) - rest_features_train(a,2);
end
for a = 1:size(rest_train_3,1)-1
    rest_features_train(a,9) = rest_features_train(a+1,3) - rest_features_train(a,3);
end
for a = 1:size(rest_train_4,1)-1
    rest_features_train(a,10) = rest_features_train(a+1,4) - rest_features_train(a,4);
end
for a = 1:size(rest_train_5,1)-1
    rest_features_train(a,11) = rest_features_train(a+1,5) - rest_features_train(a,5);
end
for a = 1:size(rest_train_6,1)-1
    rest_features_train(a,12) = rest_features_train(a+1,6) - rest_features_train(a,6);
end
for a = 1:size(rest_test_1,1)-1
    rest_features_test(a,7) = rest_features_test(a+1,1) - rest_features_test(a,1);
end
for a = 1:size(rest_test_2,1)-1
    rest_features_test(a,8) = rest_features_test(a+1,2) - rest_features_test(a,2);
end
for a = 1:size(rest_test_3,1)-1
    rest_features_test(a,9) = rest_features_test(a+1,3) - rest_features_test(a,3);
end
for a = 1:size(rest_test_4,1)-1
    rest_features_test(a,10) = rest_features_test(a+1,4) - rest_features_test(a,4);
end
for a = 1:size(rest_test_5,1)-1
    rest_features_test(a,11) = rest_features_test(a+1,5) - rest_features_test(a,5);
end
for a = 1:size(rest_test_6,1)-1
    rest_features_test(a,12) = rest_features_test(a+1,6) - rest_features_test(a,6);
end

% Zero Crossings
% Using a threshold of 10uV which accounts for noise in the system.
for a = 1:size(g_train_1,1)
    for b = 1:size(g_train_1,2)-1
        if(g_train_1(a,b)>0 && g_train_1(a,b+1)<0 && abs(g_train_1(a,b) - g_train_1(a,b+1))>=10)
            grasp_features_train(a,13) = grasp_features_train(a,13) + 1;
        elseif(g_train_1(a,b)<0 && g_train_1(a,b+1)>0 && abs(g_train_1(a,b) - g_train_1(a,b+1))>=10)
            grasp_features_train(a,13) = grasp_features_train(a,13) + 1;
        end
    end
end
for a = 1:size(g_train_2,1)
    for b = 1:size(g_train_2,2)-1
        if(g_train_2(a,b)>0 && g_train_2(a,b+1)<0 && abs(g_train_2(a,b) - g_train_2(a,b+1))>=10)
            grasp_features_train(a,14) = grasp_features_train(a,14) + 1;
        elseif(g_train_2(a,b)<0 && g_train_2(a,b+1)>0 && abs(g_train_2(a,b) - g_train_2(a,b+1))>=10)
            grasp_features_train(a,14) = grasp_features_train(a,14) + 1;
        end
    end
end
end
end
for a = 1:size(g_train_3,1)
  for b = 1:size(g_train_3,2)-1
    if g_train_3(a,b)>0 && g_train_3(a,b+1)<0 && abs(g_train_3(a,b) - g_train_3(a,b+1))>=10)
      grasp_features_train(a,15) = grasp_features_train(a,15) + 1;
    elseif g_train_3(a,b)<0 && g_train_3(a,b+1)>0 && abs(g_train_3(a,b) - g_train_3(a,b+1))>=10)
      grasp_features_train(a,15) = grasp_features_train(a,15) + 1;
    end
  end
end
for a = 1:size(g_train_4,1)
  for b = 1:size(g_train_4,2)-1
    if g_train_4(a,b)>0 && g_train_4(a,b+1)<0 && abs(g_train_4(a,b) - g_train_4(a,b+1))>=10)
      grasp_features_train(a,16) = grasp_features_train(a,16) + 1;
    elseif g_train_4(a,b)<0 && g_train_4(a,b+1)>0 && abs(g_train_4(a,b) - g_train_4(a,b+1))>=10)
      grasp_features_train(a,16) = grasp_features_train(a,16) + 1;
    end
  end
end
for a = 1:size(g_train_5,1)
  for b = 1:size(g_train_5,2)-1
    if g_train_5(a,b)>0 && g_train_5(a,b+1)<0 && abs(g_train_5(a,b) - g_train_5(a,b+1))>=10)
      grasp_features_train(a,17) = grasp_features_train(a,17) + 1;
    elseif g_train_5(a,b)<0 && g_train_5(a,b+1)>0 && abs(g_train_5(a,b) - g_train_5(a,b+1))>=10)
      grasp_features_train(a,17) = grasp_features_train(a,17) + 1;
    end
  end
end
for a = 1:size(g_train_6,1)
  for b = 1:size(g_train_6,2)-1
    if g_train_6(a,b)>0 && g_train_6(a,b+1)<0 && abs(g_train_6(a,b) - g_train_6(a,b+1))>=10)
      grasp_features_train(a,18) = grasp_features_train(a,18) + 1;
    elseif g_train_6(a,b)<0 && g_train_6(a,b+1)>0 && abs(g_train_6(a,b) - g_train_6(a,b+1))>=10)
      grasp_features_train(a,18) = grasp_features_train(a,18) + 1;
    end
  end
end
for a = 1:size(g_test_1,1)
  for b = 1:size(g_test_1,2)-1
    if g_test_1(a,b)>0 && g_test_1(a,b+1)<0 && abs(g_test_1(a,b) - g_test_1(a,b+1))>=10)
      grasp_features_test(a,13) = grasp_features_test(a,13) + 1;
    elseif g_test_1(a,b)<0 && g_test_1(a,b+1)>0 && abs(g_test_1(a,b) - g_test_1(a,b+1))>=10)
      grasp_features_test(a,13) = grasp_features_test(a,13) + 1;
    end
  end
end
for a = 1:size(g_test_2,1)
  for b = 1:size(g_test_2,2)-1
    if g_test_2(a,b)>0 && g_test_2(a,b+1)<0 && abs(g_test_2(a,b) - g_test_2(a,b+1))>=10)
      grasp_features_test(a,14) = grasp_features_test(a,14) + 1;
    elseif g_test_2(a,b)<0 && g_test_2(a,b+1)>0 && abs(g_test_2(a,b) - g_test_2(a,b+1))>=10)
      grasp_features_test(a,14) = grasp_features_test(a,14) + 1;
    end
  end
end
for a = 1:size(g_test_3,1)
  for b = 1:size(g_test_3,2)-1
    if g_test_3(a,b)>0 && g_test_3(a,b+1)<0 && abs(g_test_3(a,b) - g_test_3(a,b+1))>=10)
      grasp_features_test(a,15) = grasp_features_test(a,15) + 1;
    elseif g_test_3(a,b)<0 && g_test_3(a,b+1)>0 && abs(g_test_3(a,b) - g_test_3(a,b+1))>=10)
      grasp_features_test(a,15) = grasp_features_test(a,15) + 1;
  end
end
for a = 1:size(g_test_4,1)
    for b = 1:size(g_test_4,2)-1
        if (g_test_4(a,b)>0 & & g_test_4(a,b+1)<0 & & abs(g_test_4(a,b) - g_test_4(a,b+1))>=10)
            grasp_features_test(a,16) = grasp_features_test(a,16) + 1;
        elseif (g_test_4(a,b)<0 & & g_test_4(a,b+1)>0 & & abs(g_test_4(a,b) - g_test_4(a,b+1))>=10)
            grasp_features_test(a,16) = grasp_features_test(a,16) + 1;
        end
    end
end
end

for a = 1:size(g_test_5,1)
    for b = 1:size(g_test_5,2)-1
        if (g_test_5(a,b)>0 & & g_test_5(a,b+1)<0 & & abs(g_test_5(a,b) - g_test_5(a,b+1))>=10)
            grasp_features_test(a,17) = grasp_features_test(a,17) + 1;
        elseif (g_test_5(a,b)<0 & & g_test_5(a,b+1)>0 & & abs(g_test_5(a,b) - g_test_5(a,b+1))>=10)
            grasp_features_test(a,17) = grasp_features_test(a,17) + 1;
        end
    end
end
end

for a = 1:size(g_test_6,1)
    for b = 1:size(g_test_6,2)-1
        if (g_test_6(a,b)>0 & & g_test_6(a,b+1)<0 & & abs(g_test_6(a,b) - g_test_6(a,b+1))>=10)
            grasp_features_test(a,18) = grasp_features_test(a,18) + 1;
        elseif (g_test_6(a,b)<0 & & g_test_6(a,b+1)>0 & & abs(g_test_6(a,b) - g_test_6(a,b+1))>=10)
            grasp_features_test(a,18) = grasp_features_test(a,18) + 1;
        end
    end
end
end

for a = 1:size(r_train_1,1)
    for b = 1:size(r_train_1,2)-1
        if (r_train_1(a,b)>0 & & r_train_1(a,b+1)<0 & & abs(r_train_1(a,b) - r_train_1(a,b+1))>=10)
            releasing_features_train(a,13) = releasing_features_train(a,13) + 1;
        elseif (r_train_1(a,b)<0 & & r_train_1(a,b+1)>0 & & abs(r_train_1(a,b) - r_train_1(a,b+1))>=10)
            releasing_features_train(a,13) = releasing_features_train(a,13) + 1;
        end
    end
end
end

for a = 1:size(r_train_2,1)
    for b = 1:size(r_train_2,2)-1
        if (r_train_2(a,b)>0 & & r_train_2(a,b+1)<0 & & abs(r_train_2(a,b) - r_train_2(a,b+1))>=10)
            releasing_features_train(a,14) = releasing_features_train(a,14) + 1;
        elseif (r_train_2(a,b)<0 & & r_train_2(a,b+1)>0 & & abs(r_train_2(a,b) - r_train_2(a,b+1))>=10)
            releasing_features_train(a,14) = releasing_features_train(a,14) + 1;
        end
    end
end
end

for a = 1:size(r_train_3,1)
    for b = 1:size(r_train_3,2)-1
        if (r_train_3(a,b)>0 & & r_train_3(a,b+1)<0 & & abs(r_train_3(a,b) - r_train_3(a,b+1))>=10)
            releasing_features_train(a,15) = releasing_features_train(a,15) + 1;
        elseif (r_train_3(a,b)<0 & & r_train_3(a,b+1)>0 & & abs(r_train_3(a,b) - r_train_3(a,b+1))>=10)
            releasing_features_train(a,15) = releasing_features_train(a,15) + 1;
        end
    end
end
end

for a = 1:size(r_train_4,1)
    for b = 1:size(r_train_4,2)-1
        if (r_train_4(a,b)>0 & & r_train_4(a,b+1)<0 & & abs(r_train_4(a,b) - r_train_4(a,b+1))>=10)
            releasing_features_train(a,16) = releasing_features_train(a,16) + 1;
        elseif (r_train_4(a,b)<0 & & r_train_4(a,b+1)>0 & & abs(r_train_4(a,b) - r_train_4(a,b+1))>=10)
            releasing_features_train(a,16) = releasing_features_train(a,16) + 1;
        end
    end
end
end
releasing_features_train(a,16) = releasing_features_train(a,16) + 1;
end
end
end
for a = 1:size(r_train_5,1)
  for b = 1:size(r_train_5,2)-1
    if ( r_train_5(a,b)>0 & r_train_5(a,b+1)<0 & abs(r_train_5(a,b) - r_train_5(a,b+1))>=10)
      releasing_features_train(a,17) = releasing_features_train(a,17) + 1;
    elseif ( r_train_5(a,b)<0 & r_train_5(a,b+1)>0 & abs(r_train_5(a,b) - r_train_5(a,b+1))>=10)
      releasing_features_train(a,17) = releasing_features_train(a,17) + 1;
    end
  end
end
for a = 1:size(r_train_6,1)
  for b = 1:size(r_train_6,2)-1
    if ( r_train_6(a,b)>0 & r_train_6(a,b+1)<0 & abs(r_train_6(a,b) - r_train_6(a,b+1))>=10)
      releasing_features_train(a,18) = releasing_features_train(a,18) + 1;
    elseif ( r_train_6(a,b)<0 & r_train_6(a,b+1)>0 & abs(r_train_6(a,b) - r_train_6(a,b+1))>=10)
      releasing_features_train(a,18) = releasing_features_train(a,18) + 1;
    end
  end
end
for a = 1:size(r_test_1,1)
  for b = 1:size(r_test_1,2)-1
    if ( r_test_1(a,b)>0 & r_test_1(a,b+1)<0 & abs(r_test_1(a,b) - r_test_1(a,b+1))>=10)
      releasing_features_test(a,13) = releasing_features_test(a,13) + 1;
    elseif ( r_test_1(a,b)<0 & r_test_1(a,b+1)>0 & abs(r_test_1(a,b) - r_test_1(a,b+1))>=10)
      releasing_features_test(a,13) = releasing_features_test(a,13) + 1;
    end
  end
end
for a = 1:size(r_test_2,1)
  for b = 1:size(r_test_2,2)-1
    if ( r_test_2(a,b)>0 & r_test_2(a,b+1)<0 & abs(r_test_2(a,b) - r_test_2(a,b+1))>=10)
      releasing_features_test(a,14) = releasing_features_test(a,14) + 1;
    elseif ( r_test_2(a,b)<0 & r_test_2(a,b+1)>0 & abs(r_test_2(a,b) - r_test_2(a,b+1))>=10)
      releasing_features_test(a,14) = releasing_features_test(a,14) + 1;
    end
  end
end
for a = 1:size(r_test_3,1)
  for b = 1:size(r_test_3,2)-1
    if ( r_test_3(a,b)>0 & r_test_3(a,b+1)<0 & abs(r_test_3(a,b) - r_test_3(a,b+1))>=10)
      releasing_features_test(a,15) = releasing_features_test(a,15) + 1;
    elseif ( r_test_3(a,b)<0 & r_test_3(a,b+1)>0 & abs(r_test_3(a,b) - r_test_3(a,b+1))>=10)
      releasing_features_test(a,15) = releasing_features_test(a,15) + 1;
    end
  end
end
for a = 1:size(r_test_4,1)
  for b = 1:size(r_test_4,2)-1
    if ( r_test_4(a,b)>0 & r_test_4(a,b+1)<0 & abs(r_test_4(a,b) - r_test_4(a,b+1))>=10)
      releasing_features_test(a,16) = releasing_features_test(a,16) + 1;
    elseif ( r_test_4(a,b)<0 & r_test_4(a,b+1)>0 & abs(r_test_4(a,b) - r_test_4(a,b+1))>=10)
      releasing_features_test(a,16) = releasing_features_test(a,16) + 1;
    end
  end
end
for a = 1:size(r_test_5,1)
  for b = 1:size(r_test_5,2)-1
    if ( r_test_5(a,b)>0 & r_test_5(a,b+1)<0 & abs(r_test_5(a,b) - r_test_5(a,b+1))>=10)
      releasing_features_test(a,17) = releasing_features_test(a,17) + 1;
    end
  end
end
elseif (r_test_5(a,b)<0 & & r_test_5(a,b+1)>0 & & abs(r_test_5(a,b) - r_test_5(a,b+1))>=10)
    releasing_features_test(a,17) = releasing_features_test(a,17) + 1;
end
end
end

for a = 1:size(r_test_6,1)
    for b = 1:size(r_test_6,2)-1
        if (r_test_6(a,b)>0 & & r_test_6(a,b+1)<0 & & abs(r_test_6(a,b) - r_test_6(a,b+1))>=10)
            releasing_features_test(a,18) = releasing_features_test(a,18) + 1;
        elseif (r_test_6(a,b)<0 & & r_test_6(a,b+1)>0 & & abs(r_test_6(a,b) - r_test_6(a,b+1))>=10)
            releasing_features_test(a,18) = releasing_features_test(a,18) + 1;
        end
    end
end

for a = 1:size(p_train_1,1)
    for b = 1:size(p_train_1,2)-1
        if (p_train_1(a,b)>0 & & p_train_1(a,b+1)<0 & & abs(p_train_1(a,b) - p_train_1(a,b+1))>=10)
            pronation_features_train(a,13) = pronation_features_train(a,13) + 1;
        elseif (p_train_1(a,b)<0 & & p_train_1(a,b+1)>0 & & abs(p_train_1(a,b) - p_train_1(a,b+1))>=10)
            pronation_features_train(a,13) = pronation_features_train(a,13) + 1;
        end
    end
end

for a = 1:size(p_train_2,1)
    for b = 1:size(p_train_2,2)-1
        if (p_train_2(a,b)>0 & & p_train_2(a,b+1)<0 & & abs(p_train_2(a,b) - p_train_2(a,b+1))>=10)
            pronation_features_train(a,14) = pronation_features_train(a,14) + 1;
        elseif (p_train_2(a,b)<0 & & p_train_2(a,b+1)>0 & & abs(p_train_2(a,b) - p_train_2(a,b+1))>=10)
            pronation_features_train(a,14) = pronation_features_train(a,14) + 1;
        end
    end
end

for a = 1:size(p_train_3,1)
    for b = 1:size(p_train_3,2)-1
        if (p_train_3(a,b)>0 & & p_train_3(a,b+1)<0 & & abs(p_train_3(a,b) - p_train_3(a,b+1))>=10)
            pronation_features_train(a,15) = pronation_features_train(a,15) + 1;
        elseif (p_train_3(a,b)<0 & & p_train_3(a,b+1)>0 & & abs(p_train_3(a,b) - p_train_3(a,b+1))>=10)
            pronation_features_train(a,15) = pronation_features_train(a,15) + 1;
        end
    end
end

for a = 1:size(p_train_4,1)
    for b = 1:size(p_train_4,2)-1
        if (p_train_4(a,b)>0 & & p_train_4(a,b+1)<0 & & abs(p_train_4(a,b) - p_train_4(a,b+1))>=10)
            pronation_features_train(a,16) = pronation_features_train(a,16) + 1;
        elseif (p_train_4(a,b)<0 & & p_train_4(a,b+1)>0 & & abs(p_train_4(a,b) - p_train_4(a,b+1))>=10)
            pronation_features_train(a,16) = pronation_features_train(a,16) + 1;
        end
    end
end

for a = 1:size(p_train_5,1)
    for b = 1:size(p_train_5,2)-1
        if (p_train_5(a,b)>0 & & p_train_5(a,b+1)<0 & & abs(p_train_5(a,b) - p_train_5(a,b+1))>=10)
            pronation_features_train(a,17) = pronation_features_train(a,17) + 1;
        elseif (p_train_5(a,b)<0 & & p_train_5(a,b+1)>0 & & abs(p_train_5(a,b) - p_train_5(a,b+1))>=10)
            pronation_features_train(a,17) = pronation_features_train(a,17) + 1;
        end
    end
end

for a = 1:size(p_train_6,1)
    for b = 1:size(p_train_6,2)-1
        if (p_train_6(a,b)>0 & & p_train_6(a,b+1)<0 & & abs(p_train_6(a,b) - p_train_6(a,b+1))>=10)
pronation_features_train(a,18) = pronation_features_train(a,18) + 1;
elseif ( p_train_6(a,b)<0  &&  p_train_6(a,b+1)>0  &&  abs(p_train_6(a,b) - p_train_6(a,b+1))>=10)
    pronation_features_train(a,18) = pronation_features_train(a,18) + 1;
end
end

for a = 1:size(p_test_1,1)
    for b = 1:size(p_test_1,2)-1
        if ( p_test_1(a,b)>0  &&  p_test_1(a,b+1)<0  &&  abs(p_test_1(a,b) - p_test_1(a,b+1))>=10)
            pronation_features_test(a,13) = pronation_features_test(a,13) + 1;
        elseif ( p_test_1(a,b)<0  &&  p_test_1(a,b+1)>0  &&  abs(p_test_1(a,b) - p_test_1(a,b+1))>=10)
            pronation_features_test(a,13) = pronation_features_test(a,13) + 1;
        end
    end
end

for a = 1:size(p_test_2,1)
    for b = 1:size(p_test_2,2)-1
        if ( p_test_2(a,b)>0  &&  p_test_2(a,b+1)<0  &&  abs(p_test_2(a,b) - p_test_2(a,b+1))>=10)
            pronation_features_test(a,14) = pronation_features_test(a,14) + 1;
        elseif ( p_test_2(a,b)<0  &&  p_test_2(a,b+1)>0  &&  abs(p_test_2(a,b) - p_test_2(a,b+1))>=10)
            pronation_features_test(a,14) = pronation_features_test(a,14) + 1;
        end
    end
end

for a = 1:size(p_test_3,1)
    for b = 1:size(p_test_3,2)-1
        if ( p_test_3(a,b)>0  &&  p_test_3(a,b+1)<0  &&  abs(p_test_3(a,b) - p_test_3(a,b+1))>=10)
            pronation_features_test(a,15) = pronation_features_test(a,15) + 1;
        elseif ( p_test_3(a,b)<0  &&  p_test_3(a,b+1)>0  &&  abs(p_test_3(a,b) - p_test_3(a,b+1))>=10)
            pronation_features_test(a,15) = pronation_features_test(a,15) + 1;
        end
    end
end

for a = 1:size(p_test_4,1)
    for b = 1:size(p_test_4,2)-1
        if ( p_test_4(a,b)>0  &&  p_test_4(a,b+1)<0  &&  abs(p_test_4(a,b) - p_test_4(a,b+1))>=10)
            pronation_features_test(a,16) = pronation_features_test(a,16) + 1;
        elseif ( p_test_4(a,b)<0  &&  p_test_4(a,b+1)>0  &&  abs(p_test_4(a,b) - p_test_4(a,b+1))>=10)
            pronation_features_test(a,16) = pronation_features_test(a,16) + 1;
        end
    end
end

for a = 1:size(p_test_5,1)
    for b = 1:size(p_test_5,2)-1
        if ( p_test_5(a,b)>0  &&  p_test_5(a,b+1)<0  &&  abs(p_test_5(a,b) - p_test_5(a,b+1))>=10)
            pronation_features_test(a,17) = pronation_features_test(a,17) + 1;
        elseif ( p_test_5(a,b)<0  &&  p_test_5(a,b+1)>0  &&  abs(p_test_5(a,b) - p_test_5(a,b+1))>=10)
            pronation_features_test(a,17) = pronation_features_test(a,17) + 1;
        end
    end
end

for a = 1:size(p_test_6,1)
    for b = 1:size(p_test_6,2)-1
        if ( p_test_6(a,b)>0  &&  p_test_6(a,b+1)<0  &&  abs(p_test_6(a,b) - p_test_6(a,b+1))>=10)
            pronation_features_test(a,18) = pronation_features_test(a,18) + 1;
        elseif ( p_test_6(a,b)<0  &&  p_test_6(a,b+1)>0  &&  abs(p_test_6(a,b) - p_test_6(a,b+1))>=10)
            pronation_features_test(a,18) = pronation_features_test(a,18) + 1;
        end
    end
end

for a = 1:size(s_train_1,1)
    for b = 1:size(s_train_1,2)-1

199
if $s_{\text{train}_1}(a,b)>0$ && $s_{\text{train}_1}(a,b+1)<0$ && abs($s_{\text{train}_1}(a,b) - s_{\text{train}_1}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,13) = \text{supination_features}_{\text{train}}(a,13) + 1$

elseif $s_{\text{train}_1}(a,b)<0$ && $s_{\text{train}_1}(a,b+1)>0$ && abs($s_{\text{train}_1}(a,b) - s_{\text{train}_1}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,13) = \text{supination_features}_{\text{train}}(a,13) + 1$

end

dend

dend

dend

dend

def a = 1:size(s_{\text{train}_2},1)

def b = 1:size(s_{\text{train}_2},2)-1

if $s_{\text{train}_2}(a,b)>0$ && $s_{\text{train}_2}(a,b+1)<0$ && abs($s_{\text{train}_2}(a,b) - s_{\text{train}_2}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,14) = \text{supination_features}_{\text{train}}(a,14) + 1$

elseif $s_{\text{train}_2}(a,b)<0$ && $s_{\text{train}_2}(a,b+1)>0$ && abs($s_{\text{train}_2}(a,b) - s_{\text{train}_2}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,14) = \text{supination_features}_{\text{train}}(a,14) + 1$

end

dend

dend

dend

dend

def a = 1:size(s_{\text{train}_3},1)

def b = 1:size(s_{\text{train}_3},2)-1

if $s_{\text{train}_3}(a,b)>0$ && $s_{\text{train}_3}(a,b+1)<0$ && abs($s_{\text{train}_3}(a,b) - s_{\text{train}_3}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,15) = \text{supination_features}_{\text{train}}(a,15) + 1$

elseif $s_{\text{train}_3}(a,b)<0$ && $s_{\text{train}_3}(a,b+1)>0$ && abs($s_{\text{train}_3}(a,b) - s_{\text{train}_3}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,15) = \text{supination_features}_{\text{train}}(a,15) + 1$

end

dend

dend

dend

dend

def a = 1:size(s_{\text{train}_4},1)

def b = 1:size(s_{\text{train}_4},2)-1

if $s_{\text{train}_4}(a,b)>0$ && $s_{\text{train}_4}(a,b+1)<0$ && abs($s_{\text{train}_4}(a,b) - s_{\text{train}_4}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,16) = \text{supination_features}_{\text{train}}(a,16) + 1$

elseif $s_{\text{train}_4}(a,b)<0$ && $s_{\text{train}_4}(a,b+1)>0$ && abs($s_{\text{train}_4}(a,b) - s_{\text{train}_4}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,16) = \text{supination_features}_{\text{train}}(a,16) + 1$

end

dend

dend

dend

dend

def a = 1:size(s_{\text{train}_5},1)

def b = 1:size(s_{\text{train}_5},2)-1

if $s_{\text{train}_5}(a,b)>0$ && $s_{\text{train}_5}(a,b+1)<0$ && abs($s_{\text{train}_5}(a,b) - s_{\text{train}_5}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,17) = \text{supination_features}_{\text{train}}(a,17) + 1$

elseif $s_{\text{train}_5}(a,b)<0$ && $s_{\text{train}_5}(a,b+1)>0$ && abs($s_{\text{train}_5}(a,b) - s_{\text{train}_5}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,17) = \text{supination_features}_{\text{train}}(a,17) + 1$

end

dend

dend

dend

dend

def a = 1:size(s_{\text{train}_6},1)

def b = 1:size(s_{\text{train}_6},2)-1

if $s_{\text{train}_6}(a,b)>0$ && $s_{\text{train}_6}(a,b+1)<0$ && abs($s_{\text{train}_6}(a,b) - s_{\text{train}_6}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,18) = \text{supination_features}_{\text{train}}(a,18) + 1$

elseif $s_{\text{train}_6}(a,b)<0$ && $s_{\text{train}_6}(a,b+1)>0$ && abs($s_{\text{train}_6}(a,b) - s_{\text{train}_6}(a,b+1))>=10$

$\text{supination_features}_{\text{train}}(a,18) = \text{supination_features}_{\text{train}}(a,18) + 1$

end

dend

dend

dend

dend

def a = 1:size(s_{\text{test}_1},1)

def b = 1:size(s_{\text{test}_1},2)-1

if $s_{\text{test}_1}(a,b)>0$ && $s_{\text{test}_1}(a,b+1)<0$ && abs($s_{\text{test}_1}(a,b) - s_{\text{test}_1}(a,b+1))>=10$

$\text{supination_features}_{\text{test}}(a,13) = \text{supination_features}_{\text{test}}(a,13) + 1$

elseif $s_{\text{test}_1}(a,b)<0$ && $s_{\text{test}_1}(a,b+1)>0$ && abs($s_{\text{test}_1}(a,b) - s_{\text{test}_1}(a,b+1))>=10$

$\text{supination_features}_{\text{test}}(a,13) = \text{supination_features}_{\text{test}}(a,13) + 1$

end

dend

dend

de for a = 1:size(s_{\text{test}_2},1)
for b = 1:size(s_test_2,2)-1
    if s_test_2(a,b)>0 && s_test_2(a,b+1)<0 && abs(s_test_2(a,b) - s_test_2(a,b+1))>=10
        supination_features_test(a,14) = supination_features_test(a,14) + 1;
    elseif s_test_2(a,b)<0 && s_test_2(a,b+1)>0 && abs(s_test_2(a,b) - s_test_2(a,b+1))>=10
        supination_features_test(a,14) = supination_features_test(a,14) + 1;
    end
end
for a = 1:size(s_test_3,1)
    for b = 1:size(s_test_3,2)-1
        if s_test_3(a,b)>0 && s_test_3(a,b+1)<0 && abs(s_test_3(a,b) - s_test_3(a,b+1))>=10
            supination_features_test(a,15) = supination_features_test(a,15) + 1;
        elseif s_test_3(a,b)<0 && s_test_3(a,b+1)>0 && abs(s_test_3(a,b) - s_test_3(a,b+1))>=10
            supination_features_test(a,15) = supination_features_test(a,15) + 1;
        end
    end
end
for a = 1:size(s_test_4,1)
    for b = 1:size(s_test_4,2)-1
        if s_test_4(a,b)>0 && s_test_4(a,b+1)<0 && abs(s_test_4(a,b) - s_test_4(a,b+1))>=10
            supination_features_test(a,16) = supination_features_test(a,16) + 1;
        elseif s_test_4(a,b)<0 && s_test_4(a,b+1)>0 && abs(s_test_4(a,b) - s_test_4(a,b+1))>=10
            supination_features_test(a,16) = supination_features_test(a,16) + 1;
        end
    end
end
for a = 1:size(s_test_5,1)
    for b = 1:size(s_test_5,2)-1
        if s_test_5(a,b)>0 && s_test_5(a,b+1)<0 && abs(s_test_5(a,b) - s_test_5(a,b+1))>=10
            supination_features_test(a,17) = supination_features_test(a,17) + 1;
        elseif s_test_5(a,b)<0 && s_test_5(a,b+1)>0 && abs(s_test_5(a,b) - s_test_5(a,b+1))>=10
            supination_features_test(a,17) = supination_features_test(a,17) + 1;
        end
    end
end
for a = 1:size(s_test_6,1)
    for b = 1:size(s_test_6,2)-1
        if s_test_6(a,b)>0 && s_test_6(a,b+1)<0 && abs(s_test_6(a,b) - s_test_6(a,b+1))>=10
            supination_features_test(a,18) = supination_features_test(a,18) + 1;
        elseif s_test_6(a,b)<0 && s_test_6(a,b+1)>0 && abs(s_test_6(a,b) - s_test_6(a,b+1))>=10
            supination_features_test(a,18) = supination_features_test(a,18) + 1;
        end
    end
end
for a = 1:size(rest_train_1,1)
    for b = 1:size(rest_train_1,2)-1
        if rest_train_1(a,b)>0 && rest_train_1(a,b+1)<0 && abs(rest_train_1(a,b) - rest_train_1(a,b+1))>=10
            rest_features_train(a,13) = rest_features_train(a,13) + 1;
        elseif rest_train_1(a,b)<0 && rest_train_1(a,b+1)>0 && abs(rest_train_1(a,b) - rest_train_1(a,b+1))>=10
            rest_features_train(a,13) = rest_features_train(a,13) + 1;
        end
    end
end
for a = 1:size(rest_train_2,1)
    for b = 1:size(rest_train_2,2)-1
        if rest_train_2(a,b)>0 && rest_train_2(a,b+1)<0 && abs(rest_train_2(a,b) - rest_train_2(a,b+1))>=10
            rest_features_train(a,14) = rest_features_train(a,14) + 1;
        elseif rest_train_2(a,b)<0 && rest_train_2(a,b+1)>0 && abs(rest_train_2(a,b) - rest_train_2(a,b+1))>=10
            rest_features_train(a,14) = rest_features_train(a,14) + 1;
        end
    end
end
for a = 1:size(rest_train_3,1)
    for b = 1:size(rest_train_3,2)-1
        if rest_train_3(a,b)>0 && rest_train_3(a,b+1)<0 && abs(rest_train_3(a,b) - rest_train_3(a,b+1))>=10
            rest_features_train(a,15) = rest_features_train(a,15) + 1;
        elseif rest_train_3(a,b)<0 && rest_train_3(a,b+1)>0 && abs(rest_train_3(a,b) - rest_train_3(a,b+1))>=10
            rest_features_train(a,15) = rest_features_train(a,15) + 1;
        end
    end
end

for a = 1:size(rest_train_4,1)
    for b = 1:size(rest_train_4,2)-1
        if rest_train_4(a,b)>0 && rest_train_4(a,b+1)<0 && abs(rest_train_4(a,b) - rest_train_4(a,b+1))>=10
            rest_features_train(a,16) = rest_features_train(a,16) + 1;
        elseif rest_train_4(a,b)<0 && rest_train_4(a,b+1)>0 && abs(rest_train_4(a,b) - rest_train_4(a,b+1))>=10
            rest_features_train(a,16) = rest_features_train(a,16) + 1;
        end
    end
end

for a = 1:size(rest_train_5,1)
    for b = 1:size(rest_train_5,2)-1
        if rest_train_5(a,b)>0 && rest_train_5(a,b+1)<0 && abs(rest_train_5(a,b) - rest_train_5(a,b+1))>=10
            rest_features_train(a,17) = rest_features_train(a,17) + 1;
        elseif rest_train_5(a,b)<0 && rest_train_5(a,b+1)>0 && abs(rest_train_5(a,b) - rest_train_5(a,b+1))>=10
            rest_features_train(a,17) = rest_features_train(a,17) + 1;
        end
    end
end

for a = 1:size(rest_train_6,1)
    for b = 1:size(rest_train_6,2)-1
        if rest_train_6(a,b)>0 && rest_train_6(a,b+1)<0 && abs(rest_train_6(a,b) - rest_train_6(a,b+1))>=10
            rest_features_train(a,18) = rest_features_train(a,18) + 1;
        elseif rest_train_6(a,b)<0 && rest_train_6(a,b+1)>0 && abs(rest_train_6(a,b) - rest_train_6(a,b+1))>=10
            rest_features_train(a,18) = rest_features_train(a,18) + 1;
        end
    end
end

for a = 1:size(rest_test_1,1)
    for b = 1:size(rest_test_1,2)-1
        if rest_test_1(a,b)>0 && rest_test_1(a,b+1)<0 && abs(rest_test_1(a,b) - rest_test_1(a,b+1))>=10
            rest_features_test(a,13) = rest_features_test(a,13) + 1;
        elseif rest_test_1(a,b)<0 && rest_test_1(a,b+1)>0 && abs(rest_test_1(a,b) - rest_test_1(a,b+1))>=10
            rest_features_test(a,13) = rest_features_test(a,13) + 1;
        end
    end
end

for a = 1:size(rest_test_2,1)
    for b = 1:size(rest_test_2,2)-1
        if rest_test_2(a,b)>0 && rest_test_2(a,b+1)<0 && abs(rest_test_2(a,b) - rest_test_2(a,b+1))>=10
            rest_features_test(a,14) = rest_features_test(a,14) + 1;
        elseif rest_test_2(a,b)<0 && rest_test_2(a,b+1)>0 && abs(rest_test_2(a,b) - rest_test_2(a,b+1))>=10
            rest_features_test(a,14) = rest_features_test(a,14) + 1;
        end
    end
end

for a = 1:size(rest_test_3,1)
    for b = 1:size(rest_test_3,2)-1
        if rest_test_3(a,b)>0 && rest_test_3(a,b+1)<0 && abs(rest_test_3(a,b) - rest_test_3(a,b+1))>=10
            rest_features_test(a,15) = rest_features_test(a,15) + 1;
        elseif rest_test_3(a,b)<0 && rest_test_3(a,b+1)>0 && abs(rest_test_3(a,b) - rest_test_3(a,b+1))>=10
            rest_features_test(a,15) = rest_features_test(a,15) + 1;
        end
    end
end
for a = 1:size(rest_test_4,1)
    for b = 1:size(rest_test_4,2)-1
        if (rest_test_4(a,b)>0 && rest_test_4(a,b+1)<0 && abs(rest_test_4(a,b) - rest_test_4(a,b+1))>=10)
            rest_features_test(a,16) = rest_features_test(a,16) + 1;
        elseif (rest_test_4(a,b)<0 && rest_test_4(a,b+1)>0 && abs(rest_test_4(a,b) - rest_test_4(a,b+1))>=10)
            rest_features_test(a,16) = rest_features_test(a,16) + 1;
        end
    end
end

for a = 1:size(rest_test_5,1)
    for b = 1:size(rest_test_5,2)-1
        if (rest_test_5(a,b)>0 && rest_test_5(a,b+1)<0 && abs(rest_test_5(a,b) - rest_test_5(a,b+1))>=10)
            rest_features_test(a,17) = rest_features_test(a,17) + 1;
        elseif (rest_test_5(a,b)<0 && rest_test_5(a,b+1)>0 && abs(rest_test_5(a,b) - rest_test_5(a,b+1))>=10)
            rest_features_test(a,17) = rest_features_test(a,17) + 1;
        end
    end
end

for a = 1:size(rest_test_6,1)
    for b = 1:size(rest_test_6,2)-1
        if (rest_test_6(a,b)>0 && rest_test_6(a,b+1)<0 && abs(rest_test_6(a,b) - rest_test_6(a,b+1))>=10)
            rest_features_test(a,18) = rest_features_test(a,18) + 1;
        elseif (rest_test_6(a,b)<0 && rest_test_6(a,b+1)>0 && abs(rest_test_6(a,b) - rest_test_6(a,b+1))>=10)
            rest_features_test(a,18) = rest_features_test(a,18) + 1;
        end
    end
end

for a = 1:size(e_train_1,1)
    for b = 1:size(e_train_1,2)-1
        if (e_train_1(a,b)>0 && e_train_1(a,b+1)<0 && abs(e_train_1(a,b) - e_train_1(a,b+1))>=10)
            extension_features_train(a,13) = extension_features_train(a,13) + 1;
        elseif (e_train_1(a,b)<0 && e_train_1(a,b+1)>0 && abs(e_train_1(a,b) - e_train_1(a,b+1))>=10)
            extension_features_train(a,13) = extension_features_train(a,13) + 1;
        end
    end
end

for a = 1:size(e_train_2,1)
    for b = 1:size(e_train_2,2)-1
        if (e_train_2(a,b)>0 && e_train_2(a,b+1)<0 && abs(e_train_2(a,b) - e_train_2(a,b+1))>=10)
            extension_features_train(a,14) = extension_features_train(a,14) + 1;
        elseif (e_train_2(a,b)<0 && e_train_2(a,b+1)>0 && abs(e_train_2(a,b) - e_train_2(a,b+1))>=10)
            extension_features_train(a,14) = extension_features_train(a,14) + 1;
        end
    end
end

for a = 1:size(e_train_3,1)
    for b = 1:size(e_train_3,2)-1
        if (e_train_3(a,b)>0 && e_train_3(a,b+1)<0 && abs(e_train_3(a,b) - e_train_3(a,b+1))>=10)
            extension_features_train(a,15) = extension_features_train(a,15) + 1;
        elseif (e_train_3(a,b)<0 && e_train_3(a,b+1)>0 && abs(e_train_3(a,b) - e_train_3(a,b+1))>=10)
            extension_features_train(a,15) = extension_features_train(a,15) + 1;
        end
    end
end

for a = 1:size(e_train_4,1)
    for b = 1:size(e_train_4,2)-1
        if (e_train_4(a,b)>0 && e_train_4(a,b+1)<0 && abs(e_train_4(a,b) - e_train_4(a,b+1))>=10)
            extension_features_train(a,16) = extension_features_train(a,16) + 1;
        elseif (e_train_4(a,b)<0 && e_train_4(a,b+1)>0 && abs(e_train_4(a,b) - e_train_4(a,b+1))>=10)
            extension_features_train(a,16) = extension_features_train(a,16) + 1;
        end
    end
end
for a = 1:size(e_train_5,1)
    for b = 1:size(e_train_5,2)-1
        if ( e_train_5(a,b)>0 && e_train_5(a,b+1)<0 && abs(e_train_5(a,b) - e_train_5(a,b+1))>=10)
            extension_features_train(a,17) = extension_features_train(a,17) + 1;
        elseif ( e_train_5(a,b)<0 && e_train_5(a,b+1)>0 && abs(e_train_5(a,b) - e_train_5(a,b+1))>=10)
            extension_features_train(a,17) = extension_features_train(a,17) + 1;
        end
    end
end
end

for a = 1:size(e_train_6,1)
    for b = 1:size(e_train_6,2)-1
        if ( e_train_6(a,b)>0 && e_train_6(a,b+1)<0 && abs(e_train_6(a,b) - e_train_6(a,b+1))>=10)
            extension_features_train(a,18) = extension_features_train(a,18) + 1;
        elseif ( e_train_6(a,b)<0 && e_train_6(a,b+1)>0 && abs(e_train_6(a,b) - e_train_6(a,b+1))>=10)
            extension_features_train(a,18) = extension_features_train(a,18) + 1;
        end
    end
end
end

for a = 1:size(e_test_1,1)
    for b = 1:size(e_test_1,2)-1
        if ( e_test_1(a,b)>0 && e_test_1(a,b+1)<0 && abs(e_test_1(a,b) - e_test_1(a,b+1))>=10)
            extension_features_test(a,13) = extension_features_test(a,13) + 1;
        elseif ( e_test_1(a,b)<0 && e_test_1(a,b+1)>0 && abs(e_test_1(a,b) - e_test_1(a,b+1))>=10)
            extension_features_test(a,13) = extension_features_test(a,13) + 1;
        end
    end
end
end

for a = 1:size(e_test_2,1)
    for b = 1:size(e_test_2,2)-1
        if ( e_test_2(a,b)>0 && e_test_2(a,b+1)<0 && abs(e_test_2(a,b) - e_test_2(a,b+1))>=10)
            extension_features_test(a,14) = extension_features_test(a,14) + 1;
        elseif ( e_test_2(a,b)<0 && e_test_2(a,b+1)>0 && abs(e_test_2(a,b) - e_test_2(a,b+1))>=10)
            extension_features_test(a,14) = extension_features_test(a,14) + 1;
        end
    end
end
end

for a = 1:size(e_test_3,1)
    for b = 1:size(e_test_3,2)-1
        if ( e_test_3(a,b)>0 && e_test_3(a,b+1)<0 && abs(e_test_3(a,b) - e_test_3(a,b+1))>=10)
            extension_features_test(a,15) = extension_features_test(a,15) + 1;
        elseif ( e_test_3(a,b)<0 && e_test_3(a,b+1)>0 && abs(e_test_3(a,b) - e_test_3(a,b+1))>=10)
            extension_features_test(a,15) = extension_features_test(a,15) + 1;
        end
    end
end
end

for a = 1:size(e_test_4,1)
    for b = 1:size(e_test_4,2)-1
        if ( e_test_4(a,b)>0 && e_test_4(a,b+1)<0 && abs(e_test_4(a,b) - e_test_4(a,b+1))>=10)
            extension_features_test(a,16) = extension_features_test(a,16) + 1;
        elseif ( e_test_4(a,b)<0 && e_test_4(a,b+1)>0 && abs(e_test_4(a,b) - e_test_4(a,b+1))>=10)
            extension_features_test(a,16) = extension_features_test(a,16) + 1;
        end
    end
end
end

for a = 1:size(e_test_5,1)
    for b = 1:size(e_test_5,2)-1
        if ( e_test_5(a,b)>0 && e_test_5(a,b+1)<0 && abs(e_test_5(a,b) - e_test_5(a,b+1))>=10)
            extension_features_test(a,17) = extension_features_test(a,17) + 1;
        elseif ( e_test_5(a,b)<0 && e_test_5(a,b+1)>0 && abs(e_test_5(a,b) - e_test_5(a,b+1))>=10)
            extension_features_test(a,17) = extension_features_test(a,17) + 1;
        end
    end
end
end
extension_features_test(a,17) = extension_features_test(a,17) + 1;
end
end
end
for a = 1:size(e_test_6,1)
  for b = 1:size(e_test_6,2)-1
    if e_test_6(a,b)>0 & & e_test_6(a,b+1)<0 & & abs(e_test_6(a,b) - e_test_6(a,b+1))>=10)
      extension_features_test(a,18) = extension_features_test(a,18) + 1;
    elseif e_test_6(a,b)<0 & & e_test_6(a,b+1)>0 & & abs(e_test_6(a,b) - e_test_6(a,b+1))>=10)
      extension_features_test(a,18) = extension_features_test(a,18) + 1;
    end
  end
end
for a = 1:size(f_train_1,1)
  for b = 1:size(f_train_1,2)-1
    if(f_train_1(a,b)>0 & & f_train_1(a,b+1)<0 & & abs(f_train_1(a,b) - f_train_1(a,b+1))>=10)
      flexion_features_train(a,13) = flexion_features_train(a,13) + 1;
    elseif f_train_1(a,b)<0 & & f_train_1(a,b+1)>0 & & abs(f_train_1(a,b) - f_train_1(a,b+1))>=10)
      flexion_features_train(a,13) = flexion_features_train(a,13) + 1;
    end
  end
end
for a = 1:size(f_train_2,1)
  for b = 1:size(f_train_2,2)-1
    if(f_train_2(a,b)>0 & & f_train_2(a,b+1)<0 & & abs(f_train_2(a,b) - f_train_2(a,b+1))>=10)
      flexion_features_train(a,14) = flexion_features_train(a,14) + 1;
    elseif f_train_2(a,b)<0 & & f_train_2(a,b+1)>0 & & abs(f_train_2(a,b) - f_train_2(a,b+1))>=10)
      flexion_features_train(a,14) = flexion_features_train(a,14) + 1;
    end
  end
end
for a = 1:size(f_train_3,1)
  for b = 1:size(f_train_3,2)-1
    if(f_train_3(a,b)>0 & & f_train_3(a,b+1)<0 & & abs(f_train_3(a,b) - f_train_3(a,b+1))>=10)
      flexion_features_train(a,15) = flexion_features_train(a,15) + 1;
    elseif f_train_3(a,b)<0 & & f_train_3(a,b+1)>0 & & abs(f_train_3(a,b) - f_train_3(a,b+1))>=10)
      flexion_features_train(a,15) = flexion_features_train(a,15) + 1;
    end
  end
end
for a = 1:size(f_train_4,1)
  for b = 1:size(f_train_4,2)-1
    if(f_train_4(a,b)>0 & & f_train_4(a,b+1)<0 & & abs(f_train_4(a,b) - f_train_4(a,b+1))>=10)
      flexion_features_train(a,16) = flexion_features_train(a,16) + 1;
    elseif f_train_4(a,b)<0 & & f_train_4(a,b+1)>0 & & abs(f_train_4(a,b) - f_train_4(a,b+1))>=10)
      flexion_features_train(a,16) = flexion_features_train(a,16) + 1;
    end
  end
end
for a = 1:size(f_train_5,1)
  for b = 1:size(f_train_5,2)-1
    if(f_train_5(a,b)>0 & & f_train_5(a,b+1)<0 & & abs(f_train_5(a,b) - f_train_5(a,b+1))>=10)
      flexion_features_train(a,17) = flexion_features_train(a,17) + 1;
    elseif f_train_5(a,b)<0 & & f_train_5(a,b+1)>0 & & abs(f_train_5(a,b) - f_train_5(a,b+1))>=10)
      flexion_features_train(a,17) = flexion_features_train(a,17) + 1;
    end
  end
end
for a = 1:size(f_train_6,1)
  for b = 1:size(f_train_6,2)-1
    if(f_train_6(a,b)>0 & & f_train_6(a,b+1)<0 & & abs(f_train_6(a,b) - f_train_6(a,b+1))>=10)
      flexion_features_train(a,18) = flexion_features_train(a,18) + 1;
    end
  end
end
elseif (f_train_6(a,b)<0 && f_train_6(a,b+1)>0 && abs(f_train_6(a,b) - f_train_6(a,b+1))>=10)
    flexion_features_train(a,18) = flexion_features_train(a,18) + 1;
end
end
end

for a = 1:size(f_test_1,1)
    for b = 1:size(f_test_1,2)-1
        if (f_test_1(a,b)>0 && f_test_1(a,b+1)<0 && abs(f_test_1(a,b) - f_test_1(a,b+1))>=10)
            flexion_features_test(a,13) = flexion_features_test(a,13) + 1;
        elseif (f_test_1(a,b)<0 && f_test_1(a,b+1)>0 && abs(f_test_1(a,b) - f_test_1(a,b+1))>=10)
            flexion_features_test(a,13) = flexion_features_test(a,13) + 1;
        end
    end
end

for a = 1:size(f_test_2,1)
    for b = 1:size(f_test_2,2)-1
        if (f_test_2(a,b)>0 && f_test_2(a,b+1)<0 && abs(f_test_2(a,b) - f_test_2(a,b+1))>=10)
            flexion_features_test(a,14) = flexion_features_test(a,14) + 1;
        elseif (f_test_2(a,b)<0 && f_test_2(a,b+1)>0 && abs(f_test_2(a,b) - f_test_2(a,b+1))>=10)
            flexion_features_test(a,14) = flexion_features_test(a,14) + 1;
        end
    end
end

for a = 1:size(f_test_3,1)
    for b = 1:size(f_test_3,2)-1
        if (f_test_3(a,b)>0 && f_test_3(a,b+1)<0 && abs(f_test_3(a,b) - f_test_3(a,b+1))>=10)
            flexion_features_test(a,15) = flexion_features_test(a,15) + 1;
        elseif (f_test_3(a,b)<0 && f_test_3(a,b+1)>0 && abs(f_test_3(a,b) - f_test_3(a,b+1))>=10)
            flexion_features_test(a,15) = flexion_features_test(a,15) + 1;
        end
    end
end

for a = 1:size(f_test_4,1)
    for b = 1:size(f_test_4,2)-1
        if (f_test_4(a,b)>0 && f_test_4(a,b+1)<0 && abs(f_test_4(a,b) - f_test_4(a,b+1))>=10)
            flexion_features_test(a,16) = flexion_features_test(a,16) + 1;
        elseif (f_test_4(a,b)<0 && f_test_4(a,b+1)>0 && abs(f_test_4(a,b) - f_test_4(a,b+1))>=10)
            flexion_features_test(a,16) = flexion_features_test(a,16) + 1;
        end
    end
end

for a = 1:size(f_test_5,1)
    for b = 1:size(f_test_5,2)-1
        if (f_test_5(a,b)>0 && f_test_5(a,b+1)<0 && abs(f_test_5(a,b) - f_test_5(a,b+1))>=10)
            flexion_features_test(a,17) = flexion_features_test(a,17) + 1;
        elseif (f_test_5(a,b)<0 && f_test_5(a,b+1)>0 && abs(f_test_5(a,b) - f_test_5(a,b+1))>=10)
            flexion_features_test(a,17) = flexion_features_test(a,17) + 1;
        end
    end
end

for a = 1:size(f_test_6,1)
    for b = 1:size(f_test_6,2)-1
        if (f_test_6(a,b)>0 && f_test_6(a,b+1)<0 && abs(f_test_6(a,b) - f_test_6(a,b+1))>=10)
            flexion_features_test(a,18) = flexion_features_test(a,18) + 1;
        elseif (f_test_6(a,b)<0 && f_test_6(a,b+1)>0 && abs(f_test_6(a,b) - f_test_6(a,b+1))>=10)
            flexion_features_test(a,18) = flexion_features_test(a,18) + 1;
        end
    end
end

% Slope Sign Changes
for a = 1:size(g_train_1,1)
for b = 1:size(g_train_1,1)
    for a = 1:size(g_train_1,2)
        if ( g_train_1(a,b)>g_train_1(a,b+1) && g_train_1(a,b)>g_train_1(a,b-1) && (abs(g_train_1(a,b)-g_train_1(a,b+1))>=10 || abs(g_train_1(a,b)-g_train_1(a,b-1))>=10))
            grasp_features_train(a,19) = grasp_features_train(a,19) + 1;
        elseif ( g_train_1(a,b)<g_train_1(a,b+1) && g_train_1(a,b)<g_train_1(a,b-1) && (abs(g_train_1(a,b)-g_train_1(a,b+1))>=10 || abs(g_train_1(a,b)-g_train_1(a,b-1))>=10))
            grasp_features_train(a,19) = grasp_features_train(a,19) + 1;
        end
    end
end
for a = 1:size(g_train_2,1)
    for b = 1:size(g_train_2,2)
        if ( g_train_2(a,b)>g_train_2(a,b+1) && g_train_2(a,b)>g_train_2(a,b-1) && (abs(g_train_2(a,b)-g_train_2(a,b+1))>=10 || abs(g_train_2(a,b)-g_train_2(a,b-1))>=10))
            grasp_features_train(a,20) = grasp_features_train(a,20) + 1;
        elseif ( g_train_2(a,b)<g_train_2(a,b+1) && g_train_2(a,b)<g_train_2(a,b-1) && (abs(g_train_2(a,b)-g_train_2(a,b+1))>=10 || abs(g_train_2(a,b)-g_train_2(a,b-1))>=10))
            grasp_features_train(a,20) = grasp_features_train(a,20) + 1;
        end
    end
end
for a = 1:size(g_train_3,1)
    for b = 1:size(g_train_3,2)
        if ( g_train_3(a,b)>g_train_3(a,b+1) && g_train_3(a,b)>g_train_3(a,b-1) && (abs(g_train_3(a,b)-g_train_3(a,b+1))>=10 || abs(g_train_3(a,b)-g_train_3(a,b-1))>=10))
            grasp_features_train(a,21) = grasp_features_train(a,21) + 1;
        elseif ( g_train_3(a,b)<g_train_3(a,b+1) && g_train_3(a,b)<g_train_3(a,b-1) && (abs(g_train_3(a,b)-g_train_3(a,b+1))>=10 || abs(g_train_3(a,b)-g_train_3(a,b-1))>=10))
            grasp_features_train(a,21) = grasp_features_train(a,21) + 1;
        end
    end
end
for a = 1:size(g_train_4,1)
    for b = 1:size(g_train_4,2)
        if ( g_train_4(a,b)>g_train_4(a,b+1) && g_train_4(a,b)>g_train_4(a,b-1) && (abs(g_train_4(a,b)-g_train_4(a,b+1))>=10 || abs(g_train_4(a,b)-g_train_4(a,b-1))>=10))
            grasp_features_train(a,22) = grasp_features_train(a,22) + 1;
        elseif ( g_train_4(a,b)<g_train_4(a,b+1) && g_train_4(a,b)<g_train_4(a,b-1) && (abs(g_train_4(a,b)-g_train_4(a,b+1))>=10 || abs(g_train_4(a,b)-g_train_4(a,b-1))>=10))
            grasp_features_train(a,22) = grasp_features_train(a,22) + 1;
        end
    end
end
for a = 1:size(g_train_5,1)
    for b = 1:size(g_train_5,2)
        if ( g_train_5(a,b)>g_train_5(a,b+1) && g_train_5(a,b)>g_train_5(a,b-1) && (abs(g_train_5(a,b)-g_train_5(a,b+1))>=10 || abs(g_train_5(a,b)-g_train_5(a,b-1))>=10))
            grasp_features_train(a,23) = grasp_features_train(a,23) + 1;
        elseif ( g_train_5(a,b)<g_train_5(a,b+1) && g_train_5(a,b)<g_train_5(a,b-1) && (abs(g_train_5(a,b)-g_train_5(a,b+1))>=10 || abs(g_train_5(a,b)-g_train_5(a,b-1))>=10))
            grasp_features_train(a,23) = grasp_features_train(a,23) + 1;
        end
    end
end
for a = 1:size(g_train_6,1)
    for b = 1:size(g_train_6,2)
        if ( g_train_6(a,b)>g_train_6(a,b+1) && g_train_6(a,b)>g_train_6(a,b-1) && (abs(g_train_6(a,b)-g_train_6(a,b+1))>=10 || abs(g_train_6(a,b)-g_train_6(a,b-1))>=10))
            grasp_features_train(a,24) = grasp_features_train(a,24) + 1;
        elseif ( g_train_6(a,b)<g_train_6(a,b+1) && g_train_6(a,b)<g_train_6(a,b-1) && (abs(g_train_6(a,b)-g_train_6(a,b+1))>=10 || abs(g_train_6(a,b)-g_train_6(a,b-1))>=10))
            grasp_features_train(a,24) = grasp_features_train(a,24) + 1;
        end
    end
end
for a = 1:size(g_test_1,1)
    for b = 2:size(g_test_1,2)-1
        if ( g_test_1(a,b)>g_test_1(a,b-1) && g_test_1(a,b)>g_test_1(a,b+1) && (abs(g_test_1(a,b)-g_test_1(a,b-1))>=10 || abs(g_test_1(a,b)-g_test_1(a,b+1))>=10) )
            grasp_features_test(a,19) = grasp_features_test(a,19) + 1;
        elseif ( g_test_1(a,b)<g_test_1(a,b-1) && g_test_1(a,b)<g_test_1(a,b+1) && (abs(g_test_1(a,b)-g_test_1(a,b-1))>=10 || abs(g_test_1(a,b)-g_test_1(a,b+1))>=10) )
            grasp_features_test(a,19) = grasp_features_test(a,19) + 1;
        end
    end
end
for a = 1:size(g_test_2,1)
    for b = 2:size(g_test_2,2)-1
        if ( g_test_2(a,b)>g_test_2(a,b-1) && g_test_2(a,b)>g_test_2(a,b+1) && (abs(g_test_2(a,b)-g_test_2(a,b-1))>=10 || abs(g_test_2(a,b)-g_test_2(a,b+1))>=10) )
            grasp_features_test(a,20) = grasp_features_test(a,20) + 1;
        elseif ( g_test_2(a,b)<g_test_2(a,b-1) && g_test_2(a,b)<g_test_2(a,b+1) && (abs(g_test_2(a,b)-g_test_2(a,b-1))>=10 || abs(g_test_2(a,b)-g_test_2(a,b+1))>=10) )
            grasp_features_test(a,20) = grasp_features_test(a,20) + 1;
        end
    end
end
for a = 1:size(g_test_3,1)
    for b = 2:size(g_test_3,2)-1
        if ( g_test_3(a,b)>g_test_3(a,b-1) && g_test_3(a,b)>g_test_3(a,b+1) && (abs(g_test_3(a,b)-g_test_3(a,b-1))>=10 || abs(g_test_3(a,b)-g_test_3(a,b+1))>=10) )
            grasp_features_test(a,21) = grasp_features_test(a,21) + 1;
        elseif ( g_test_3(a,b)<g_test_3(a,b-1) && g_test_3(a,b)<g_test_3(a,b+1) && (abs(g_test_3(a,b)-g_test_3(a,b-1))>=10 || abs(g_test_3(a,b)-g_test_3(a,b+1))>=10) )
            grasp_features_test(a,21) = grasp_features_test(a,21) + 1;
        end
    end
end
for a = 1:size(g_test_4,1)
    for b = 2:size(g_test_4,2)-1
        if ( g_test_4(a,b)>g_test_4(a,b-1) && g_test_4(a,b)>g_test_4(a,b+1) && (abs(g_test_4(a,b)-g_test_4(a,b-1))>=10 || abs(g_test_4(a,b)-g_test_4(a,b+1))>=10) )
            grasp_features_test(a,22) = grasp_features_test(a,22) + 1;
        elseif ( g_test_4(a,b)<g_test_4(a,b-1) && g_test_4(a,b)<g_test_4(a,b+1) && (abs(g_test_4(a,b)-g_test_4(a,b-1))>=10 || abs(g_test_4(a,b)-g_test_4(a,b+1))>=10) )
            grasp_features_test(a,22) = grasp_features_test(a,22) + 1;
        end
    end
end
for a = 1:size(g_test_5,1)
    for b = 2:size(g_test_5,2)-1
        if ( g_test_5(a,b)>g_test_5(a,b-1) && g_test_5(a,b)>g_test_5(a,b+1) && (abs(g_test_5(a,b)-g_test_5(a,b-1))>=10 || abs(g_test_5(a,b)-g_test_5(a,b+1))>=10) )
            grasp_features_test(a,23) = grasp_features_test(a,23) + 1;
        elseif ( g_test_5(a,b)<g_test_5(a,b-1) && g_test_5(a,b)<g_test_5(a,b+1) && (abs(g_test_5(a,b)-g_test_5(a,b-1))>=10 || abs(g_test_5(a,b)-g_test_5(a,b+1))>=10) )
            grasp_features_test(a,23) = grasp_features_test(a,23) + 1;
        end
    end
end
for a = 1:size(g_test_6,1)
    for b = 2:size(g_test_6,2)-1
        if ( g_test_6(a,b)>g_test_6(a,b-1) && g_test_6(a,b)>g_test_6(a,b+1) && (abs(g_test_6(a,b)-g_test_6(a,b-1))>=10 || abs(g_test_6(a,b)-g_test_6(a,b+1))>=10) )
            grasp_features_test(a,24) = grasp_features_test(a,24) + 1;
        elseif ( g_test_6(a,b)<g_test_6(a,b-1) && g_test_6(a,b)<g_test_6(a,b+1) && (abs(g_test_6(a,b)-g_test_6(a,b-1))>=10 || abs(g_test_6(a,b)-g_test_6(a,b+1))>=10) )
            grasp_features_test(a,24) = grasp_features_test(a,24) + 1;
        end
    end
end

grasp_features_test(a,24) = grasp_features_test(a,24) + 1;
elseif g_test_6(a,b)<g_test_6(a,b-1) & & g_test_6(a,b)<g_test_6(a,b+1) & & (abs(g_test_6(a,b)-g_test_6(a,b-1))>=10 || abs(g_test_6(a,b)-g_test_6(a,b+1))>=10)
  grasp_features_test(a,24) = grasp_features_test(a,24) + 1;
end
end
for a = 1:size(r_train_1,1)
  for b = 2:size(r_train_1,2)-1
    if r_train_1(a,b)>r_train_1(a,b-1) & & r_train_1(a,b)>r_train_1(a,b+1) & & (abs(r_train_1(a,b)-r_train_1(a,b-1))>=10 || abs(r_train_1(a,b)-r_train_1(a,b+1))>=10)
      releasing_features_train(a,19) = releasing_features_train(a,19) + 1;
    elseif r_train_1(a,b)<r_train_1(a,b-1) & & r_train_1(a,b)<r_train_1(a,b+1) & & (abs(r_train_1(a,b)-r_train_1(a,b-1))>=10 || abs(r_train_1(a,b)-r_train_1(a,b+1))>=10)
      releasing_features_train(a,19) = releasing_features_train(a,19) + 1;
    end
  end
end
for a = 1:size(r_train_2,1)
  for b = 2:size(r_train_2,2)-1
    if r_train_2(a,b)>r_train_2(a,b-1) & & r_train_2(a,b)>r_train_2(a,b+1) & & (abs(r_train_2(a,b)-r_train_2(a,b-1))>=10 || abs(r_train_2(a,b)-r_train_2(a,b+1))>=10)
      releasing_features_train(a,20) = releasing_features_train(a,20) + 1;
    elseif r_train_2(a,b)<r_train_2(a,b-1) & & r_train_2(a,b)<r_train_2(a,b+1) & & (abs(r_train_2(a,b)-r_train_2(a,b-1))>=10 || abs(r_train_2(a,b)-r_train_2(a,b+1))>=10)
      releasing_features_train(a,20) = releasing_features_train(a,20) + 1;
    end
  end
end
for a = 1:size(r_train_3,1)
  for b = 2:size(r_train_3,2)-1
    if r_train_3(a,b)>r_train_3(a,b-1) & & r_train_3(a,b)>r_train_3(a,b+1) & & (abs(r_train_3(a,b)-r_train_3(a,b-1))>=10 || abs(r_train_3(a,b)-r_train_3(a,b+1))>=10)
      releasing_features_train(a,21) = releasing_features_train(a,21) + 1;
    elseif r_train_3(a,b)<r_train_3(a,b-1) & & r_train_3(a,b)<r_train_3(a,b+1) & & (abs(r_train_3(a,b)-r_train_3(a,b-1))>=10 || abs(r_train_3(a,b)-r_train_3(a,b+1))>=10)
      releasing_features_train(a,21) = releasing_features_train(a,21) + 1;
    end
  end
end
for a = 1:size(r_train_4,1)
  for b = 2:size(r_train_4,2)-1
    if r_train_4(a,b)>r_train_4(a,b-1) & & r_train_4(a,b)>r_train_4(a,b+1) & & (abs(r_train_4(a,b)-r_train_4(a,b-1))>=10 || abs(r_train_4(a,b)-r_train_4(a,b+1))>=10)
      releasing_features_train(a,22) = releasing_features_train(a,22) + 1;
    elseif r_train_4(a,b)<r_train_4(a,b-1) & & r_train_4(a,b)<r_train_4(a,b+1) & & (abs(r_train_4(a,b)-r_train_4(a,b-1))>=10 || abs(r_train_4(a,b)-r_train_4(a,b+1))>=10)
      releasing_features_train(a,22) = releasing_features_train(a,22) + 1;
    end
  end
end
for a = 1:size(r_train_5,1)
  for b = 2:size(r_train_5,2)-1
    if r_train_5(a,b)>r_train_5(a,b-1) & & r_train_5(a,b)>r_train_5(a,b+1) & & (abs(r_train_5(a,b)-r_train_5(a,b-1))>=10 || abs(r_train_5(a,b)-r_train_5(a,b+1))>=10)
      releasing_features_train(a,23) = releasing_features_train(a,23) + 1;
    elseif r_train_5(a,b)<r_train_5(a,b-1) & & r_train_5(a,b)<r_train_5(a,b+1) & & (abs(r_train_5(a,b)-r_train_5(a,b-1))>=10 || abs(r_train_5(a,b)-r_train_5(a,b+1))>=10)
      releasing_features_train(a,23) = releasing_features_train(a,23) + 1;
    end
  end
end
end
for a = 1:size(r_train_6,1)
    for b = 2:size(r_train_6,2)-1
        if r_train_6(a,b)>r_train_6(a,b-1) && r_train_6(a,b)>r_train_6(a,b+1) && (abs(r_train_6(a,b)-r_train_6(a,b+1))>=10 || abs(r_train_6(a,b)-r_train_6(a,b-1))>=10)
            releasing_features_train(a,24) = releasing_features_train(a,24) + 1;
        elseif r_train_6(a,b)<r_train_6(a,b-1) && r_train_6(a,b)<r_train_6(a,b+1) && (abs(r_train_6(a,b)-r_train_6(a,b-1))>=10 || abs(r_train_6(a,b)-r_train_6(a,b+1))>=10)
            releasing_features_train(a,24) = releasing_features_train(a,24) + 1;
        end
    end
end

for a = 1:size(r_test_1,1)
    for b = 2:size(r_test_1,2)-1
        if r_test_1(a,b)>r_test_1(a,b-1) && r_test_1(a,b)>r_test_1(a,b+1) && (abs(r_test_1(a,b)-r_test_1(a,b+1))>=10 || abs(r_test_1(a,b)-r_test_1(a,b-1))>=10)
            releasing_features_test(a,19) = releasing_features_test(a,19) + 1;
        elseif r_test_1(a,b)<r_test_1(a,b-1) && r_test_1(a,b)<r_test_1(a,b+1) && (abs(r_test_1(a,b)-r_test_1(a,b-1))>=10 || abs(r_test_1(a,b)-r_test_1(a,b+1))>=10)
            releasing_features_test(a,19) = releasing_features_test(a,19) + 1;
        end
    end
end

for a = 1:size(r_test_2,1)
    for b = 2:size(r_test_2,2)-1
        if r_test_2(a,b)>r_test_2(a,b-1) && r_test_2(a,b)>r_test_2(a,b+1) && (abs(r_test_2(a,b)-r_test_2(a,b+1))>=10 || abs(r_test_2(a,b)-r_test_2(a,b-1))>=10)
            releasing_features_test(a,20) = releasing_features_test(a,20) + 1;
        elseif r_test_2(a,b)<r_test_2(a,b-1) && r_test_2(a,b)<r_test_2(a,b+1) && (abs(r_test_2(a,b)-r_test_2(a,b-1))>=10 || abs(r_test_2(a,b)-r_test_2(a,b+1))>=10)
            releasing_features_test(a,20) = releasing_features_test(a,20) + 1;
        end
    end
end

for a = 1:size(r_test_3,1)
    for b = 2:size(r_test_3,2)-1
        if r_test_3(a,b)>r_test_3(a,b-1) && r_test_3(a,b)>r_test_3(a,b+1) && (abs(r_test_3(a,b)-r_test_3(a,b+1))>=10 || abs(r_test_3(a,b)-r_test_3(a,b-1))>=10)
            releasing_features_test(a,21) = releasing_features_test(a,21) + 1;
        elseif r_test_3(a,b)<r_test_3(a,b-1) && r_test_3(a,b)<r_test_3(a,b+1) && (abs(r_test_3(a,b)-r_test_3(a,b-1))>=10 || abs(r_test_3(a,b)-r_test_3(a,b+1))>=10)
            releasing_features_test(a,21) = releasing_features_test(a,21) + 1;
        end
    end
end

for a = 1:size(r_test_4,1)
    for b = 2:size(r_test_4,2)-1
        if r_test_4(a,b)>r_test_4(a,b-1) && r_test_4(a,b)>r_test_4(a,b+1) && (abs(r_test_4(a,b)-r_test_4(a,b+1))>=10 || abs(r_test_4(a,b)-r_test_4(a,b-1))>=10)
            releasing_features_test(a,22) = releasing_features_test(a,22) + 1;
        elseif r_test_4(a,b)<r_test_4(a,b-1) && r_test_4(a,b)<r_test_4(a,b+1) && (abs(r_test_4(a,b)-r_test_4(a,b-1))>=10 || abs(r_test_4(a,b)-r_test_4(a,b+1))>=10)
            releasing_features_test(a,22) = releasing_features_test(a,22) + 1;
        end
    end
end

for a = 1:size(r_test_5,1)
    for b = 2:size(r_test_5,2)-1
        if r_test_5(a,b)>r_test_5(a,b-1) && r_test_5(a,b)>r_test_5(a,b+1) && (abs(r_test_5(a,b)-r_test_5(a,b+1))>=10 || abs(r_test_5(a,b)-r_test_5(a,b-1))>=10)
            releasing_features_test(a,23) = releasing_features_test(a,23) + 1;
        elseif r_test_5(a,b)<r_test_5(a,b-1) && r_test_5(a,b)<r_test_5(a,b+1) && (abs(r_test_5(a,b)-r_test_5(a,b-1))>=10 || abs(r_test_5(a,b)-r_test_5(a,b+1))>=10)
            releasing_features_test(a,23) = releasing_features_test(a,23) + 1;
        end
    end
end
releasing_features_test(a,23) = releasing_features_test(a,23) + 1;

end
end

for a = 1:size(r_test_6,1)
    for b = 2:size(r_test_6,2)-1
        if ( r_test_6(a,b)>r_test_6(a,b-1) && r_test_6(a,b)>r_test_6(a,b+1) && (abs(r_test_6(a,b)-r_test_6(a,b-1))>=10 || abs(r_test_6(a,b)-r_test_6(a,b+1))>=10) )
            releasing_features_test(a,24) = releasing_features_test(a,24) + 1;
        elseif ( r_test_6(a,b)<r_test_6(a,b-1) && r_test_6(a,b)<r_test_6(a,b+1) && (abs(r_test_6(a,b)-r_test_6(a,b-1))>=10 || abs(r_test_6(a,b)-r_test_6(a,b+1))>=10) )
            releasing_features_test(a,24) = releasing_features_test(a,24) + 1;
        end
    end
end

for a = 1:size(p_train_1,1)
    for b = 2:size(p_train_1,2)-1
        if ( p_train_1(a,b)>p_train_1(a,b-1) && p_train_1(a,b)>p_train_1(a,b+1) && (abs(p_train_1(a,b)-p_train_1(a,b-1))>=10 || abs(p_train_1(a,b)-p_train_1(a,b+1))>=10) )
            pronation_features_train(a,19) = pronation_features_train(a,19) + 1;
        elseif ( p_train_1(a,b)<p_train_1(a,b-1) && p_train_1(a,b)<p_train_1(a,b+1) && (abs(p_train_1(a,b)-p_train_1(a,b-1))>=10 || abs(p_train_1(a,b)-p_train_1(a,b+1))>=10) )
            pronation_features_train(a,19) = pronation_features_train(a,19) + 1;
        end
    end
end

for a = 1:size(p_train_2,1)
    for b = 2:size(p_train_2,2)-1
        if ( p_train_2(a,b)>p_train_2(a,b-1) && p_train_2(a,b)>p_train_2(a,b+1) && (abs(p_train_2(a,b)-p_train_2(a,b-1))>=10 || abs(p_train_2(a,b)-p_train_2(a,b+1))>=10) )
            pronation_features_train(a,20) = pronation_features_train(a,20) + 1;
        elseif ( p_train_2(a,b)<p_train_2(a,b-1) && p_train_2(a,b)<p_train_2(a,b+1) && (abs(p_train_2(a,b)-p_train_2(a,b-1))>=10 || abs(p_train_2(a,b)-p_train_2(a,b+1))>=10) )
            pronation_features_train(a,20) = pronation_features_train(a,20) + 1;
        end
    end
end

for a = 1:size(p_train_3,1)
    for b = 2:size(p_train_3,2)-1
        if ( p_train_3(a,b)>p_train_3(a,b-1) && p_train_3(a,b)>p_train_3(a,b+1) && (abs(p_train_3(a,b)-p_train_3(a,b-1))>=10 || abs(p_train_3(a,b)-p_train_3(a,b+1))>=10) )
            pronation_features_train(a,21) = pronation_features_train(a,21) + 1;
        elseif ( p_train_3(a,b)<p_train_3(a,b-1) && p_train_3(a,b)<p_train_3(a,b+1) && (abs(p_train_3(a,b)-p_train_3(a,b-1))>=10 || abs(p_train_3(a,b)-p_train_3(a,b+1))>=10) )
            pronation_features_train(a,21) = pronation_features_train(a,21) + 1;
        end
    end
end

for a = 1:size(p_train_4,1)
    for b = 2:size(p_train_4,2)-1
        if ( p_train_4(a,b)>p_train_4(a,b-1) && p_train_4(a,b)>p_train_4(a,b+1) && (abs(p_train_4(a,b)-p_train_4(a,b-1))>=10 || abs(p_train_4(a,b)-p_train_4(a,b+1))>=10) )
            pronation_features_train(a,22) = pronation_features_train(a,22) + 1;
        elseif ( p_train_4(a,b)<p_train_4(a,b-1) && p_train_4(a,b)<p_train_4(a,b+1) && (abs(p_train_4(a,b)-p_train_4(a,b-1))>=10 || abs(p_train_4(a,b)-p_train_4(a,b+1))>=10) )
            pronation_features_train(a,22) = pronation_features_train(a,22) + 1;
        end
    end
end

for a = 1:size(p_train_5,1)
for b = 2:size(p_train_5,2)-1
    if ( p_train_5(a,b)>p_train_5(a,b-1) && p_train_5(a,b)>p_train_5(a,b+1) && (abs(p_train_5(a,b)-p_train_5(a,b+1))>=10 || abs(p_train_5(a,b)-p_train_5(a,b-1))>=10) )
        pronation_features_train(a,23) = pronation_features_train(a,23) + 1;
    elseif ( p_train_5(a,b)<p_train_5(a,b-1) && p_train_5(a,b)<p_train_5(a,b+1) && (abs(p_train_5(a,b)-p_train_5(a,b-1))>=10 || abs(p_train_5(a,b)-p_train_5(a,b+1))>=10) )
        pronation_features_train(a,23) = pronation_features_train(a,23) + 1;
    end
end
for a = 1:size(p_train_6,1)
    for b = 2:size(p_train_6,2)-1
        if ( p_train_6(a,b)>p_train_6(a,b-1) && p_train_6(a,b)>p_train_6(a,b+1) && (abs(p_train_6(a,b)-p_train_6(a,b+1))>=10 || abs(p_train_6(a,b)-p_train_6(a,b-1))>=10) )
            pronation_features_train(a,24) = pronation_features_train(a,24) + 1;
        elseif ( p_train_6(a,b)<p_train_6(a,b-1) && p_train_6(a,b)<p_train_6(a,b+1) && (abs(p_train_6(a,b)-p_train_6(a,b-1))>=10 || abs(p_train_6(a,b)-p_train_6(a,b+1))>=10) )
            pronation_features_train(a,24) = pronation_features_train(a,24) + 1;
        end
    end
end
for a = 1:size(p_test_1,1)
    for b = 2:size(p_test_1,2)-1
        if ( p_test_1(a,b)>p_test_1(a,b-1) && p_test_1(a,b)>p_test_1(a,b+1) && (abs(p_test_1(a,b)-p_test_1(a,b+1))>=10 || abs(p_test_1(a,b)-p_test_1(a,b-1))>=10) )
            pronation_features_test(a,19) = pronation_features_test(a,19) + 1;
        elseif ( p_test_1(a,b)<p_test_1(a,b-1) && p_test_1(a,b)<p_test_1(a,b+1) && (abs(p_test_1(a,b)-p_test_1(a,b-1))>=10 || abs(p_test_1(a,b)-p_test_1(a,b+1))>=10) )
            pronation_features_test(a,19) = pronation_features_test(a,19) + 1;
        end
    end
end
for a = 1:size(p_test_2,1)
    for b = 2:size(p_test_2,2)-1
        if ( p_test_2(a,b)>p_test_2(a,b-1) && p_test_2(a,b)>p_test_2(a,b+1) && (abs(p_test_2(a,b)-p_test_2(a,b+1))>=10 || abs(p_test_2(a,b)-p_test_2(a,b-1))>=10) )
            pronation_features_test(a,20) = pronation_features_test(a,20) + 1;
        elseif ( p_test_2(a,b)<p_test_2(a,b-1) && p_test_2(a,b)<p_test_2(a,b+1) && (abs(p_test_2(a,b)-p_test_2(a,b-1))>=10 || abs(p_test_2(a,b)-p_test_2(a,b+1))>=10) )
            pronation_features_test(a,20) = pronation_features_test(a,20) + 1;
        end
    end
end
for a = 1:size(p_test_3,1)
    for b = 2:size(p_test_3,2)-1
        if ( p_test_3(a,b)>p_test_3(a,b-1) && p_test_3(a,b)>p_test_3(a,b+1) && (abs(p_test_3(a,b)-p_test_3(a,b+1))>=10 || abs(p_test_3(a,b)-p_test_3(a,b-1))>=10) )
            pronation_features_test(a,21) = pronation_features_test(a,21) + 1;
        elseif ( p_test_3(a,b)<p_test_3(a,b-1) && p_test_3(a,b)<p_test_3(a,b+1) && (abs(p_test_3(a,b)-p_test_3(a,b-1))>=10 || abs(p_test_3(a,b)-p_test_3(a,b+1))>=10) )
            pronation_features_test(a,21) = pronation_features_test(a,21) + 1;
        end
    end
end
for a = 1:size(p_test_4,1)
    for b = 2:size(p_test_4,2)-1
        if ( p_test_4(a,b)>p_test_4(a,b-1) && p_test_4(a,b)>p_test_4(a,b+1) && (abs(p_test_4(a,b)-p_test_4(a,b+1))>=10 || abs(p_test_4(a,b)-p_test_4(a,b-1))>=10) )
            pronation_features_test(a,22) = pronation_features_test(a,22) + 1;
        elseif ( p_test_4(a,b)<p_test_4(a,b-1) && p_test_4(a,b)<p_test_4(a,b+1) && (abs(p_test_4(a,b)-p_test_4(a,b-1))>=10 || abs(p_test_4(a,b)-p_test_4(a,b+1))>=10) )
            pronation_features_test(a,22) = pronation_features_test(a,22) + 1;
        end
    end
end
for a = 1:size(p_test_5,1)
    for b = 2:size(p_test_5,2)-1
        if ( p_test_5(a,b)>p_test_5(a,b-1) && p_test_5(a,b)>p_test_5(a,b+1) && (abs(p_test_5(a,b)-p_test_5(a,b-1))>=10 || abs(p_test_5(a,b)-p_test_5(a,b+1))>=10) )
            pronation_features_test(a,23) = pronation_features_test(a,23) + 1;
        else if ( p_test_5(a,b)<p_test_5(a,b-1) && p_test_5(a,b)<p_test_5(a,b+1) && (abs(p_test_5(a,b)-p_test_5(a,b-1))>=10 || abs(p_test_5(a,b)-p_test_5(a,b+1))>=10) )
            pronation_features_test(a,23) = pronation_features_test(a,23) + 1;
        end
    end
end

for a = 1:size(p_test_6,1)
    for b = 2:size(p_test_6,2)-1
        if ( p_test_6(a,b)>p_test_6(a,b-1) && p_test_6(a,b)>p_test_6(a,b+1) && (abs(p_test_6(a,b)-p_test_6(a,b-1))>=10 || abs(p_test_6(a,b)-p_test_6(a,b+1))>=10) )
            pronation_features_test(a,24) = pronation_features_test(a,24) + 1;
        else if ( p_test_6(a,b)<p_test_6(a,b-1) && p_test_6(a,b)<p_test_6(a,b+1) && (abs(p_test_6(a,b)-p_test_6(a,b-1))>=10 || abs(p_test_6(a,b)-p_test_6(a,b+1))>=10) )
            pronation_features_test(a,24) = pronation_features_test(a,24) + 1;
        end
    end
end

for a = 1:size(s_train_1,1)
    for b = 2:size(s_train_1,2)-1
        if ( s_train_1(a,b)>s_train_1(a,b-1) && s_train_1(a,b)>s_train_1(a,b+1) && (abs(s_train_1(a,b)-s_train_1(a,b-1))>=10 || abs(s_train_1(a,b)-s_train_1(a,b+1))>=10) )
            supination_features_train(a,19) = supination_features_train(a,19) + 1;
        else if ( s_train_1(a,b)<s_train_1(a,b-1) && s_train_1(a,b)<s_train_1(a,b+1) && (abs(s_train_1(a,b)-s_train_1(a,b-1))>=10 || abs(s_train_1(a,b)-s_train_1(a,b+1))>=10) )
            supination_features_train(a,19) = supination_features_train(a,19) + 1;
        end
    end
end

for a = 1:size(s_train_2,1)
    for b = 2:size(s_train_2,2)-1
        if ( s_train_2(a,b)>s_train_2(a,b-1) && s_train_2(a,b)>s_train_2(a,b+1) && (abs(s_train_2(a,b)-s_train_2(a,b-1))>=10 || abs(s_train_2(a,b)-s_train_2(a,b+1))>=10) )
            supination_features_train(a,20) = supination_features_train(a,20) + 1;
        else if ( s_train_2(a,b)<s_train_2(a,b-1) && s_train_2(a,b)<s_train_2(a,b+1) && (abs(s_train_2(a,b)-s_train_2(a,b-1))>=10 || abs(s_train_2(a,b)-s_train_2(a,b+1))>=10) )
            supination_features_train(a,20) = supination_features_train(a,20) + 1;
        end
    end
end

for a = 1:size(s_train_3,1)
    for b = 2:size(s_train_3,2)-1
        if ( s_train_3(a,b)>s_train_3(a,b-1) && s_train_3(a,b)>s_train_3(a,b+1) && (abs(s_train_3(a,b)-s_train_3(a,b-1))>=10 || abs(s_train_3(a,b)-s_train_3(a,b+1))>=10) )
            supination_features_train(a,21) = supination_features_train(a,21) + 1;
        else if ( s_train_3(a,b)<s_train_3(a,b-1) && s_train_3(a,b)<s_train_3(a,b+1) && (abs(s_train_3(a,b)-s_train_3(a,b-1))>=10 || abs(s_train_3(a,b)-s_train_3(a,b+1))>=10) )
            supination_features_train(a,21) = supination_features_train(a,21) + 1;
        end
    end
end

for a = 1:size(s_train_4,1)
    for b = 2:size(s_train_4,2)-1
        if ( s_train_4(a,b)>s_train_4(a,b-1) && s_train_4(a,b)>s_train_4(a,b+1) && (abs(s_train_4(a,b)-s_train_4(a,b-1))>=10 || abs(s_train_4(a,b)-s_train_4(a,b+1))>=10) )
            supination_features_train(a,21) = supination_features_train(a,21) + 1;
        else if ( s_train_4(a,b)<s_train_4(a,b-1) && s_train_4(a,b)<s_train_4(a,b+1) && (abs(s_train_4(a,b)-s_train_4(a,b-1))>=10 || abs(s_train_4(a,b)-s_train_4(a,b+1))>=10) )
            supination_features_train(a,21) = supination_features_train(a,21) + 1;
        end
    end
end
supination_features_train(a,22) = supination_features_train(a,22) + 1;

elseif s_train_4(a,b)<s_train_4(a,b-1) && s_train_4(a,b)<s_train_4(a,b+1) && (abs(s_train_4(a,b)-s_train_4(a,b-1))>=10 || abs(s_train_4(a,b)-s_train_4(a,b+1))>=10)
  supination_features_train(a,22) = supination_features_train(a,22) + 1;
end
end
end

for a = 1:size(s_train_5,1)
  for b = 2:size(s_train_5,2)-1
    if s_train_5(a,b)>s_train_5(a,b-1) && s_train_5(a,b)>s_train_5(a,b+1) && (abs(s_train_5(a,b)-s_train_5(a,b-1))>=10 || abs(s_train_5(a,b)-s_train_5(a,b+1))>=10)
      supination_features_train(a,23) = supination_features_train(a,23) + 1;
    elseif s_train_5(a,b)<s_train_5(a,b-1) && s_train_5(a,b)<s_train_5(a,b+1) && (abs(s_train_5(a,b)-s_train_5(a,b-1))>=10 || abs(s_train_5(a,b)-s_train_5(a,b+1))>=10)
      supination_features_train(a,23) = supination_features_train(a,23) + 1;
    end
  end
end

for a = 1:size(s_test_1,1)
  for b = 2:size(s_test_1,2)-1
    if s_test_1(a,b)>s_test_1(a,b-1) && s_test_1(a,b)>s_test_1(a,b+1) && (abs(s_test_1(a,b)-s_test_1(a,b-1))>=10 || abs(s_test_1(a,b)-s_test_1(a,b+1))>=10)
      supination_features_test(a,19) = supination_features_test(a,19) + 1;
    elseif s_test_1(a,b)<s_test_1(a,b-1) && s_test_1(a,b)<s_test_1(a,b+1) && (abs(s_test_1(a,b)-s_test_1(a,b-1))>=10 || abs(s_test_1(a,b)-s_test_1(a,b+1))>=10)
      supination_features_test(a,19) = supination_features_test(a,19) + 1;
    end
  end
end

for a = 1:size(s_test_2,1)
  for b = 2:size(s_test_2,2)-1
    if s_test_2(a,b)>s_test_2(a,b-1) && s_test_2(a,b)>s_test_2(a,b+1) && (abs(s_test_2(a,b)-s_test_2(a,b-1))>=10 || abs(s_test_2(a,b)-s_test_2(a,b+1))>=10)
      supination_features_test(a,20) = supination_features_test(a,20) + 1;
    elseif s_test_2(a,b)<s_test_2(a,b-1) && s_test_2(a,b)<s_test_2(a,b+1) && (abs(s_test_2(a,b)-s_test_2(a,b-1))>=10 || abs(s_test_2(a,b)-s_test_2(a,b+1))>=10)
      supination_features_test(a,20) = supination_features_test(a,20) + 1;
    end
  end
end

for a = 1:size(s_test_3,1)
  for b = 2:size(s_test_3,2)-1
    if s_test_3(a,b)>s_test_3(a,b-1) && s_test_3(a,b)>s_test_3(a,b+1) && (abs(s_test_3(a,b)-s_test_3(a,b-1))>=10 || abs(s_test_3(a,b)-s_test_3(a,b+1))>=10)
      supination_features_test(a,21) = supination_features_test(a,21) + 1;
    elseif s_test_3(a,b)<s_test_3(a,b-1) && s_test_3(a,b)<s_test_3(a,b+1) && (abs(s_test_3(a,b)-s_test_3(a,b-1))>=10 || abs(s_test_3(a,b)-s_test_3(a,b+1))>=10)
      supination_features_test(a,21) = supination_features_test(a,21) + 1;
    end
  end
end

for a = 1:size(s_test_4,1)
  for b = 2:size(s_test_4,2)-1
    if (s_test_4(a,b)>s_test_4(a,b-1) & & s_test_4(a,b)>s_test_4(a,b+1) & & (abs(s_test_4(a,b)-s_test_4(a,b-1))>=10 || abs(s_test_4(a,b)-s_test_4(a,b+1))>=10)
      supination_features_test(a,22) = supination_features_test(a,22) + 1;
    elseif (s_test_4(a,b)<s_test_4(a,b-1) & & s_test_4(a,b)<s_test_4(a,b+1) & & (abs(s_test_4(a,b)-s_test_4(a,b-1))>=10 || abs(s_test_4(a,b)-s_test_4(a,b+1))>=10)
      supination_features_test(a,22) = supination_features_test(a,22) + 1;
    end
  end
end

for a = 1:size(s_test_5,1)
  for b = 2:size(s_test_5,2)-1
    if (s_test_5(a,b)>s_test_5(a,b-1) & & s_test_5(a,b)>s_test_5(a,b+1) & & (abs(s_test_5(a,b)-s_test_5(a,b-1))>=10 || abs(s_test_5(a,b)-s_test_5(a,b+1))>=10)
      supination_features_test(a,23) = supination_features_test(a,23) + 1;
    elseif (s_test_5(a,b)<s_test_5(a,b-1) & & s_test_5(a,b)<s_test_5(a,b+1) & & (abs(s_test_5(a,b)-s_test_5(a,b-1))>=10 || abs(s_test_5(a,b)-s_test_5(a,b+1))>=10)
      supination_features_test(a,23) = supination_features_test(a,23) + 1;
    end
  end
end

for a = 1:size(s_test_6,1)
  for b = 2:size(s_test_6,2)-1
    if (s_test_6(a,b)>s_test_6(a,b-1) & & s_test_6(a,b)>s_test_6(a,b+1) & & (abs(s_test_6(a,b)-s_test_6(a,b-1))>=10 || abs(s_test_6(a,b)-s_test_6(a,b+1))>=10)
      supination_features_test(a,24) = supination_features_test(a,24) + 1;
    elseif (s_test_6(a,b)<s_test_6(a,b-1) & & s_test_6(a,b)<s_test_6(a,b+1) & & (abs(s_test_6(a,b)-s_test_6(a,b-1))>=10 || abs(s_test_6(a,b)-s_test_6(a,b+1))>=10)
      supination_features_test(a,24) = supination_features_test(a,24) + 1;
    end
  end
end

for a = 1:size(rest_train_1)
  for b = 2:size(rest_train_1,2)-1
    if (rest_train_1(a,b)>rest_train_1(a,b-1) & & rest_train_1(a,b)>rest_train_1(a,b+1) & & (abs(rest_train_1(a,b)-rest_train_1(a,b-1))>=10 || abs(rest_train_1(a,b)-rest_train_1(a,b+1))>=10)
      rest_features_train(a,19) = rest_features_train(a,19) + 1;
    elseif (rest_train_1(a,b)<rest_train_1(a,b-1) & & rest_train_1(a,b)<rest_train_1(a,b+1) & & (abs(rest_train_1(a,b)-rest_train_1(a,b-1))>=10 || abs(rest_train_1(a,b)-rest_train_1(a,b+1))>=10)
      rest_features_train(a,19) = rest_features_train(a,19) + 1;
    end
  end
end

for a = 1:size(rest_train_2)
  for b = 2:size(rest_train_2,2)-1
    if (rest_train_2(a,b)>rest_train_2(a,b-1) & & rest_train_2(a,b)>rest_train_2(a,b+1) & & (abs(rest_train_2(a,b)-rest_train_2(a,b-1))>=10 || abs(rest_train_2(a,b)-rest_train_2(a,b+1))>=10)
      rest_features_train(a,20) = rest_features_train(a,20) + 1;
    elseif (rest_train_2(a,b)<rest_train_2(a,b-1) & & rest_train_2(a,b)<rest_train_2(a,b+1) & & (abs(rest_train_2(a,b)-rest_train_2(a,b-1))>=10 || abs(rest_train_2(a,b)-rest_train_2(a,b+1))>=10)
      rest_features_train(a,20) = rest_features_train(a,20) + 1;
    end
  end
end

for a = 1:size(rest_train_3)
  for b = 2:size(rest_train_3,2)-1
    if (r_train_3(a,b)>r_train_3(a,b-1) & & r_train_3(a,b)>r_train_3(a,b+1) & & (abs(r_train_3(a,b)-r_train_3(a,b-1))>=10 || abs(r_train_3(a,b)-r_train_3(a,b+1))>=10)
      rest_features_train(a,21) = rest_features_train(a,21) + 1;
    elseif (rest_train_3(a,b)<rest_train_3(a,b-1) & & rest_train_3(a,b)<rest_train_3(a,b+1) & & (abs(rest_train_3(a,b)-rest_train_3(a,b-1))>=10 || abs(rest_train_3(a,b)-rest_train_3(a,b+1))>=10)
      rest_features_train(a,21) = rest_features_train(a,21) + 1;
    end
  end
end
rest_features_train(a,21) = rest_features_train(a,21) + 1;
end
end
for a = 1:size(rest_train_4,1)
  for b = 2:size(rest_train_4,2)-1
    if( r_train_4(a,b)>r_train_4(a,b-1) && r_train_4(a,b)>r_train_4(a,b+1) && (abs(r_train_4(a,b)-r_train_4(a,b-1))>=10 || abs(r_train_4(a,b)-r_train_4(a,b+1))>=10) )
      rest_features_train(a,22) = rest_features_train(a,22) + 1;
    elseif( rest_train_4(a,b)<rest_train_4(a,b-1) && rest_train_4(a,b)<rest_train_4(a,b+1) && (abs(rest_train_4(a,b)-rest_train_4(a,b-1))>=10 || abs(rest_train_4(a,b)-rest_train_4(a,b+1))>=10) )
      rest_features_train(a,22) = rest_features_train(a,22) + 1;
    end
  end
end
for a = 1:size(rest_train_5,1)
  for b = 2:size(rest_train_5,2)-1
    if( r_train_5(a,b)>r_train_5(a,b-1) && r_train_5(a,b)>r_train_5(a,b+1) && (abs(r_train_5(a,b)-r_train_5(a,b-1))>=10 || abs(r_train_5(a,b)-r_train_5(a,b+1))>=10) )
      rest_features_train(a,23) = rest_features_train(a,23) + 1;
    elseif( rest_train_5(a,b)<rest_train_5(a,b-1) && rest_train_5(a,b)<rest_train_5(a,b+1) && (abs(rest_train_5(a,b)-rest_train_5(a,b-1))>=10 || abs(rest_train_5(a,b)-rest_train_5(a,b+1))>=10) )
      rest_features_train(a,23) = rest_features_train(a,23) + 1;
    end
  end
end
for a = 1:size(rest_train_6,1)
  for b = 2:size(rest_train_6,2)-1
    if( r_train_6(a,b)>r_train_6(a,b-1) && r_train_6(a,b)>r_train_6(a,b+1) && (abs(r_train_6(a,b)-r_train_6(a,b-1))>=10 || abs(r_train_6(a,b)-r_train_6(a,b+1))>=10) )
      rest_features_train(a,24) = rest_features_train(a,24) + 1;
    elseif( rest_train_6(a,b)<rest_train_6(a,b-1) && rest_train_6(a,b)<rest_train_6(a,b+1) && (abs(rest_train_6(a,b)-rest_train_6(a,b-1))>=10 || abs(rest_train_6(a,b)-rest_train_6(a,b+1))>=10) )
      rest_features_train(a,24) = rest_features_train(a,24) + 1;
    end
  end
end
for a = 1:size(rest_test_1,1)
  for b = 2:size(rest_test_1,2)-1
    if( rest_test_1(a,b)>rest_test_1(a,b-1) && rest_test_1(a,b)>rest_test_1(a,b+1) && (abs(rest_test_1(a,b)-rest_test_1(a,b-1))>=10 || abs(rest_test_1(a,b)-rest_test_1(a,b+1))>=10) )
      rest_features_test(a,19) = rest_features_test(a,19) + 1;
    elseif( rest_test_1(a,b)<rest_test_1(a,b-1) && rest_test_1(a,b)<rest_test_1(a,b+1) && (abs(rest_test_1(a,b)-rest_test_1(a,b-1))>=10 || abs(rest_test_1(a,b)-rest_test_1(a,b+1))>=10) )
      rest_features_test(a,19) = rest_features_test(a,19) + 1;
    end
  end
end
for a = 1:size(rest_test_2,1)
  for b = 2:size(rest_test_2,2)-1
    if( rest_test_2(a,b)>rest_test_2(a,b-1) && rest_test_2(a,b)>rest_test_2(a,b+1) && (abs(rest_test_2(a,b)-rest_test_2(a,b-1))>=10 || abs(rest_test_2(a,b)-rest_test_2(a,b+1))>=10) )
      rest_features_test(a,20) = rest_features_test(a,20) + 1;
    elseif( rest_test_2(a,b)<rest_test_2(a,b-1) && rest_test_2(a,b)<rest_test_2(a,b+1) && (abs(rest_test_2(a,b)-rest_test_2(a,b-1))>=10 || abs(rest_test_2(a,b)-rest_test_2(a,b+1))>=10) )
      rest_features_test(a,20) = rest_features_test(a,20) + 1;
    end
  end
end
for a = 1:size(rest_test_3,1)
  for b = 2:size(rest_test_3,2)-1
    if( rest_test_3(a,b)>rest_test_3(a,b-1) && rest_test_3(a,b)>rest_test_3(a,b+1) && (abs(rest_test_3(a,b)-rest_test_3(a,b-1))>=10 || abs(rest_test_3(a,b)-rest_test_3(a,b+1))>=10) )
      rest_features_test(a,21) = rest_features_test(a,21) + 1;
  end
end
end
end
end
end
end
if( rest_test_3(a,b)>rest_test_3(a,b-1) && rest_test_3(a,b)>rest_test_3(a,b+1) && (abs(rest_test_3(a,b)-rest_test_3(a,b-1))>=10 || abs(rest_test_3(a,b)-rest_test_3(a,b+1))>=10) )
    rest_features_test(a,21) = rest_features_test(a,21) + 1;
elseif( rest_test_3(a,b)<rest_test_3(a,b-1) && rest_test_3(a,b)<rest_test_3(a,b+1) && (abs(rest_test_3(a,b)-rest_test_3(a,b-1))>=10 || abs(rest_test_3(a,b)-rest_test_3(a,b+1))>=10) )
    rest_features_test(a,21) = rest_features_test(a,21) + 1;
end

for a = 1:size(rest_test_4,1)
    for b = 2:size(rest_test_4,2)-1
        if( rest_test_4(a,b)>rest_test_4(a,b-1) && rest_test_4(a,b)>rest_test_4(a,b+1) && (abs(rest_test_4(a,b)-rest_test_4(a,b-1))>=10 || abs(rest_test_4(a,b)-rest_test_4(a,b+1))>=10) )
            rest_features_test(a,22) = rest_features_test(a,22) + 1;
        elseif( rest_test_4(a,b)<rest_test_4(a,b-1) && rest_test_4(a,b)<rest_test_4(a,b+1) && (abs(rest_test_4(a,b)-rest_test_4(a,b-1))>=10 || abs(rest_test_4(a,b)-rest_test_4(a,b+1))>=10) )
            rest_features_test(a,22) = rest_features_test(a,22) + 1;
        end
    end
end

for a = 1:size(rest_test_5,1)
    for b = 2:size(rest_test_5,2)-1
        if( rest_test_5(a,b)>rest_test_5(a,b-1) && rest_test_5(a,b)>rest_test_5(a,b+1) && (abs(rest_test_5(a,b)-rest_test_5(a,b-1))>=10 || abs(rest_test_5(a,b)-rest_test_5(a,b+1))>=10) )
            rest_features_test(a,23) = rest_features_test(a,23) + 1;
        elseif( rest_test_5(a,b)<rest_test_5(a,b-1) && rest_test_5(a,b)<rest_test_5(a,b+1) && (abs(rest_test_5(a,b)-rest_test_5(a,b-1))>=10 || abs(rest_test_5(a,b)-rest_test_5(a,b+1))>=10) )
            rest_features_test(a,23) = rest_features_test(a,23) + 1;
        end
    end
end

for a = 1:size(rest_test_6,1)
    for b = 2:size(rest_test_6,2)-1
        if( rest_test_6(a,b)>rest_test_6(a,b-1) && rest_test_6(a,b)>rest_test_6(a,b+1) && (abs(rest_test_6(a,b)-rest_test_6(a,b-1))>=10 || abs(rest_test_6(a,b)-rest_test_6(a,b+1))>=10) )
            rest_features_test(a,24) = rest_features_test(a,24) + 1;
        elseif( rest_test_6(a,b)<rest_test_6(a,b-1) && rest_test_6(a,b)<rest_test_6(a,b+1) && (abs(rest_test_6(a,b)-rest_test_6(a,b-1))>=10 || abs(rest_test_6(a,b)-rest_test_6(a,b+1))>=10) )
            rest_features_test(a,24) = rest_features_test(a,24) + 1;
        end
    end
end

for a = 1:size(e_train_1,1)
    for b = 2:size(e_train_1,2)-1
        if( e_train_1(a,b)>e_train_1(a,b-1) && e_train_1(a,b)>e_train_1(a,b+1) && (abs(e_train_1(a,b)-e_train_1(a,b-1))>=10 || abs(e_train_1(a,b)-e_train_1(a,b+1))>=10) )
            extension_features_train(a,19) = extension_features_train(a,19) + 1;
        elseif( e_train_1(a,b)<e_train_1(a,b-1) && e_train_1(a,b)<e_train_1(a,b+1) && (abs(e_train_1(a,b)-e_train_1(a,b-1))>=10 || abs(e_train_1(a,b)-e_train_1(a,b+1))>=10) )
            extension_features_train(a,19) = extension_features_train(a,19) + 1;
        end
    end
end

for a = 1:size(e_train_2,1)
    for b = 2:size(e_train_2,2)-1
        if( e_train_2(a,b)>e_train_2(a,b-1) && e_train_2(a,b)>e_train_2(a,b+1) && (abs(e_train_2(a,b)-e_train_2(a,b-1))>=10 || abs(e_train_2(a,b)-e_train_2(a,b+1))>=10) )
            extension_features_train(a,20) = extension_features_train(a,20) + 1;
        elseif( e_train_2(a,b)<e_train_2(a,b-1) && e_train_2(a,b)<e_train_2(a,b+1) && (abs(e_train_2(a,b)-e_train_2(a,b-1))>=10 || abs(e_train_2(a,b)-e_train_2(a,b+1))>=10) )
            extension_features_train(a,20) = extension_features_train(a,20) + 1;
        end
    end
end
for a = 1:size(e_train_3,1)
    for b = 2:size(e_train_3,2)-1
        if e_train_3(a,b)>e_train_3(a,b+1) & e_train_3(a,b)>e_train_3(a,b-1) & (abs(e_train_3(a,b)-e_train_3(a,b+1))>=10 || abs(e_train_3(a,b)-e_train_3(a,b-1))>=10)
            extension_features_train(a,21) = extension_features_train(a,21) + 1;
        elseif e_train_3(a,b)<e_train_3(a,b+1) & e_train_3(a,b)<e_train_3(a,b-1) & (abs(e_train_3(a,b)-e_train_3(a,b+1))>=10 || abs(e_train_3(a,b)-e_train_3(a,b-1))>=10)
            extension_features_train(a,21) = extension_features_train(a,21) + 1;
        end
    end
end
for a = 1:size(e_train_4,1)
    for b = 2:size(e_train_4,2)-1
        if e_train_4(a,b)>e_train_4(a,b+1) & e_train_4(a,b)>e_train_4(a,b-1) & (abs(e_train_4(a,b)-e_train_4(a,b+1))>=10 || abs(e_train_4(a,b)-e_train_4(a,b-1))>=10)
            extension_features_train(a,22) = extension_features_train(a,22) + 1;
        elseif e_train_4(a,b)<e_train_4(a,b+1) & e_train_4(a,b)<e_train_4(a,b-1) & (abs(e_train_4(a,b)-e_train_4(a,b+1))>=10 || abs(e_train_4(a,b)-e_train_4(a,b-1))>=10)
            extension_features_train(a,22) = extension_features_train(a,22) + 1;
        end
    end
end
for a = 1:size(e_train_5,1)
    for b = 2:size(e_train_5,2)-1
        if e_train_5(a,b)>e_train_5(a,b+1) & e_train_5(a,b)>e_train_5(a,b-1) & (abs(e_train_5(a,b)-e_train_5(a,b+1))>=10 || abs(e_train_5(a,b)-e_train_5(a,b-1))>=10)
            extension_features_train(a,23) = extension_features_train(a,23) + 1;
        elseif e_train_5(a,b)<e_train_5(a,b+1) & e_train_5(a,b)<e_train_5(a,b-1) & (abs(e_train_5(a,b)-e_train_5(a,b+1))>=10 || abs(e_train_5(a,b)-e_train_5(a,b-1))>=10)
            extension_features_train(a,23) = extension_features_train(a,23) + 1;
        end
    end
end
for a = 1:size(e_train_6,1)
    for b = 2:size(e_train_6,2)-1
        if e_train_6(a,b)>e_train_6(a,b+1) & e_train_6(a,b)>e_train_6(a,b-1) & (abs(e_train_6(a,b)-e_train_6(a,b+1))>=10 || abs(e_train_6(a,b)-e_train_6(a,b-1))>=10)
            extension_features_train(a,24) = extension_features_train(a,24) + 1;
        elseif e_train_6(a,b)<e_train_6(a,b+1) & e_train_6(a,b)<e_train_6(a,b-1) & (abs(e_train_6(a,b)-e_train_6(a,b+1))>=10 || abs(e_train_6(a,b)-e_train_6(a,b-1))>=10)
            extension_features_train(a,24) = extension_features_train(a,24) + 1;
        end
    end
end
for a = 1:size(e_test_1,1)
    for b = 2:size(e_test_1,2)-1
        if e_test_1(a,b)>e_test_1(a,b+1) & e_test_1(a,b)>e_test_1(a,b-1) & (abs(e_test_1(a,b)-e_test_1(a,b+1))>=10 || abs(e_test_1(a,b)-e_test_1(a,b-1))>=10)
            extension_features_test(a,19) = extension_features_test(a,19) + 1;
        elseif e_test_1(a,b)<e_test_1(a,b+1) & e_test_1(a,b)<e_test_1(a,b-1) & (abs(e_test_1(a,b)-e_test_1(a,b+1))>=10 || abs(e_test_1(a,b)-e_test_1(a,b-1))>=10)
            extension_features_test(a,19) = extension_features_test(a,19) + 1;
        end
    end
end
for a = 1:size(e_test_2,1)
    for b = 2:size(e_test_2,2)-1
        if e_test_2(a,b)>e_test_2(a,b+1) & e_test_2(a,b)>e_test_2(a,b-1) & (abs(e_test_2(a,b)-e_test_2(a,b+1))>=10 || abs(e_test_2(a,b)-e_test_2(a,b+1))>=10)
            extension_features_test(a,20) = extension_features_test(a,20) + 1;
        end
    end
end

218
elseif \( e_{test\_2}(a,b)<e_{test\_2}(a,b-1) \) && \( e_{test\_2}(a,b)<e_{test\_2}(a,b+1) \) && (abs(e_{test\_2}(a,b)-e_{test\_2}(a,b-1))>=10 || abs(e_{test\_2}(a,b)-e_{test\_2}(a,b+1))>=10)

extension_features_test(a,20) = extension_features_test(a,20) + 1;
end
end
end

for a = 1:size(e_{test\_3},1)
  for b = 2:size(e_{test\_3},2)-1
    if \( e_{test\_3}(a,b)>e_{test\_3}(a,b-1) \) && \( e_{test\_3}(a,b)>e_{test\_3}(a,b+1) \) && (abs(e_{test\_3}(a,b)-e_{test\_3}(a,b-1))>=10 || abs(e_{test\_3}(a,b)-e_{test\_3}(a,b+1))>=10)
      extension_features_test(a,21) = extension_features_test(a,21) + 1;
    elseif \( e_{test\_3}(a,b)<e_{test\_3}(a,b-1) \) && \( e_{test\_3}(a,b)<e_{test\_3}(a,b+1) \) && (abs(e_{test\_3}(a,b)-e_{test\_3}(a,b-1))>=10 || abs(e_{test\_3}(a,b)-e_{test\_3}(a,b+1))>=10)
      extension_features_test(a,21) = extension_features_test(a,21) + 1;
    end
  end
end

for a = 1:size(e_{test\_4},1)
  for b = 2:size(e_{test\_4},2)-1
    if \( e_{test\_4}(a,b)>e_{test\_4}(a,b-1) \) && \( e_{test\_4}(a,b)>e_{test\_4}(a,b+1) \) && (abs(e_{test\_4}(a,b)-e_{test\_4}(a,b-1))>=10 || abs(e_{test\_4}(a,b)-e_{test\_4}(a,b+1))>=10)
      extension_features_test(a,22) = extension_features_test(a,22) + 1;
    elseif \( e_{test\_4}(a,b)<e_{test\_4}(a,b-1) \) && \( e_{test\_4}(a,b)<e_{test\_4}(a,b+1) \) && (abs(e_{test\_4}(a,b)-e_{test\_4}(a,b-1))>=10 || abs(e_{test\_4}(a,b)-e_{test\_4}(a,b+1))>=10)
      extension_features_test(a,22) = extension_features_test(a,22) + 1;
    end
  end
end

for a = 1:size(e_{test\_5},1)
  for b = 2:size(e_{test\_5},2)-1
    if \( e_{test\_5}(a,b)>e_{test\_5}(a,b-1) \) && \( e_{test\_5}(a,b)>e_{test\_5}(a,b+1) \) && (abs(e_{test\_5}(a,b)-e_{test\_5}(a,b-1))>=10 || abs(e_{test\_5}(a,b)-e_{test\_5}(a,b+1))>=10)
      extension_features_test(a,23) = extension_features_test(a,23) + 1;
    elseif \( e_{test\_5}(a,b)<e_{test\_5}(a,b-1) \) && \( e_{test\_5}(a,b)<e_{test\_5}(a,b+1) \) && (abs(e_{test\_5}(a,b)-e_{test\_5}(a,b-1))>=10 || abs(e_{test\_5}(a,b)-e_{test\_5}(a,b+1))>=10)
      extension_features_test(a,23) = extension_features_test(a,23) + 1;
    end
  end
end

for a = 1:size(e_{test\_6},1)
  for b = 2:size(e_{test\_6},2)-1
    if \( e_{test\_6}(a,b)>e_{test\_6}(a,b-1) \) && \( e_{test\_6}(a,b)>e_{test\_6}(a,b+1) \) && (abs(e_{test\_6}(a,b)-e_{test\_6}(a,b-1))>=10 || abs(e_{test\_6}(a,b)-e_{test\_6}(a,b+1))>=10)
      extension_features_test(a,24) = extension_features_test(a,24) + 1;
    elseif \( e_{test\_6}(a,b)<e_{test\_6}(a,b-1) \) && \( e_{test\_6}(a,b)<e_{test\_6}(a,b+1) \) && (abs(e_{test\_6}(a,b)-e_{test\_6}(a,b-1))>=10 || abs(e_{test\_6}(a,b)-e_{test\_6}(a,b+1))>=10)
      extension_features_test(a,24) = extension_features_test(a,24) + 1;
    end
  end
end

for a = 1:size(f_{train\_1},1)
  for b = 2:size(f_{train\_1},2)-1
    if \( f_{train\_1}(a,b)>f_{train\_1}(a,b-1) \) && \( f_{train\_1}(a,b)>f_{train\_1}(a,b+1) \) && (abs(f_{train\_1}(a,b)-f_{train\_1}(a,b-1))>=10 || abs(f_{train\_1}(a,b)-f_{train\_1}(a,b+1))>=10)
      flexion_features_train(a,19) = flexion_features_train(a,19) + 1;
    elseif \( f_{train\_1}(a,b)<f_{train\_1}(a,b-1) \) && \( f_{train\_1}(a,b)<f_{train\_1}(a,b+1) \) && (abs(f_{train\_1}(a,b)-f_{train\_1}(a,b-1))>=10 || abs(f_{train\_1}(a,b)-f_{train\_1}(a,b+1))>=10)
      flexion_features_train(a,19) = flexion_features_train(a,19) + 1;
    end
  end
end
for a = 1:size(f_{train\_2},1)
for \( b = 2: \text{size}(f_{\text{train}_2,2})-1 \)
\[
\begin{align*}
&\text{if} (f_{\text{train}_2(a,b)}>f_{\text{train}_2(a,b+1)} \&\& (\text{abs}(f_{\text{train}_2(a,b)}-f_{\text{train}_2(a,b+1)})>=10) \\
&\quad \text{flexion_features_train}(a,20) = \text{flexion_features_train}(a,20) + 1; \\
&\text{elseif} (f_{\text{train}_2(a,b)}<f_{\text{train}_2(a,b+1)} \&\& (\text{abs}(f_{\text{train}_2(a,b)}-f_{\text{train}_2(a,b+1)})>=10) \\
&\quad \text{flexion_features_train}(a,20) = \text{flexion_features_train}(a,20) + 1;
\end{align*}
\]
end
end
end

for \( a = 1: \text{size}(f_{\text{train}_3,1}) \)
\[
\begin{align*}
&\quad \text{for} \ b = 2: \text{size}(f_{\text{train}_3,2})-1 \\
&\quad \quad \text{if} (f_{\text{train}_3(a,b)}>f_{\text{train}_3(a,b+1)} \&\& (\text{abs}(f_{\text{train}_3(a,b)}-f_{\text{train}_3(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_train}(a,21) = \text{flexion_features_train}(a,21) + 1; \\
&\quad \quad \text{elseif} (f_{\text{train}_3(a,b)}<f_{\text{train}_3(a,b+1)} \&\& (\text{abs}(f_{\text{train}_3(a,b)}-f_{\text{train}_3(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_train}(a,21) = \text{flexion_features_train}(a,21) + 1;
\end{align*}
\]
end
end
end

for \( a = 1: \text{size}(f_{\text{train}_4,1}) \)
\[
\begin{align*}
&\quad \text{for} \ b = 2: \text{size}(f_{\text{train}_4,2})-1 \\
&\quad \quad \text{if} (f_{\text{train}_4(a,b)}>f_{\text{train}_4(a,b+1)} \&\& (\text{abs}(f_{\text{train}_4(a,b)}-f_{\text{train}_4(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_train}(a,22) = \text{flexion_features_train}(a,22) + 1; \\
&\quad \quad \text{elseif} (f_{\text{train}_4(a,b)}<f_{\text{train}_4(a,b+1)} \&\& (\text{abs}(f_{\text{train}_4(a,b)}-f_{\text{train}_4(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_train}(a,22) = \text{flexion_features_train}(a,22) + 1;
\end{align*}
\]
end
end
end

for \( a = 1: \text{size}(f_{\text{train}_5,1}) \)
\[
\begin{align*}
&\quad \text{for} \ b = 2: \text{size}(f_{\text{train}_5,2})-1 \\
&\quad \quad \text{if} (f_{\text{train}_5(a,b)}>f_{\text{train}_5(a,b+1)} \&\& (\text{abs}(f_{\text{train}_5(a,b)}-f_{\text{train}_5(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_train}(a,23) = \text{flexion_features_train}(a,23) + 1; \\
&\quad \quad \text{elseif} (f_{\text{train}_5(a,b)}<f_{\text{train}_5(a,b+1)} \&\& (\text{abs}(f_{\text{train}_5(a,b)}-f_{\text{train}_5(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_train}(a,23) = \text{flexion_features_train}(a,23) + 1;
\end{align*}
\]
end
end
end

for \( a = 1: \text{size}(f_{\text{test}_1,1}) \)
\[
\begin{align*}
&\quad \text{for} \ b = 2: \text{size}(f_{\text{test}_1,2})-1 \\
&\quad \quad \text{if} (f_{\text{test}_1(a,b)}>f_{\text{test}_1(a,b+1)} \&\& (\text{abs}(f_{\text{test}_1(a,b)}-f_{\text{test}_1(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_test}(a,19) = \text{flexion_features_test}(a,19) + 1; \\
&\quad \quad \text{elseif} (f_{\text{test}_1(a,b)}<f_{\text{test}_1(a,b+1)} \&\& (\text{abs}(f_{\text{test}_1(a,b)}-f_{\text{test}_1(a,b+1)})>=10) \\
&\quad \quad \quad \text{flexion_features_test}(a,19) = \text{flexion_features_test}(a,19) + 1;
\end{align*}
\]
end
end
end

220
end
end
end

for a = 1:size(f_test_2,1)
    for b = 2:size(f_test_2,2)-1
        if (f_test_2(a,b)>f_test_2(a,b-1) && f_test_2(a,b)>f_test_2(a,b+1) && (abs(f_test_2(a,b)-f_test_2(a,b-1))>=10 || abs(f_test_2(a,b)-f_test_2(a,b+1))>=10))
            flexion_features_test(a,20) = flexion_features_test(a,20) + 1;
        elseif (f_test_2(a,b)<f_test_2(a,b-1) && f_test_2(a,b)<f_test_2(a,b+1) && (abs(f_test_2(a,b)-f_test_2(a,b-1))>=10 || abs(f_test_2(a,b)-f_test_2(a,b+1))>=10))
            flexion_features_test(a,20) = flexion_features_test(a,20) + 1;
        end
    end
end

for a = 1:size(f_test_3,1)
    for b = 2:size(f_test_3,2)-1
        if (f_test_3(a,b)>f_test_3(a,b-1) && f_test_3(a,b)>f_test_3(a,b+1) && (abs(f_test_3(a,b)-f_test_3(a,b-1))>=10 || abs(f_test_3(a,b)-f_test_3(a,b+1))>=10))
            flexion_features_test(a,21) = flexion_features_test(a,21) + 1;
        elseif (f_test_3(a,b)<f_test_3(a,b-1) && f_test_3(a,b)<f_test_3(a,b+1) && (abs(f_test_3(a,b)-f_test_3(a,b-1))>=10 || abs(f_test_3(a,b)-f_test_3(a,b+1))>=10))
            flexion_features_test(a,21) = flexion_features_test(a,21) + 1;
        end
    end
end

for a = 1:size(f_test_4,1)
    for b = 2:size(f_test_4,2)-1
        if (f_test_4(a,b)>f_test_4(a,b-1) && f_test_4(a,b)>f_test_4(a,b+1) && (abs(f_test_4(a,b)-f_test_4(a,b-1))>=10 || abs(f_test_4(a,b)-f_test_4(a,b+1))>=10))
            flexion_features_test(a,22) = flexion_features_test(a,22) + 1;
        elseif (f_test_4(a,b)<f_test_4(a,b-1) && f_test_4(a,b)<f_test_4(a,b+1) && (abs(f_test_4(a,b)-f_test_4(a,b-1))>=10 || abs(f_test_4(a,b)-f_test_4(a,b+1))>=10))
            flexion_features_test(a,22) = flexion_features_test(a,22) + 1;
        end
    end
end

for a = 1:size(f_test_5,1)
    for b = 2:size(f_test_5,2)-1
        if (f_test_5(a,b)>f_test_5(a,b-1) && f_test_5(a,b)>f_test_5(a,b+1) && (abs(f_test_5(a,b)-f_test_5(a,b-1))>=10 || abs(f_test_5(a,b)-f_test_5(a,b+1))>=10))
            flexion_features_test(a,23) = flexion_features_test(a,23) + 1;
        elseif (f_test_5(a,b)<f_test_5(a,b-1) && f_test_5(a,b)<f_test_5(a,b+1) && (abs(f_test_5(a,b)-f_test_5(a,b-1))>=10 || abs(f_test_5(a,b)-f_test_5(a,b+1))>=10))
            flexion_features_test(a,23) = flexion_features_test(a,23) + 1;
        end
    end
end

for a = 1:size(f_test_6,1)
    for b = 2:size(f_test_6,2)-1
        if (f_test_6(a,b)>f_test_6(a,b-1) && f_test_6(a,b)>f_test_6(a,b+1) && (abs(f_test_6(a,b)-f_test_6(a,b-1))>=10 || abs(f_test_6(a,b)-f_test_6(a,b+1))>=10))
            flexion_features_test(a,24) = flexion_features_test(a,24) + 1;
        elseif (f_test_6(a,b)<f_test_6(a,b-1) && f_test_6(a,b)<f_test_6(a,b+1) && (abs(f_test_6(a,b)-f_test_6(a,b-1))>=10 || abs(f_test_6(a,b)-f_test_6(a,b+1))>=10))
            flexion_features_test(a,24) = flexion_features_test(a,24) + 1;
        end
    end
end

% Waveform Length
for a = 1:size(g_train_1,1)-1
    for b = 2:size(g_train_1,2)-1
grasp_features_train(a,25) = grasp_features_train(a,25) + abs(g_train_1(a,b) - g_train_1(a,b-1));
end
for a = 1:size(g_train_2,1)-1
  for b = 2:size(g_train_2,2)-1
    grasp_features_train(a,26) = grasp_features_train(a,26) + abs(g_train_2(a,b) - g_train_2(a,b-1));
  end
end
for a = 1:size(g_train_3,1)-1
  for b = 2:size(g_train_3,2)-1
    grasp_features_train(a,27) = grasp_features_train(a,27) + abs(g_train_3(a,b) - g_train_3(a,b-1));
  end
end
for a = 1:size(g_train_4,1)-1
  for b = 2:size(g_train_4,2)-1
    grasp_features_train(a,28) = grasp_features_train(a,28) + abs(g_train_4(a,b) - g_train_4(a,b-1));
  end
end
for a = 1:size(g_train_5,1)-1
  for b = 2:size(g_train_5,2)-1
    grasp_features_train(a,29) = grasp_features_train(a,29) + abs(g_train_5(a,b) - g_train_5(a,b-1));
  end
end
for a = 1:size(g_train_6,1)-1
  for b = 2:size(g_train_6,2)-1
    grasp_features_train(a,30) = grasp_features_train(a,30) + abs(g_train_6(a,b) - g_train_6(a,b-1));
  end
end
for a = 1:size(g_test_1,1)-1
  for b = 2:size(g_test_1,2)-1
    grasp_features_test(a,25) = grasp_features_test(a,25) + abs(g_test_1(a,b) - g_test_1(a,b-1));
  end
end
for a = 1:size(g_test_2,1)-1
  for b = 2:size(g_test_2,2)-1
    grasp_features_test(a,26) = grasp_features_test(a,26) + abs(g_test_2(a,b) - g_test_2(a,b-1));
  end
end
for a = 1:size(g_test_3,1)-1
  for b = 2:size(g_test_3,2)-1
    grasp_features_test(a,27) = grasp_features_test(a,27) + abs(g_test_3(a,b) - g_test_3(a,b-1));
  end
end
for a = 1:size(g_test_4,1)-1
  for b = 2:size(g_test_4,2)-1
    grasp_features_test(a,28) = grasp_features_test(a,28) + abs(g_test_4(a,b) - g_test_4(a,b-1));
  end
end
for a = 1:size(g_test_5,1)-1
  for b = 2:size(g_test_5,2)-1
    grasp_features_test(a,29) = grasp_features_test(a,29) + abs(g_test_5(a,b) - g_test_5(a,b-1));
  end
end
for a = 1:size(g_test_6,1)-1
  for b = 2:size(g_test_6,2)-1
    grasp_features_test(a,30) = grasp_features_test(a,30) + abs(g_test_6(a,b) - g_test_6(a,b-1));
  end
end
for a = 1:size(r_train_1,1)-1
  for b = 2:size(r_train_1,2)-1
    releasing_features_train(a,25) = releasing_features_train(a,25) + abs(r_train_1(a,b) - r_train_1(a,b-1));
  end
end
for a = 1:size(r_train_2,1)-1
    for b = 2:size(r_train_2,2)-1
        releasing_features_train(a,26) = releasing_features_train(a,26) + abs(r_train_2(a,b) - r_train_2(a,b-1));
    end
end
for a = 1:size(r_train_3,1)-1
    for b = 2:size(r_train_3,2)-1
        releasing_features_train(a,27) = releasing_features_train(a,27) + abs(r_train_3(a,b) - r_train_3(a,b-1));
    end
end
for a = 1:size(r_train_4,1)-1
    for b = 2:size(r_train_4,2)-1
        releasing_features_train(a,28) = releasing_features_train(a,28) + abs(r_train_4(a,b) - r_train_4(a,b-1));
    end
end
for a = 1:size(r_train_5,1)-1
    for b = 2:size(r_train_5,2)-1
        releasing_features_train(a,29) = releasing_features_train(a,29) + abs(r_train_5(a,b) - r_train_5(a,b-1));
    end
end
for a = 1:size(r_train_6,1)-1
    for b = 2:size(r_train_6,2)-1
        releasing_features_train(a,30) = releasing_features_train(a,30) + abs(r_train_6(a,b) - r_train_6(a,b-1));
    end
end
for a = 1:size(r_test_1,1)-1
    for b = 2:size(r_test_1,2)-1
        releasing_features_test(a,25) = releasing_features_test(a,25) + abs(r_test_1(a,b) - r_test_1(a,b-1));
    end
end
for a = 1:size(r_test_2,1)-1
    for b = 2:size(r_test_2,2)-1
        releasing_features_test(a,26) = releasing_features_test(a,26) + abs(r_test_2(a,b) - r_test_2(a,b-1));
    end
end
for a = 1:size(r_test_3,1)-1
    for b = 2:size(r_test_3,2)-1
        releasing_features_test(a,27) = releasing_features_test(a,27) + abs(r_test_3(a,b) - r_test_3(a,b-1));
    end
end
for a = 1:size(r_test_4,1)-1
    for b = 2:size(r_test_4,2)-1
        releasing_features_test(a,28) = releasing_features_test(a,28) + abs(r_test_4(a,b) - r_test_4(a,b-1));
    end
end
for a = 1:size(r_test_5,1)-1
    for b = 2:size(r_test_5,2)-1
        releasing_features_test(a,29) = releasing_features_test(a,29) + abs(r_test_5(a,b) - r_test_5(a,b-1));
    end
end
for a = 1:size(r_test_6,1)-1
    for b = 2:size(r_test_6,2)-1
        releasing_features_test(a,30) = releasing_features_test(a,30) + abs(r_test_6(a,b) - r_test_6(a,b-1));
    end
end
for a = 1:size(p_train_1,1)-1
    for b = 2:size(p_train_1,2)-1
        pronation_features_train(a,25) = pronation_features_train(a,25) + abs(p_train_1(a,b) - p_train_1(a,b-1));
    end
end
for a = 1:size(p_train_2,1)-1
    for b = 2:size(p_train_2,2)-1
        pronation_features_train(a,25) = pronation_features_train(a,25) + abs(p_train_2(a,b) - p_train_2(a,b-1));
    end
end
for b = 2:size(p_train_2,2)-1
    pronation_features_train(a,26) = pronation_features_train(a,26) + abs(p_train_2(a,b) - p_train_2(a,b-1));
end
end
for a = 1:size(p_train_3,1)-1
    for b = 2:size(p_train_3,2)-1
        pronation_features_train(a,27) = pronation_features_train(a,27) + abs(p_train_3(a,b) - p_train_3(a,b-1));
    end
end
for a = 1:size(p_train_4,1)-1
    for b = 2:size(p_train_4,2)-1
        pronation_features_train(a,28) = pronation_features_train(a,28) + abs(p_train_4(a,b) - p_train_4(a,b-1));
    end
end
for a = 1:size(p_train_5,1)-1
    for b = 2:size(p_train_5,2)-1
        pronation_features_train(a,29) = pronation_features_train(a,29) + abs(p_train_5(a,b) - p_train_5(a,b-1));
    end
end
for a = 1:size(p_train_6,1)-1
    for b = 2:size(p_train_6,2)-1
        pronation_features_train(a,30) = pronation_features_train(a,30) + abs(p_train_6(a,b) - p_train_6(a,b-1));
    end
end
for a = 1:size(p_test_1,1)-1
    for b = 2:size(p_test_1,2)-1
        pronation_features_test(a,25) = pronation_features_test(a,25) + abs(p_test_1(a,b) - p_test_1(a,b-1));
    end
end
for a = 1:size(p_test_2,1)-1
    for b = 2:size(p_test_2,2)-1
        pronation_features_test(a,26) = pronation_features_test(a,26) + abs(p_test_2(a,b) - p_test_2(a,b-1));
    end
end
for a = 1:size(p_test_3,1)-1
    for b = 2:size(p_test_3,2)-1
        pronation_features_test(a,27) = pronation_features_test(a,27) + abs(p_test_3(a,b) - p_test_3(a,b-1));
    end
end
for a = 1:size(p_test_4,1)-1
    for b = 2:size(p_test_4,2)-1
        pronation_features_test(a,28) = pronation_features_test(a,28) + abs(p_test_4(a,b) - p_test_4(a,b-1));
    end
end
for a = 1:size(p_test_5,1)-1
    for b = 2:size(p_test_5,2)-1
        pronation_features_test(a,29) = pronation_features_test(a,29) + abs(p_test_5(a,b) - p_test_5(a,b-1));
    end
end
for a = 1:size(p_test_6,1)-1
    for b = 2:size(p_test_6,2)-1
        pronation_features_test(a,30) = pronation_features_test(a,30) + abs(p_test_6(a,b) - p_test_6(a,b-1));
    end
end
for a = 1:size(s_train_1,1)-1
    for b = 2:size(s_train_1,2)-1
        supination_features_train(a,25) = supination_features_train(a,25) + abs(s_train_1(a,b) - s_train_1(a,b-1));
    end
end
for a = 1:size(s_train_2,1)-1
    for b = 2:size(s_train_2,2)-1
        supination_features_train(a,26) = supination_features_train(a,26) + abs(s_train_2(a,b) - s_train_2(a,b-1));
    end
end
for a = 1:size(s_train_3,1)-1
    for b = 2:size(s_train_3,2)-1
        supination_features_train(a,27) = supination_features_train(a,27) + abs(s_train_3(a,b) - s_train_3(a,b-1));
    end
end
for a = 1:size(s_train_4,1)-1
    for b = 2:size(s_train_4,2)-1
        supination_features_train(a,28) = supination_features_train(a,28) + abs(s_train_4(a,b) - s_train_4(a,b-1));
    end
end
for a = 1:size(s_train_5,1)-1
    for b = 2:size(s_train_5,2)-1
        supination_features_train(a,29) = supination_features_train(a,29) + abs(s_train_5(a,b) - s_train_5(a,b-1));
    end
end
for a = 1:size(s_train_6,1)-1
    for b = 2:size(s_train_6,2)-1
        supination_features_train(a,30) = supination_features_train(a,30) + abs(s_train_6(a,b) - s_train_6(a,b-1));
    end
end
for a = 1:size(s_test_1,1)-1
    for b = 2:size(s_test_1,2)-1
        supination_features_test(a,25) = supination_features_test(a,25) + abs(s_test_1(a,b) - s_test_1(a,b-1));
    end
end
for a = 1:size(s_test_2,1)-1
    for b = 2:size(s_test_2,2)-1
        supination_features_test(a,26) = supination_features_test(a,26) + abs(s_test_2(a,b) - s_test_2(a,b-1));
    end
end
for a = 1:size(s_test_3,1)-1
    for b = 2:size(s_test_3,2)-1
        supination_features_test(a,27) = supination_features_test(a,27) + abs(s_test_3(a,b) - s_test_3(a,b-1));
    end
end
for a = 1:size(s_test_4,1)-1
    for b = 2:size(s_test_4,2)-1
        supination_features_test(a,28) = supination_features_test(a,28) + abs(s_test_4(a,b) - s_test_4(a,b-1));
    end
end
for a = 1:size(s_test_5,1)-1
    for b = 2:size(s_test_5,2)-1
        supination_features_test(a,29) = supination_features_test(a,29) + abs(s_test_5(a,b) - s_test_5(a,b-1));
    end
end
for a = 1:size(s_test_6,1)-1
    for b = 2:size(s_test_6,2)-1
        supination_features_test(a,30) = supination_features_test(a,30) + abs(s_test_6(a,b) - s_test_6(a,b-1));
    end
end

224
end
end
for a = 1:size(s_train_3,1)-1
    for b = 2:size(s_train_3,2)-1
        supination_features_train(a,27) = supination_features_train(a,27) + abs(s_train_3(a,b) - s_train_3(a,b-1));
    end
end
for a = 1:size(s_train_4,1)-1
    for b = 2:size(s_train_4,2)-1
        supination_features_train(a,28) = supination_features_train(a,28) + abs(s_train_4(a,b) - s_train_4(a,b-1));
    end
end
for a = 1:size(s_train_5,1)-1
    for b = 2:size(s_train_5,2)-1
        supination_features_train(a,29) = supination_features_train(a,29) + abs(s_train_5(a,b) - s_train_5(a,b-1));
    end
end
for a = 1:size(s_train_6,1)-1
    for b = 2:size(s_train_6,2)-1
        supination_features_train(a,30) = supination_features_train(a,30) + abs(s_train_6(a,b) - s_train_6(a,b-1));
    end
end
for a = 1:size(s_test_1,1)-1
    for b = 2:size(s_test_1,2)-1
        supination_features_test(a,25) = supination_features_test(a,25) + abs(s_test_1(a,b) - s_test_1(a,b-1));
    end
end
for a = 1:size(s_test_2,1)-1
    for b = 2:size(s_test_2,2)-1
        supination_features_test(a,26) = supination_features_test(a,26) + abs(s_test_2(a,b) - s_test_2(a,b-1));
    end
end
for a = 1:size(s_test_3,1)-1
    for b = 2:size(s_test_3,2)-1
        supination_features_test(a,27) = supination_features_test(a,27) + abs(s_test_3(a,b) - s_test_3(a,b-1));
    end
end
for a = 1:size(s_test_4,1)-1
    for b = 2:size(s_test_4,2)-1
        supination_features_test(a,28) = supination_features_test(a,28) + abs(s_test_4(a,b) - s_test_4(a,b-1));
    end
end
for a = 1:size(s_test_5,1)-1
    for b = 2:size(s_test_5,2)-1
        supination_features_test(a,29) = supination_features_test(a,29) + abs(s_test_5(a,b) - s_test_5(a,b-1));
    end
end
for a = 1:size(s_test_6,1)-1
    for b = 2:size(s_test_6,2)-1
        supination_features_test(a,30) = supination_features_test(a,30) + abs(s_test_6(a,b) - s_test_6(a,b-1));
    end
end
for a = 1:size(rest_train_1,1)-1
    for b = 2:size(rest_train_1,2)-1
        rest_features_train(a,25) = rest_features_train(a,25) + abs(rest_train_1(a,b) - rest_train_1(a,b-1));
    end
end
for a = 1:size(rest_train_2,1)-1
    for b = 2:size(rest_train_2,2)-1
        rest_features_train(a,26) = rest_features_train(a,26) + abs(rest_train_2(a,b) - rest_train_2(a,b-1));
    end
end
end
for a = 1:size(rest_train_3,1)-1
    for b = 1:size(rest_train_3,2)-1
        rest_features_train(a,27) = rest_features_train(a,27) + abs(rest_train_3(a,b) - rest_train_3(a,b-1));
    end
end
for a = 1:size(rest_train_4,1)-1
    for b = 1:size(rest_train_4,2)-1
        rest_features_train(a,28) = rest_features_train(a,28) + abs(rest_train_4(a,b) - rest_train_4(a,b-1));
    end
end
for a = 1:size(rest_train_5,1)-1
    for b = 1:size(rest_train_5,2)-1
        rest_features_train(a,29) = rest_features_train(a,29) + abs(rest_train_5(a,b) - rest_train_5(a,b-1));
    end
end
for a = 1:size(rest_train_6,1)-1
    for b = 1:size(rest_train_6,2)-1
        rest_features_train(a,30) = rest_features_train(a,30) + abs(rest_train_6(a,b) - rest_train_6(a,b-1));
    end
end
for a = 1:size(rest_test_1,1)-1
    for b = 1:size(rest_test_1,2)-1
        rest_features_test(a,25) = rest_features_test(a,25) + abs(rest_test_1(a,b) - rest_test_1(a,b-1));
    end
end
for a = 1:size(rest_test_2,1)-1
    for b = 1:size(rest_test_2,2)-1
        rest_features_test(a,26) = rest_features_test(a,26) + abs(rest_test_2(a,b) - rest_test_2(a,b-1));
    end
end
for a = 1:size(rest_test_3,1)-1
    for b = 1:size(rest_test_3,2)-1
        rest_features_test(a,27) = rest_features_test(a,27) + abs(rest_test_3(a,b) - rest_test_3(a,b-1));
    end
end
for a = 1:size(rest_test_4,1)-1
    for b = 1:size(rest_test_4,2)-1
        rest_features_test(a,28) = rest_features_test(a,28) + abs(rest_test_4(a,b) - rest_test_4(a,b-1));
    end
end
for a = 1:size(rest_test_5,1)-1
    for b = 1:size(rest_test_5,2)-1
        rest_features_test(a,29) = rest_features_test(a,29) + abs(rest_test_5(a,b) - rest_test_5(a,b-1));
    end
end
for a = 1:size(rest_test_6,1)-1
    for b = 1:size(rest_test_6,2)-1
        rest_features_test(a,30) = rest_features_test(a,30) + abs(rest_test_6(a,b) - rest_test_6(a,b-1));
    end
end
for a = 1:size(e_train_1,1)-1
    for b = 1:size(e_train_1,2)-1
        extension_features_train(a,25) = extension_features_train(a,25) + abs(e_train_1(a,b) - e_train_1(a,b-1));
    end
end
for a = 1:size(e_train_2,1)-1
    for b = 1:size(e_train_2,2)-1
        extension_features_train(a,26) = extension_features_train(a,26) + abs(e_train_2(a,b) - e_train_2(a,b-1));
    end
end
for a = 1:size(e_train_3,1)-1
    for b = 1:size(e_train_3,2)-1
        extension_features_train(a,27) = extension_features_train(a,27) + abs(e_train_3(a,b) - e_train_3(a,b-1));
    end
end
extension_features_train(a,27) = extension_features_train(a,27) + abs(e_train_3(a,b) - e_train_3(a,b-1));
end
end
for a = 1:size(e_train_4,1)-1
    for b = 2:size(e_train_4,2)-1
        extension_features_train(a,28) = extension_features_train(a,28) + abs(e_train_4(a,b) - e_train_4(a,b-1));
    end
end
for a = 1:size(e_train_5,1)-1
    for b = 2:size(e_train_5,2)-1
        extension_features_train(a,29) = extension_features_train(a,29) + abs(e_train_5(a,b) - e_train_5(a,b-1));
    end
end
for a = 1:size(e_train_6,1)-1
    for b = 2:size(e_train_6,2)-1
        extension_features_train(a,30) = extension_features_train(a,30) + abs(e_train_6(a,b) - e_train_6(a,b-1));
    end
end
for a = 1:size(e_test_1,1)-1
    for b = 2:size(e_test_1,2)-1
        extension_features_test(a,25) = extension_features_test(a,25) + abs(e_test_1(a,b) - e_test_1(a,b-1));
    end
end
for a = 1:size(e_test_2,1)-1
    for b = 2:size(e_test_2,2)-1
        extension_features_test(a,26) = extension_features_test(a,26) + abs(e_test_2(a,b) - e_test_2(a,b-1));
    end
end
for a = 1:size(e_test_3,1)-1
    for b = 2:size(e_test_3,2)-1
        extension_features_test(a,27) = extension_features_test(a,27) + abs(e_test_3(a,b) - e_test_3(a,b-1));
    end
end
for a = 1:size(e_test_4,1)-1
    for b = 2:size(e_test_4,2)-1
        extension_features_test(a,28) = extension_features_test(a,28) + abs(e_test_4(a,b) - e_test_4(a,b-1));
    end
end
for a = 1:size(e_test_5,1)-1
    for b = 2:size(e_test_5,2)-1
        extension_features_test(a,29) = extension_features_test(a,29) + abs(e_test_5(a,b) - e_test_5(a,b-1));
    end
end
for a = 1:size(e_test_6,1)-1
    for b = 2:size(e_test_6,2)-1
        extension_features_test(a,30) = extension_features_test(a,30) + abs(e_test_6(a,b) - e_test_6(a,b-1));
    end
end
for a = 1:size(f_train_1,1)-1
    for b = 2:size(f_train_1,2)-1
        flexion_features_train(a,25) = flexion_features_train(a,25) + abs(f_train_1(a,b) - f_train_1(a,b-1));
    end
end
for a = 1:size(f_train_2,1)-1
    for b = 2:size(f_train_2,2)-1
        flexion_features_train(a,26) = flexion_features_train(a,26) + abs(f_train_2(a,b) - f_train_2(a,b-1));
    end
end
for a = 1:size(f_train_3,1)-1
    for b = 2:size(f_train_3,2)-1
        flexion_features_train(a,27) = flexion_features_train(a,27) + abs(f_train_3(a,b) - f_train_3(a,b-1));
    end
end
end
for a = 1:size(f_train_4,1)-1
    for b = 2:size(f_train_4,2)-1
        flexion_features_train(a,28) = flexion_features_train(a,28) + abs(f_train_4(a,b) - f_train_4(a,b-1));
    end
end
for a = 1:size(f_train_5,1)-1
    for b = 2:size(f_train_5,2)-1
        flexion_features_train(a,29) = flexion_features_train(a,29) + abs(f_train_5(a,b) - f_train_5(a,b-1));
    end
end
for a = 1:size(f_train_6,1)-1
    for b = 2:size(f_train_6,2)-1
        flexion_features_train(a,30) = flexion_features_train(a,30) + abs(f_train_6(a,b) - f_train_6(a,b-1));
    end
end
for a = 1:size(f_test_1,1)-1
    for b = 2:size(f_test_1,2)-1
        flexion_features_test(a,25) = flexion_features_test(a,25) + abs(f_test_1(a,b) - f_test_1(a,b-1));
    end
end
for a = 1:size(f_test_2,1)-1
    for b = 2:size(f_test_2,2)-1
        flexion_features_test(a,26) = flexion_features_test(a,26) + abs(f_test_2(a,b) - f_test_2(a,b-1));
    end
end
for a = 1:size(f_test_3,1)-1
    for b = 2:size(f_test_3,2)-1
        flexion_features_test(a,27) = flexion_features_test(a,27) + abs(f_test_3(a,b) - f_test_3(a,b-1));
    end
end
for a = 1:size(f_test_4,1)-1
    for b = 2:size(f_test_4,2)-1
        flexion_features_test(a,28) = flexion_features_test(a,28) + abs(f_test_4(a,b) - f_test_4(a,b-1));
    end
end
for a = 1:size(f_test_5,1)-1
    for b = 2:size(f_test_5,2)-1
        flexion_features_test(a,29) = flexion_features_test(a,29) + abs(f_test_5(a,b) - f_test_5(a,b-1));
    end
end
for a = 1:size(f_test_6,1)-1
    for b = 2:size(f_test_6,2)-1
        flexion_features_test(a,30) = flexion_features_test(a,30) + abs(f_test_6(a,b) - f_test_6(a,b-1));
    end
end

% This section of code uses the time domain features calculated above
% from training data and implements the LDA on the test data. It calculates
% the accuracy using each TD feature independently and as a set.
% "results" holds the classification accuracies using each feature and the
% feature set. The last row corresponds to majority voting.

results = zeros(6,8);

% Calculation of the mean vectors and covariance matrices for mean absolute
% value as the feature
mean_g = mean(grasp_features_train(:,1:6));
mean_r = mean(releasing_features_train(:,1:6));
mean_p = mean(pronation_features_train(:,1:6));
mean_s = mean(supination_features_train(:,1:6));
mean_e = mean(extension_features_train(:,1:6));
mean_f = mean([flexion_features_train(:,1:6)]);
mean_rest = mean([rest_features_train(:,1:6)]);
cov_g = cov([grasp_features_train(:,1:6)]);
cov_r = cov([releasing_features_train(:,1:6)]);
cov_p = cov([pronation_features_train(:,1:6)]);
cov_s = cov([supination_features_train(:,1:6)]);
cov_e = cov([extension_features_train(:,1:6)]);
cov_f = cov([flexion_features_train(:,1:6)]);
cov_rest = cov([rest_features_train(:,1:6)]);

%Testing Rest
correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)
    test = [rest_features_test(a,1:6)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
    prob_rest>prob_f)
        correct_rest = correct_rest + 1;
    else
        wrong_rest = wrong_rest + 1;
    end
end
results(1,1) = correct_rest / (correct_rest + wrong_rest);

%Testing Grasping
correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)
    test = [grasp_features_test(a,1:6)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_g>prob_rest & & prob_g>prob_r & & prob_g>prob_p & & prob_g>prob_s & & prob_g>prob_e & &
    prob_g>prob_f)
        correct_g = correct_g + 1;
    end
end
229
else
    wrong_g = wrong_g + 1;
end
end

results(1,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing

correct_r = 0;
wrong_r = 0;
for a = 1:size(releasing_features_test,1)
    test = [releasing_features_test(a,1:6)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*(pi)^(6/2)*det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*(pi)^(6/2)*det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*(pi)^(6/2)*det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*(pi)^(6/2)*det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*(pi)^(6/2)*det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*(pi)^(6/2)*det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*(pi)^(6/2)*det(cov_f))^(1/2))*exp(-0.5*w_f);

    if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
        correct_r = correct_r + 1;
    else
        wrong_r = wrong_r + 1;
    end
end

results(1,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation

correct_p = 0;
wrong_p = 0;
for a = 1:size(pronation_features_test,1)
    test = [pronation_features_test(a,1:6)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*(pi)^(6/2)*det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*(pi)^(6/2)*det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*(pi)^(6/2)*det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*(pi)^(6/2)*det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*(pi)^(6/2)*det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*(pi)^(6/2)*det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*(pi)^(6/2)*det(cov_f))^(1/2))*exp(-0.5*w_f);

    if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
        correct_p = correct_p + 1;
    else
        wrong_p = wrong_p + 1;
    end
end

results(1,4) = correct_p / (correct_p + wrong_p);

% Testing Supination
correct_s = 0;
wrong_s = 0;
for a = 1:size(supination_features_test,1)
test = [supination_features_test(a,1:6)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
correct_s = correct_s + 1;
else
wrong_s = wrong_s + 1;
end
end
results(1,5) = correct_s / (correct_s + wrong_s);

%Testing Extension

correct_e = 0;
wrong_e = 0;
for a = 1:size(extension_features_test,1)
test = [extension_features_test(a,1:6)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
correct_e = correct_e + 1;
else
wrong_e = wrong_e + 1;
end
end
results(1,6) = correct_e / (correct_e + wrong_e);

%Testing Flexion

correct_f = 0;
wrong_f = 0;
for a = 1:size(flexion_features_test,1)
test = [flexion_features_test(a,1:6)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
correct_f = correct_f + 1;
else
wrong_f = wrong_f + 1;
end
end
results(1,6) = correct_f / (correct_f + wrong_f);
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);

w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);

w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);

w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);

w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);

w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);

w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_f>prob_rest && prob_f>prob_g && prob_f>prob_r && prob_f>prob_p && prob_f>prob_s && prob_f>prob_e)
correct_f = correct_f + 1;
else
  wrong_f = wrong_f + 1;
end
end

results(1,7) = correct_f / (correct_f + wrong_f);
results(1,8) = (correct_g+correct_r+correct_p+correct_s+correct_rest)/(wrong_g+wrong_r+wrong_p+wrong_s+wrong_rest+correct_g+correct_r+correct_p+correct_s+correct_rest);

% Calculation of the mean vectors and covariance matrices for using mean absolute value slope as the features
mean_g = mean([grasp_features_train(:,7:12)]);
mean_r = mean([releasing_features_train(:,7:12)]);
mean_p = mean([pronation_features_train(:,7:12)]);
mean_s = mean([supination_features_train(:,7:12)]);
mean_e = mean([extension_features_train(:,7:12)]);
mean_f = mean([flexion_features_train(:,7:12)]);
mean_rest = mean([rest_features_train(:,7:12)]);
cov_g = cov([grasp_features_train(:,7:12)]);
cov_r = cov([releasing_features_train(:,7:12)]);
cov_p = cov([pronation_features_train(:,7:12)]);
cov_s = cov([supination_features_train(:,7:12)]);
cov_e = cov([extension_features_train(:,7:12)]);
cov_f = cov([flexion_features_train(:,7:12)]);
cov_rest = cov([rest_features_train(:,7:12)]);

% Testing Rest

correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)
  test = [rest_features_test(a,7:12)];
  w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
  prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
  w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
  prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
  w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
  prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
  w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
  prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
  w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
  prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
  w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
  prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
  w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
  prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
end
if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
prob_rest>prob_f)
correct_rest = correct_rest + 1;
else
wrong_rest = wrong_rest + 1;
end
end
results(2,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping
correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)
test = [grasp_features_test(a,7:12)];
w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e &&
prob_g>prob_f)
correct_g = correct_g + 1;
else
wrong_g = wrong_g + 1;
end
end
results(2,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing
correct_r = 0;
wrong_r = 0;
for a = 1:size(releasing_features_test,1)
test = [releasing_features_test(a,7:12)];
w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e &&
prob_r>prob_f)
correct_r = correct_r + 1;
else
wrong_r = wrong_r + 1;
end
end
results(2,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation
correct_p = 0;
wrong_p = 0;
for a = 1:size(pronation_features_test,1)
    test = [pronation_features_test(a,7:12)];
    w_rest = [test-mean_rest] * inv(cov_rest) * [test-mean_rest];
    prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g] * inv(cov_g) * [test-mean_g];
    prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r] * inv(cov_r) * [test-mean_r];
    prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p] * inv(cov_p) * [test-mean_p];
    prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s] * inv(cov_s) * [test-mean_s];
    prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e] * inv(cov_e) * [test-mean_e];
    prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f] * inv(cov_f) * [test-mean_f];
    prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
        correct_p = correct_p + 1;
    else
        wrong_p = wrong_p + 1;
    end
end
results(2,4) = correct_p / (correct_p + wrong_p);

% Testing Supination
correct_s = 0;
wrong_s = 0;
for a = 1:size(supination_features_test,1)
    test = [supination_features_test(a,7:12)];
    w_rest = [test-mean_rest] * inv(cov_rest) * [test-mean_rest];
    prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g] * inv(cov_g) * [test-mean_g];
    prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r] * inv(cov_r) * [test-mean_r];
    prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p] * inv(cov_p) * [test-mean_p];
    prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s] * inv(cov_s) * [test-mean_s];
    prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e] * inv(cov_e) * [test-mean_e];
    prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f] * inv(cov_f) * [test-mean_f];
    prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
        correct_s = correct_s + 1;
    else
        wrong_s = wrong_s + 1;
    end
end
results(2,5) = correct_s / (correct_s + wrong_s);

% Testing Extension
correct_e = 0;
wrong_e = 0;
for a = 1:size(extension_features_test,1)
    test = [extension_features_test(a,7:12)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);
    if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
        correct_e = correct_e + 1;
    else
        wrong_e = wrong_e + 1;
    end
end
results(2,6) = correct_e / (correct_e + wrong_e);
% Testing Flexion
correct_f = 0;
wrong_f = 0;
for a = 1:size(flexion_features_test,1)
    test = [flexion_features_test(a,7:12)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);
    if(prob_f>prob_rest && prob_f>prob_g && prob_f>prob_r && prob_f>prob_p && prob_f>prob_s && prob_f>prob_e)
        correct_f = correct_f + 1;
    else
        wrong_f = wrong_f + 1;
    end
end
results(2,7) = correct_f / (correct_f + wrong_f);
results(2,8) = (correct_g+correct_r+correct_p+correct_s+correct_rest)/(wrong_g+wrong_r+wrong_p+wrong_s+wrong_rest+correct_g+correct_r+correct_p+correct_s+correct_rest);
% Calculation of the mean vectors and covariance matrices for using zero crossings as features
mean_g = mean(grasp_features_train(:,13:18));
mean_r = mean(releasing_features_train(:,13:18));
mean_p = mean(pronation_features_train(:,13:18));
mean_s = mean([supination_features_train(:,13:18)]);
mean_e = mean([extension_features_train(:,13:18)]);
mean_f = mean([flexion_features_train(:,13:18)]);
mean_rest = mean([rest_features_train(:,13:18)]);
cov_g = cov([grasp_features_train(:,13:18)]);
cov_r = cov([releasing_features_train(:,13:18)]);
cov_p = cov([pronation_features_train(:,13:18)]);
cov_s = cov([supination_features_train(:,13:18)]);
cov_e = cov([extension_features_train(:,13:18)]);
cov_f = cov([flexion_features_train(:,13:18)]);
cov_rest = cov([rest_features_train(:,13:18)]);

%Testing Rest
correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)
test = [rest_features_test(a,13:18)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)/(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)/(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)/(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)/(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)/(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)/(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)/(det(cov_f))^(1/2)*exp(-0.5*w_f);
if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
prob_rest>prob_f)
correct_rest = correct_rest + 1;
else
    wrong_rest = wrong_rest + 1;
end
end
results(3,1) = correct_rest / (correct_rest + wrong_rest);

%Testing Grasping

correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)
test = [grasp_features_test(a,13:18)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)/(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)/(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)/(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)/(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)/(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)/(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)/(det(cov_f))^(1/2)*exp(-0.5*w_f);
if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
correct_g = correct_g + 1;
else
    wrong_g = wrong_g + 1;
end
end
results(3,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing

for a = 1:size(releasing_features_test,1)
    test = [releasing_features_test(a,13:18)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(-1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(-1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(-1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(-1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(-1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(-1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(-1/2)*exp(-0.5*w_f);
    if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
correct_r = correct_r + 1;
else
    wrong_r = wrong_r + 1;
end
end
results(3,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation

for a = 1:size(pronation_features_test,1)
    test = [pronation_features_test(a,13:18)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(-1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(-1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(-1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(-1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(-1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(-1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(-1/2)*exp(-0.5*w_f);
    if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
correct_p = correct_p + 1;
else
    wrong_p = wrong_p + 1;
end
end
results(3,4) = correct_p / (correct_p + wrong_p);

%Testing Supination

\[
\text{correct}_s = 0; \\
\text{wrong}_s = 0; \\
\text{for } a = 1:\text{size(supination_features_test,1)} \\
\text{test} = \{\text{supination_features_test(a,13:18)}\};
\]

\[
\text{w_rest} = \{\text{test-mean_rest}\}^{\text{inv(cov_rest)}} \{\text{test-mean_rest}\}; \\
\text{prob_rest} = 1/(2^{(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_rest)}})/2}} \cdot \exp(-0.5 \cdot \text{w_rest}); \\
\text{w_g = test-mean_g}^{\text{inv(cov_g)}} \{\text{test-mean_g}\}; \\
\text{prob_g = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_g)}})/2}} \cdot \exp(-0.5 \cdot w_g); \\
\text{w_r = test-mean_r}^{\text{inv(cov_r)}} \{\text{test-mean_r}\}; \\
\text{prob_r = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_r)}})/2}} \cdot \exp(-0.5 \cdot w_r); \\
\text{w_p = test-mean_p}^{\text{inv(cov_p)}} \{\text{test-mean_p}\}; \\
\text{prob_p = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_p)}})/2}} \cdot \exp(-0.5 \cdot w_p); \\
\text{w_s = test-mean_s}^{\text{inv(cov_s)}} \{\text{test-mean_s}\}; \\
\text{prob_s = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_s)}})/2}} \cdot \exp(-0.5 \cdot w_s); \\
\text{w_e = test-mean_e}^{\text{inv(cov_e)}} \{\text{test-mean_e}\}; \\
\text{prob_e = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_e)}})/2}} \cdot \exp(-0.5 \cdot w_e); \\
\text{w_f = test-mean_f}^{\text{inv(cov_f)}} \{\text{test-mean_f}\}; \\
\text{prob_f = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_f)}})/2}} \cdot \exp(-0.5 \cdot w_f);
\]

\[
\text{if (prob_s > prob_rest && prob_s > prob_g && prob_s > prob_r && prob_s > prob_p && prob_s > prob_e && prob_s > prob_f)} \\
\text{correct_s = correct_s + 1;}
\]

\[
\text{else} \\
\text{wrong_s = wrong_s + 1;}
\]

end

end

results(3,5) = correct_s / (correct_s + wrong_s);

%Testing Extension

\[
\text{correct}_e = 0; \\
\text{wrong}_e = 0; \\
\text{for } a = 1:\text{size(extension_features_test,1)} \\
\text{test} = \{\text{extension_features_test(a,13:18)}\};
\]

\[
\text{w_rest = test-mean_rest}^{\text{inv(cov_rest)}} \{\text{test-mean_rest}\}; \\
\text{prob_rest = 1/(2^{(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_rest)}})/2}} \cdot \exp(-0.5 \cdot \text{w_rest}); \\
\text{w_g = test-mean_g}^{\text{inv(cov_g)}} \{\text{test-mean_g}\}; \\
\text{prob_g = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_g)}})/2}} \cdot \exp(-0.5 \cdot w_g); \\
\text{w_r = test-mean_r}^{\text{inv(cov_r)}} \{\text{test-mean_r}\}; \\
\text{prob_r = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_r)}})/2}} \cdot \exp(-0.5 \cdot w_r); \\
\text{w_p = test-mean_p}^{\text{inv(cov_p)}} \{\text{test-mean_p}\}; \\
\text{prob_p = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_p)}})/2}} \cdot \exp(-0.5 \cdot w_p); \\
\text{w_s = test-mean_s}^{\text{inv(cov_s)}} \{\text{test-mean_s}\}; \\
\text{prob_s = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_s)}})/2}} \cdot \exp(-0.5 \cdot w_s); \\
\text{w_e = test-mean_e}^{\text{inv(cov_e)}} \{\text{test-mean_e}\}; \\
\text{prob_e = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_e)}})/2}} \cdot \exp(-0.5 \cdot w_e); \\
\text{w_f = test-mean_f}^{\text{inv(cov_f)}} \{\text{test-mean_f}\}; \\
\text{prob_f = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_f)}})/2}} \cdot \exp(-0.5 \cdot w_f);
\]

\[
\text{if (prob_e > prob_rest && prob_e > prob_g && prob_e > prob_r && prob_e > prob_p && prob_e > prob_s && prob_e > prob_f)} \\
\text{correct_e = correct_e + 1;}
\]

\[
\text{else} \\
\text{wrong_e = wrong_e + 1;}
\]

end

end

results(3,6) = correct_e / (correct_e + wrong_e);

%Testing Flexion

\[
\text{correct}_f = 0; \\
\text{wrong}_f = 0; \\
\text{for } a = 1:\text{size(flexion_features_test,1)} \\
\text{test} = \{\text{flexion_features_test(a,13:18)}\};
\]

\[
\text{w_rest = test-mean_rest}^{\text{inv(cov_rest)}} \{\text{test-mean_rest}\}; \\
\text{prob_rest = 1/(2^{(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_rest)}})/2}} \cdot \exp(-0.5 \cdot \text{w_rest}); \\
\text{w_g = test-mean_g}^{\text{inv(cov_g)}} \{\text{test-mean_g}\}; \\
\text{prob_g = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_g)}})/2}} \cdot \exp(-0.5 \cdot w_g); \\
\text{w_r = test-mean_r}^{\text{inv(cov_r)}} \{\text{test-mean_r}\}; \\
\text{prob_r = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_r)}})/2}} \cdot \exp(-0.5 \cdot w_r); \\
\text{w_p = test-mean_p}^{\text{inv(cov_p)}} \{\text{test-mean_p}\}; \\
\text{prob_p = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_p)}})/2}} \cdot \exp(-0.5 \cdot w_p); \\
\text{w_s = test-mean_s}^{\text{inv(cov_s)}} \{\text{test-mean_s}\}; \\
\text{prob_s = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_s)}})/2}} \cdot \exp(-0.5 \cdot w_s); \\
\text{w_e = test-mean_e}^{\text{inv(cov_e)}} \{\text{test-mean_e}\}; \\
\text{prob_e = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_e)}})/2}} \cdot \exp(-0.5 \cdot w_e); \\
\text{w_f = test-mean_f}^{\text{inv(cov_f)}} \{\text{test-mean_f}\}; \\
\text{prob_f = 1/(2*\pi)} \cdot (6/2)^{(\sqrt{\text{det(cov_f)}})/2}} \cdot \exp(-0.5 \cdot w_f);
\]

\[
\text{if (prob_f > prob_rest && prob_f > prob_g && prob_f > prob_r && prob_f > prob_p && prob_f > prob_s && prob_f > prob_e)} \\
\text{correct_f = correct_f + 1;}
\]

\[
\text{else} \\
\text{wrong_f = wrong_f + 1;}
\]

end

end

results(3,6) = correct_f / (correct_f + wrong_f);
if (prob_f > prob_rest && prob_f > prob_g && prob_f > prob_r && prob_f > prob_p && prob_f > prob_s && prob_f > prob_e)
    correct_f = correct_f + 1;
else
    wrong_f = wrong_f + 1;
end
end
results(3,7) = correct_f / (correct_f + wrong_f);
results(3,8) = (correct_g + correct_r + correct_p + correct_s + correct_rest) / (wrong_g + wrong_r + wrong_p + wrong_s + wrong_rest + correct_g + correct_r + correct_p + correct_s + correct_rest);

% Calculation of the mean vectors and covariance matrices for using slope
% sign changes as features
mean_g = mean([grasp_features_train(:,19:24)]);
mean_r = mean([releasing_features_train(:,19:24)]);
mean_p = mean([pronation_features_train(:,19:24)]);
mean_s = mean([supination_features_train(:,19:24)]);
mean_e = mean([extension_features_train(:,19:24)]);
mean_f = mean([flexion_features_train(:,19:24)]);
mean_rest = mean([rest_features_train(:,19:24)]);
cov_g = cov([grasp_features_train(:,19:24)]);
cov_r = cov([releasing_features_train(:,19:24)]);
cov_p = cov([pronation_features_train(:,19:24)]);
cov_s = cov([supination_features_train(:,19:24)]);
cov_e = cov([extension_features_train(:,19:24)]);
cov_f = cov([flexion_features_train(:,19:24)]);
cov_rest = cov([rest_features_train(:,19:24)]);

% Testing Rest
correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)
    test = [rest_features_test(a,19:24)];
    w_rest = (test - mean_rest)' * inv(cov_rest) * (test - mean_rest);
    prob_rest = 1 / ((2*pi)^(6/2) * det(cov_rest)^(1/2)) * exp(-0.5 * w_rest);
    w_g = (test - mean_gl)' * inv(cov_g) * (test - mean_gl);
    prob_g = 1 / ((2*pi)^(6/2) * det(cov_g)^(1/2)) * exp(-0.5 * w_g);
    w_r = (test - mean_rl)' * inv(cov_r) * (test - mean_rl);
    prob_r = 1 / ((2*pi)^(6/2) * det(cov_r)^(1/2)) * exp(-0.5 * w_r);
    w_p = (test - mean_pl)' * inv(cov_p) * (test - mean_pl);
    prob_p = 1 / ((2*pi)^(6/2) * det(cov_p)^(1/2)) * exp(-0.5 * w_p);
    w_s = (test - mean_sl)' * inv(cov_s) * (test - mean_sl);
    prob_s = 1 / ((2*pi)^(6/2) * det(cov_s)^(1/2)) * exp(-0.5 * w_s);
    w_e = (test - mean_el)' * inv(cov_e) * (test - mean_el);
    prob_e = 1 / ((2*pi)^(6/2) * det(cov_e)^(1/2)) * exp(-0.5 * w_e);
    if (prob_f > prob_rest && prob_f > prob_g && prob_f > prob_r && prob_f > prob_p && prob_f > prob_s && prob_f > prob_e)
        correct_f = correct_f + 1;
    else
        wrong_f = wrong_f + 1;
    end
end
results(3,7) = correct_f / (correct_f + wrong_f);
results(3,8) = (correct_g + correct_r + correct_p + correct_s + correct_rest) / (wrong_g + wrong_r + wrong_p + wrong_s + wrong_rest + correct_g + correct_r + correct_p + correct_s + correct_rest);
w_f = (test-mean_f)'*inv(cov_f)*[test-mean_f]';
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
prob_rest>prob_f)
correct_rest = correct_rest + 1;
else
wrong_rest = wrong_rest + 1;
end
end
results(4,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping
correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)
test = [grasp_features_test(a,19:24)];
w_rest = (test-mean_rest)'*inv(cov_rest)*[test-mean_rest]';
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = (test-mean_g)'*inv(cov_g)*[test-mean_g]';
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = (test-mean_r)'*inv(cov_r)*[test-mean_r]';
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = (test-mean_p)'*inv(cov_p)*[test-mean_p]';
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = (test-mean_s)'*inv(cov_s)*[test-mean_s]';
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = (test-mean_e)'*inv(cov_e)*[test-mean_e]';
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = (test-mean_f)'*inv(cov_f)*[test-mean_f]';
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e &&
prob_g>prob_f)
correct_g = correct_g + 1;
else
wrong_g = wrong_g + 1;
end
end
results(4,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing
correct_r = 0;
wrong_r = 0;
for a = 1:size(releasing_features_test,1)
test = [releasing_features_test(a,19:24)];
w_rest = (test-mean_rest)'*inv(cov_rest)*[test-mean_rest]';
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = (test-mean_g)'*inv(cov_g)*[test-mean_g]';
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = (test-mean_r)'*inv(cov_r)*[test-mean_r]';
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = (test-mean_p)'*inv(cov_p)*[test-mean_p]';
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = (test-mean_s)'*inv(cov_s)*[test-mean_s]';
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = (test-mean_e)'*inv(cov_e)*[test-mean_e]';
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = (test-mean_f)'*inv(cov_f)*[test-mean_f]';
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e &&
prob_r>prob_f)
correct_r = correct_r + 1;
else
wrong_r = wrong_r + 1;
end
end
results(4,2) = correct_r / (correct_r + wrong_r);
else
    wrong_r = wrong_r + 1;
end
end
results(4,3) = correct_r / (correct_r + wrong_r);
%
Testing Pronation
correct_p = 0;
wrong_p = 0;
for a = 1:size(pronation_features_test,1)
    test = [pronation_features_test(a,19:24)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(-1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(-1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(-1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(-1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(-1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(-1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(-1/2)*exp(-0.5*w_f);
    if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
        correct_p = correct_p + 1;
    else
        wrong_p = wrong_p + 1;
    end
end
results(4,4) = correct_p / (correct_p + wrong_p);
%
Testing Supination
correct_s = 0;
wrong_s = 0;
for a = 1:size(supination_features_test,1)
    test = [supination_features_test(a,19:24)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(-1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(-1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(-1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(-1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(-1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(-1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(-1/2)*exp(-0.5*w_f);
    if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
        correct_s = correct_s + 1;
    else
        wrong_s = wrong_s + 1;
    end
end
results(4,5) = correct_s / (correct_s + wrong_s);
%
Testing Extension
correct_e = 0;
wrong_e = 0;
for a = 1:size(extension_features_test,1)
test = [extension_features_test(a,19:24)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
correct_e = correct_e + 1;
else
wrong_e = wrong_e + 1;
end
end
results(4,6) = correct_e / (correct_e + wrong_e);

% Testing Flexion

correct_f = 0;
wrong_f = 0;
for a = 1:size(flexion_features_test,1)
test = [flexion_features_test(a,19:24)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_f>prob_rest && prob_f>prob_g && prob_f>prob_r && prob_f>prob_p && prob_f>prob_s && prob_f>prob_e)
correct_f = correct_f + 1;
else
wrong_f = wrong_f + 1;
end
end
results(4,7) = correct_f / (correct_f + wrong_f);
results(4,8) =

% Calculation of the mean vectors and covariance matrices for using waveform length as features
mean_g = mean([grasp_features_train(:,25:30)]);
mean_r = mean([releasing_features_train(:,25:30)]);
mean_p = mean([pronation_features_train(:,25:30)]);
mean_s = mean([supination_features_train(:,25:30)]);
mean_e = mean([extension_features_train(:,25:30)]);
mean_f = mean([flexion_features_train(:,25:30)]);
mean_rest = mean([rest_features_train(:,25:30)]);
cov_g = cov([grasp_features_train(:,25:30)]);
cov_r = cov([releasing_features_train(:,25:30)]);
cov_p = cov([pronation_features_train(:,25:30)]);
cov_s = cov([supination_features_train(:,25:30)]);
cov_e = cov([extension_features_train(:,25:30)]);
cov_f = cov([flexion_features_train(:,25:30)]);
cov_rest = cov([rest_features_train(:,25:30)]);

% Testing Rest
correct_rest = 0;
wrong_rest = 0;
for a = 1:size(rest_features_test,1)
    test = [rest_features_test(a,25:30)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

    if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)
        correct_rest = correct_rest + 1;
    else
        wrong_rest = wrong_rest + 1;
    end
end
results(5,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping
correct_g = 0;
wrong_g = 0;
for a = 1:size(grasp_features_test,1)
    test = [grasp_features_test(a,25:30)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/(2*pi)^(6/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/(2*pi)^(6/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/(2*pi)^(6/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/(2*pi)^(6/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/(2*pi)^(6/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/(2*pi)^(6/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/(2*pi)^(6/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

    if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)
        correct_g = correct_g + 1;
    else
        wrong_g = wrong_g + 1;
    end
end
prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_g>prob_rest && prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
    correct_g = correct_g + 1;
else
    wrong_g = wrong_g + 1;
end

end

data_results(5,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing

correct_r = 0;
wrong_r = 0;
for a = 1:size(releasing_features_test,1)
    test = [releasing_features_test(a,25:30)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
        correct_r = correct_r + 1;
    else
        wrong_r = wrong_r + 1;
    end
end

data_results(5,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation

correct_p = 0;
wrong_p = 0;
for a = 1:size(pronation_features_test,1)
    test = [pronation_features_test(a,25:30)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/((2*pi)^(6/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/((2*pi)^(6/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/((2*pi)^(6/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/((2*pi)^(6/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/((2*pi)^(6/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/((2*pi)^(6/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/((2*pi)^(6/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
        correct_p = correct_p + 1;
    else
        wrong_p = wrong_p + 1;
    end
end

data_results(5,4) = correct_p / (correct_p + wrong_p);
end
end
results(5,4) = correct_p / (correct_p + wrong_p);

%Testing Supination
correct_s = 0;
wrong_s = 0;
for a = 1:size(supination_features_test,1)
test = supination_features_test(a,25:30);
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^6*(det(cov_rest))^1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^6*(det(cov_g))^1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^6*(det(cov_r))^1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^6*(det(cov_p))^1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^6*(det(cov_s))^1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^6*(det(cov_e))^1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^6*(det(cov_f))^1/2)*exp(-0.5*w_f);
    if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
correct_s = correct_s + 1;
else
    wrong_s = wrong_s + 1;
end
end
results(5,5) = correct_s / (correct_s + wrong_s);

%Testing Extension
correct_e = 0;
wrong_e = 0;
for a = 1:size(extension_features_test,1)
test = extension_features_test(a,25:30);
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^6*(det(cov_rest))^1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^6*(det(cov_g))^1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^6*(det(cov_r))^1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^6*(det(cov_p))^1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^6*(det(cov_s))^1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^6*(det(cov_e))^1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^6*(det(cov_f))^1/2)*exp(-0.5*w_f);
    if(prob_e>prob_rest && prob_e>prob_g && prob_e>prob_r && prob_e>prob_p && prob_e>prob_s && prob_e>prob_f)
correct_e = correct_e + 1;
else
    wrong_e = wrong_e + 1;
end
end
results(5,6) = correct_e / (correct_e + wrong_e);

%Testing Flexion
correct_f = 0;
wrong_f = 0;
for a = 1:size(flexion_features_test,1)
    test = [flexion_features_test(a,25:30)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/(2*pi)^(30/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/(2*pi)^(30/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/(2*pi)^(30/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/(2*pi)^(30/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/(2*pi)^(30/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
    prob_e = 1/(2*pi)^(30/2)*(det(cov_e))^(1/2)*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
    prob_f = 1/(2*pi)^(30/2)*(det(cov_f))^(1/2)*exp(-0.5*w_f);

    if(prob_f>prob_rest && prob_f>prob_g && prob_f>prob_r && prob_f>prob_p && prob_f>prob_s && prob_f>prob_e)
        correct_f = correct_f + 1;
    else
        wrong_f = wrong_f + 1;
    end
end

results(5,7) = correct_f / (correct_f + wrong_f);
results(5,8) = (correct_g+correct_r+correct_p+correct_s+correct_rest)/(wrong_g+wrong_r+wrong_p+wrong_s+wrong_rest+correct_g+correct_r+correct_p+correct_s+correct_rest);

% Calculation of the mean vectors and covariance matrices for using all of
% the time domain features as a set
mean_g = mean([grasp_features_train]);
mean_r = mean([releasing_features_train]);
mean_p = mean([pronation_features_train]);
mean_s = mean([supination_features_train]);
mean_e = mean([extension_features_train]);
mean_f = mean([flexion_features_train]);
mean_rest = mean([rest_features_train]);
cov_g = cov([grasp_features_train]);
cov_r = cov([releasing_features_train]);
cov_p = cov([pronation_features_train]);
cov_s = cov([supination_features_train]);
cov_e = cov([extension_features_train]);
cov_f = cov([flexion_features_train]);
cov_rest = cov([rest_features_train]);

% Testing Rest
for a = 1:size(rest_features_test,1)
    test = [rest_features_test(a,:)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
    prob_rest = 1/(2*pi)^(30/2)*(det(cov_rest))^(1/2)*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
    prob_g = 1/(2*pi)^(30/2)*(det(cov_g))^(1/2)*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
    prob_r = 1/(2*pi)^(30/2)*(det(cov_r))^(1/2)*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
    prob_p = 1/(2*pi)^(30/2)*(det(cov_p))^(1/2)*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
    prob_s = 1/(2*pi)^(30/2)*(det(cov_s))^(1/2)*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
prob_e = 1/((2*pi)^30/(det(cov_e))^(1/2))*exp(-0.5*w_e);

w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
prob_f = 1/((2*pi)^30/(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
   prob_rest>prob_f)
correct_rest = correct_rest + 1;
else
    wrong_rest = wrong_rest + 1;
end

if(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e &&
   prob_rest>prob_f)
    mv_rest(a) = 0;
elseif(prob_g>prob_r && prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
    mv_rest(a) = 1;
elseif(prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
    mv_rest(a) = 2;
elseif(prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
    mv_rest(a) = 3;
elseif(prob_s>prob_e && prob_s>prob_f)
    mv_rest(a) = 4;
elseif(prob_e>prob_f)
    mv_rest(a) = 5;
else
    mv_rest(a) = 6;
end
end

results(6,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping

correct_g = 0;
wrong_g = 0;

for a = 1:size(grasp_features_test,1)
test = [grasp_features_test(a,:)];

w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest]';
prob_rest = 1/((2*pi)^30/(det(cov_rest))^(1/2))*exp(-0.5*w_rest);

w_g = [test-mean_g]*inv(cov_g)*[test-mean_g]';
prob_g = 1/((2*pi)^30/(det(cov_g))^(1/2))*exp(-0.5*w_g);

w_r = [test-mean_r]*inv(cov_r)*[test-mean_r]';
prob_r = 1/((2*pi)^30/(det(cov_r))^(1/2))*exp(-0.5*w_r);

w_p = [test-mean_p]*inv(cov_p)*[test-mean_p]';
prob_p = 1/((2*pi)^30/(det(cov_p))^(1/2))*exp(-0.5*w_p);

w_s = [test-mean_s]*inv(cov_s)*[test-mean_s]';
prob_s = 1/((2*pi)^30/(det(cov_s))^(1/2))*exp(-0.5*w_s);

w_e = [test-mean_e]*inv(cov_e)*[test-mean_e]';
prob_e = 1/((2*pi)^30/(det(cov_e))^(1/2))*exp(-0.5*w_e);

w_f = [test-mean_f]*inv(cov_f)*[test-mean_f]';
prob_f = 1/((2*pi)^30/(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_g>prob_rest &&
   prob_g>prob_r &&
   prob_g>prob_p &&
   prob_g>prob_s &&
   prob_g>prob_e &&
   prob_g>prob_f)
correct_g = correct_g + 1;
else
    wrong_g = wrong_g + 1;
end

if(prob_g>prob_rest &&
   prob_g>prob_r &&
   prob_g>prob_p &&
   prob_g>prob_s &&
   prob_g>prob_e &&
   prob_g>prob_f)
mv_g(a) = 1;
elseif(prob_rest>prob_r &&
       prob_rest>prob_p &&
       prob_rest>prob_s &&
       prob_rest>prob_e &&
       prob_rest>prob_f)
mv_g(a) = 0;
elseif(prob_r>prob_p &&
       prob_r>prob_s &&
       prob_r>prob_e &&
       prob_r>prob_f)

247
mv_g(a) = 2;
elseif(prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
    mv_g(a) = 3;
elseif(prob_s>prob_e && prob_s>prob_f)
    mv_g(a) = 4;
elseif(prob_e>prob_f)
    mv_g(a) = 5;
else
    mv_g(a) = 6;
end
end
results(6,2) = correct_g / (correct_g + wrong_g);

% Testing Releasing
correct_r = 0;
wrong_r = 0;
for a = 1:size(releasing_features_test,1)
    test = [releasing_features_test(a,:)];
    w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
    prob_rest = 1/(2*pi)^((30/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
    w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
    prob_g = 1/(2*pi)^((30/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
    w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
    prob_r = 1/(2*pi)^((30/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
    w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
    prob_p = 1/(2*pi)^((30/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
    w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
    prob_s = 1/(2*pi)^((30/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
    w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
    prob_e = 1/(2*pi)^((30/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
    w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
    prob_f = 1/(2*pi)^((30/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);
    if(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
        correct_r = correct_r + 1;
    else
        wrong_r = wrong_r + 1;
    end
endif(prob_r>prob_rest && prob_r>prob_g && prob_r>prob_p && prob_r>prob_s && prob_r>prob_e && prob_r>prob_f)
    mv_r(a) = 2;
elseif(prob_rest>prob_g && prob_rest>prob_p && prob_rest>prob_s && prob_rest>prob_e && prob_rest>prob_f)
    mv_r(a) = 0;
elseif(prob_g>prob_p && prob_g>prob_s && prob_g>prob_e && prob_g>prob_f)
    mv_r(a) = 1;
elseif(prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
    mv_r(a) = 3;
elseif(prob_s>prob_e && prob_s>prob_f)
    mv_r(a) = 4;
elseif(prob_e>prob_f)
    mv_r(a) = 5;
else
    mv_r(a) = 6;
end
end
results(6,3) = correct_r / (correct_r + wrong_r);

% Testing Pronation
correct_p = 0;
wrong_p = 0;
for a = 1:size(pronation_features_test,1)
    test = [pronation_features_test(a,:)];
w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(30/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(30/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(30/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/(2*pi)^(30/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/(2*pi)^(30/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/(2*pi)^(30/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
prob_f = 1/(2*pi)^(30/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
correct_p = correct_p + 1;
else
wrong_p = wrong_p + 1;
end

if(prob_p>prob_rest && prob_p>prob_g && prob_p>prob_r && prob_p>prob_s && prob_p>prob_e && prob_p>prob_f)
mv_p(a) = 3;
elseif(prob_rest>prob_g && prob_rest>prob_r && prob_rest>prob_s && prob_rest>prob_p && prob_rest>prob_e && prob_rest>prob_f)
mv_p(a) = 0;
elseif(prob_g>prob_r && prob_g>prob_s && prob_g>prob_e && prob_g>prob_p)
mv_p(a) = 1;
elseif(prob_r>prob_s && prob_r>prob_e && prob_r>prob_p)
mv_p(a) = 2;
elseif(prob_s>prob_e)
mv_p(a) = 4;
elseif(prob_e>prob_f)
mv_p(a) = 5;
else
mv_p(a) = 6;
end

results(6,4) = correct_p / (correct_p + wrong_p);

% Testing Supination

correct_s = 0;
wrong_s = 0;
for a = 1:size(supination_features_test,1)
test = [supination_features_test(a,:)];
w_rest = [test-mean_rest]*inv(cov_rest)*[test-mean_rest];
prob_rest = 1/(2*pi)^(30/2)*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
w_g = [test-mean_g]*inv(cov_g)*[test-mean_g];
prob_g = 1/(2*pi)^(30/2)*(det(cov_g))^(1/2))*exp(-0.5*w_g);
w_r = [test-mean_r]*inv(cov_r)*[test-mean_r];
prob_r = 1/(2*pi)^(30/2)*(det(cov_r))^(1/2))*exp(-0.5*w_r);
w_p = [test-mean_p]*inv(cov_p)*[test-mean_p];
prob_p = 1/(2*pi)^(30/2)*(det(cov_p))^(1/2))*exp(-0.5*w_p);
w_s = [test-mean_s]*inv(cov_s)*[test-mean_s];
prob_s = 1/(2*pi)^(30/2)*(det(cov_s))^(1/2))*exp(-0.5*w_s);
w_e = [test-mean_e]*inv(cov_e)*[test-mean_e];
prob_e = 1/(2*pi)^(30/2)*(det(cov_e))^(1/2))*exp(-0.5*w_e);
w_f = [test-mean_f]*inv(cov_f)*[test-mean_f];
prob_f = 1/(2*pi)^(30/2)*(det(cov_f))^(1/2))*exp(-0.5*w_f);

if(prob_s>prob_rest && prob_s>prob_g && prob_s>prob_r && prob_s>prob_p && prob_s>prob_e && prob_s>prob_f)
correct_s = correct_s + 1;
else
wrong_s = wrong_s + 1;
end
\[
\text{wrong}_s = \text{wrong}_s + 1;
\]
\text{end}

\text{if}(\text{prob}_s > \text{prob}_\text{rest} \& \& \text{prob}_s > \text{prob}_g \& \& \text{prob}_s > \text{prob}_r \& \& \text{prob}_s > \text{prob}_p \& \& \text{prob}_s > \text{prob}_e \& \& \text{prob}_s > \text{prob}_f) \\text{mv}_s(a) = 4;\text{elseif}(\text{prob}_\text{rest} > \text{prob}_g \& \& \text{prob}_\text{rest} > \text{prob}_r \& \& \text{prob}_\text{rest} > \text{prob}_p \& \& \text{prob}_\text{rest} > \text{prob}_e \& \& \text{prob}_\text{rest} > \text{prob}_f) \\text{mv}_s(a) = 0;\text{elseif}(\text{prob}_g > \text{prob}_r \& \& \text{prob}_g > \text{prob}_p \& \& \text{prob}_g > \text{prob}_e \& \& \text{prob}_g > \text{prob}_f) \\text{mv}_s(a) = 1;\text{elseif}(\text{prob}_r > \text{prob}_p \& \& \text{prob}_r > \text{prob}_e \& \& \text{prob}_r > \text{prob}_f) \\text{mv}_s(a) = 2;\text{elseif}(\text{prob}_p > \text{prob}_e \& \& \text{prob}_p > \text{prob}_f) \\text{mv}_s(a) = 3;\text{elseif}(\text{prob}_e > \text{prob}_f) \\text{mv}_s(a) = 5;\text{else} \text{mv}_s(a) = 6;\text{end}\text{end}
\]
\text{results}(6,5) = \text{correct}_s / (\text{correct}_s + \text{wrong}_s);
\%
\text{Testing Extension}
\text{correct}_e = 0;
\text{wrong}_e = 0;
\text{for a = 1:size(extension_features_test,1)}
\text{test} = [\text{extension_features_test}(a,:)];
\text{w_rest} = [(\text{test-mean_rest})*inv(cov_rest)*[\text{test-mean_rest}]';
\text{prob_rest} = 1/(2*\pi)^{(30/2)}*(det(cov_rest))^(1/2))*exp(-0.5*w_rest);
\text{w}_g = [(\text{test-mean_g})*inv(cov_g)*[\text{test-mean_g}]';
\text{prob}_g = 1/(2*\pi)^{(30/2)}*(det(cov_g))^(1/2))*exp(-0.5*w_g);
\text{w}_r = [(\text{test-mean_r})*inv(cov_r)*[\text{test-mean_r}]';
\text{prob}_r = 1/(2*\pi)^{(30/2)}*(det(cov_r))^(1/2))*exp(-0.5*w_r);
\text{w}_p = [(\text{test-mean_p})*inv(cov_p)*[\text{test-mean_p}]';
\text{prob}_p = 1/(2*\pi)^{(30/2)}*(det(cov_p))^(1/2))*exp(-0.5*w_p);
\text{w}_s = [(\text{test-mean_s})*inv(cov_s)*[\text{test-mean_s}]';
\text{prob}_s = 1/(2*\pi)^{(30/2)}*(det(cov_s))^(1/2))*exp(-0.5*w_s);
\text{w}_e = [(\text{test-mean_e})*inv(cov_e)*[\text{test-mean_e}]';
\text{prob}_e = 1/(2*\pi)^{(30/2)}*(det(cov_e))^(1/2))*exp(-0.5*w_e);
\text{w}_f = [(\text{test-mean_f})*inv(cov_f)*[\text{test-mean_f}]';
\text{prob}_f = 1/(2*\pi)^{(30/2)}*(det(cov_f))^(1/2))*exp(-0.5*w_f);
\text{if}(\text{prob}_e > \text{prob}_\text{rest} \& \& \text{prob}_e > \text{prob}_g \& \& \text{prob}_e > \text{prob}_r \& \& \text{prob}_e > \text{prob}_p \& \& \text{prob}_e > \text{prob}_s \& \& \text{prob}_e > \text{prob}_f) \\text{correct}_e = \text{correct}_e + 1;\text{else} \\text{wrong}_e = \text{wrong}_e + 1;\text{end}
\text{if}(\text{prob}_e > \text{prob}_\text{rest} \& \& \text{prob}_e > \text{prob}_g \& \& \text{prob}_e > \text{prob}_r \& \& \text{prob}_e > \text{prob}_p \& \& \text{prob}_e > \text{prob}_s \& \& \text{prob}_e > \text{prob}_f) \\text{mv}_e(a) = 5;\text{elseif}(\text{prob}_\text{rest} > \text{prob}_g \& \& \text{prob}_\text{rest} > \text{prob}_r \& \& \text{prob}_\text{rest} > \text{prob}_p \& \& \text{prob}_\text{rest} > \text{prob}_s \& \& \text{prob}_\text{rest} > \text{prob}_f) \\text{mv}_e(a) = 0;\text{elseif}(\text{prob}_g > \text{prob}_r \& \& \text{prob}_g > \text{prob}_p \& \& \text{prob}_g > \text{prob}_s \& \& \text{prob}_g > \text{prob}_f) \\text{mv}_e(a) = 1;\text{elseif}(\text{prob}_r > \text{prob}_p \& \& \text{prob}_r > \text{prob}_s \& \& \text{prob}_r > \text{prob}_f) \\text{mv}_e(a) = 2;\text{elseif}(\text{prob}_p > \text{prob}_s \& \& \text{prob}_p > \text{prob}_f) \\text{mv}_e(a) = 3;\text{elseif}(\text{prob}_p > \text{prob}_f) \\text{mv}_e(a) = 4;\text{else} \text{mv}_e(a) = 6;\text{end}
% Testing Flexion

\[
\text{correct}_f = 0; \\
\text{wrong}_f = 0;
\]

for \( a = 1: \text{size(flexion_features_test,1)} \)
\[
\text{test} = \text{flexion_features_test}(a,:);
\]
\[
\w_{\text{rest}} = \frac{\text{test} - \text{mean_rest}}{\text{inv(cov_rest)} * (\text{test} - \text{mean_rest})'};
\]
\[
\text{prob}_{\text{rest}} = \frac{1}{((2\pi)^{30/2} * (\text{det(cov_rest)})^{1/2})} * \exp(-0.5 * \w_{\text{rest}});
\]
\[
\w_g = \frac{\text{test} - \text{mean_g}}{\text{inv(cov_g)} * (\text{test} - \text{mean_g})'};
\]
\[
\text{prob}_g = \frac{1}{((2\pi)^{30/2} * (\text{det(cov_g)})^{1/2})} * \exp(-0.5 * \w_g);
\]
\[
\w_r = \frac{\text{test} - \text{mean_r}}{\text{inv(cov_r)} * (\text{test} - \text{mean_r})'};
\]
\[
\text{prob}_r = \frac{1}{((2\pi)^{30/2} * (\text{det(cov_r)})^{1/2})} * \exp(-0.5 * \w_r);
\]
\[
\w_p = \frac{\text{test} - \text{mean_p}}{\text{inv(cov_p)} * (\text{test} - \text{mean_p})'};
\]
\[
\text{prob}_p = \frac{1}{((2\pi)^{30/2} * (\text{det(cov_p)})^{1/2})} * \exp(-0.5 * \w_p);
\]
\[
\w_s = \frac{\text{test} - \text{mean_s}}{\text{inv(cov_s)} * (\text{test} - \text{mean_s})'};
\]
\[
\text{prob}_s = \frac{1}{((2\pi)^{30/2} * (\text{det(cov_s)})^{1/2})} * \exp(-0.5 * \w_s);
\]
\[
\w_e = \frac{\text{test} - \text{mean_e}}{\text{inv(cov_e)} * (\text{test} - \text{mean_e})'};
\]
\[
\text{prob}_e = \frac{1}{((2\pi)^{30/2} * (\text{det(cov_e)})^{1/2})} * \exp(-0.5 * \w_e);
\]
\[
\w_f = \frac{\text{test} - \text{mean_f}}{\text{inv(cov_f)} * (\text{test} - \text{mean_f})'};
\]
\[
\text{prob}_f = \frac{1}{((2\pi)^{30/2} * (\text{det(cov_f)})^{1/2})} * \exp(-0.5 * \w_f);
\]

if (\text{prob}_f > \text{prob}_g && \text{prob}_f > \text{prob}_r && \text{prob}_f > \text{prob}_p && \text{prob}_f > \text{prob}_s && \text{prob}_f > \text{prob}_e)
\text{correct}_f = \text{correct}_f + 1;
\] else
\text{wrong}_f = \text{wrong}_f + 1;
\end{if}

end

if (\text{prob}_f > \text{prob}_g && \text{prob}_f > \text{prob}_p && \text{prob}_f > \text{prob}_s && \text{prob}_f > \text{prob}_e)  \\
\text{mv}_f(a) = 6;
\] elseif (\text{prob}_g > \text{prob}_r && \text{prob}_g > \text{prob}_p && \text{prob}_g > \text{prob}_s && \text{prob}_g > \text{prob}_e)  \\
\text{mv}_f(a) = 1;
\] elseif (\text{prob}_r > \text{prob}_p && \text{prob}_r > \text{prob}_s && \text{prob}_r > \text{prob}_e)  \\
\text{mv}_f(a) = 2;
\] elseif (\text{prob}_p > \text{prob}_s && \text{prob}_p > \text{prob}_e)  \\
\text{mv}_f(a) = 3;
\] elseif (\text{prob}_s > \text{prob}_e)  \\
\text{mv}_f(a) = 4;
\] else
\text{mv}_f(a) = 6;
\end{if}

end

\[\text{results}(6,7) = \frac{\text{correct}_f}{(\text{correct}_f + \text{wrong}_f)};\]
\[\text{results}(6,8) = \frac{(\text{correct}_g + \text{correct}_r + \text{correct}_p + \text{correct}_s + \text{correct}_\text{rest})}{(\text{correct}_g + \text{correct}_r + \text{correct}_p + \text{correct}_s + \text{correct}_\text{rest})};\]

% Majority Voting
% This section performs majority voting on the classification of the TD set
% features. The windows are 50ms and a total of 5 windows (250ms) is
% considered in the majority vote. In the 5 windows whichever class was
% classified the most times, results in that class being chosen.

% Testing Rest
\text{correct}_\text{rest} = 0; \\
\text{wrong}_\text{rest} = 0; \\
\text{last_mv} = 0; \\
for \( a = 1: \text{length(mv_rest)} \)
rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;

if (mv_rest(a) == 0)
    rest_votes = rest_votes + 1;
else if (mv_rest(a) == 1)
    g_votes = g_votes + 1;
else if (mv_rest(a) == 2)
    r_votes = r_votes + 1;
else if (mv_rest(a) == 3)
    p_votes = p_votes + 1;
else if (mv_rest(a) == 4)
    s_votes = s_votes + 1;
else if (mv_rest(a) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if (mv_rest(a-1) == 0)
    rest_votes = rest_votes + 1;
else if (mv_rest(a-1) == 1)
    g_votes = g_votes + 1;
else if (mv_rest(a-1) == 2)
    r_votes = r_votes + 1;
else if (mv_rest(a-1) == 3)
    p_votes = p_votes + 1;
else if (mv_rest(a-1) == 4)
    s_votes = s_votes + 1;
else if (mv_rest(a-1) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if (mv_rest(a-2) == 0)
    rest_votes = rest_votes + 1;
else if (mv_rest(a-2) == 1)
    g_votes = g_votes + 1;
else if (mv_rest(a-2) == 2)
    r_votes = r_votes + 1;
else if (mv_rest(a-2) == 3)
    p_votes = p_votes + 1;
else if (mv_rest(a-2) == 4)
    s_votes = s_votes + 1;
else if (mv_rest(a-2) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if (mv_rest(a-3) == 0)
    rest_votes = rest_votes + 1;
else if (mv_rest(a-3) == 1)
    g_votes = g_votes + 1;
else if (mv_rest(a-3) == 2)
    r_votes = r_votes + 1;
else if (mv_rest(a-3) == 3)
    p_votes = p_votes + 1;
else if (mv_rest(a-3) == 4)
s_votes = s_votes + 1;
elseif(mv_rest(a-3) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
endif
if(mv_rest(a-4) == 0)
rest_votes = rest_votes + 1;
elseif(mv_rest(a-4) == 1)
g_votes = g_votes + 1;
elseif(mv_rest(a-4) == 2)
r_votes = r_votes + 1;
elseif(mv_rest(a-4) == 3)
p_votes = p_votes + 1;
elseif(mv_rest(a-4) == 4)
s_votes = s_votes + 1;
elseif(mv_rest(a-4) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
endif
if(rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes && rest_votes>f_votes)
correct_rest = correct_rest + 1;
last_mv = 0;
elseif(g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes && g_votes>f_votes)
wrong_rest = wrong_rest + 1;
last_mv = 1;
elseif(r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes && r_votes>f_votes)
wrong_rest = wrong_rest + 1;
last_mv = 2;
elseif(p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes && p_votes>f_votes)
wrong_rest = wrong_rest + 1;
last_mv = 3;
elseif(s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes && s_votes>f_votes)
wrong_rest = wrong_rest + 1;
last_mv = 4;
elseif(e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes && e_votes>f_votes)
wrong_rest = wrong_rest + 1;
last_mv = 5;
elseif(f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>s_votes && f_votes>e_votes)
wrong_rest = wrong_rest + 1;
elseif(last_mv == 0)
correct_rest = correct_rest + 1;
last_mv = 0;
else
wrong_rest = wrong_rest + 1;
endif
results(7,1) = correct_rest / (correct_rest + wrong_rest);

% Testing Grasping
correct_g = 0;
wrong_g = 0;
last_mv = 0;
for a = 5:length(mv_g)
    rest_votes = 0;
    g_votes = 0;
    r_votes = 0;
    p_votes = 0;
    s_votes = 0;
    e_votes = 0;
    f_votes = 0;

    if(mv_g(a) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_g(a) == 1)
        g_votes = g_votes + 1;
    elseif(mv_g(a) == 2)
        r_votes = r_votes + 1;
    elseif(mv_g(a) == 3)
        p_votes = p_votes + 1;
    elseif(mv_g(a) == 4)
        s_votes = s_votes + 1;
    elseif(mv_g(a) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end

    if(mv_g(a-1) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_g(a-1) == 1)
        g_votes = g_votes + 1;
    elseif(mv_g(a-1) == 2)
        r_votes = r_votes + 1;
    elseif(mv_g(a-1) == 3)
        p_votes = p_votes + 1;
    elseif(mv_g(a-1) == 4)
        s_votes = s_votes + 1;
    elseif(mv_g(a-1) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end

    if(mv_g(a-2) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_g(a-2) == 1)
        g_votes = g_votes + 1;
    elseif(mv_g(a-2) == 2)
        r_votes = r_votes + 1;
    elseif(mv_g(a-2) == 3)
        p_votes = p_votes + 1;
    elseif(mv_g(a-2) == 4)
        s_votes = s_votes + 1;
    elseif(mv_g(a-2) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end

    if(mv_g(a-3) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_g(a-3) == 1)
        g_votes = g_votes + 1;
    elseif(mv_g(a-3) == 2)
        r_votes = r_votes + 1;
    elseif(mv_g(a-3) == 3)
p_votes = p_votes + 1;
elseif(mv_g(a-3) == 4)
s_votes = s_votes + 1;
elseif(mv_g(a-3) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end
if(mv_g(a-4) == 0)
rest_votes = rest_votes + 1;
elseif(mv_g(a-4) == 1)
g_votes = g_votes + 1;
elseif(mv_g(a-4) == 2)
r_votes = r_votes + 1;
elseif(mv_g(a-4) == 3)
p_votes = p_votes + 1;
elseif(mv_g(a-4) == 4)
s_votes = s_votes + 1;
elseif(mv_g(a-4) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end
if(rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes && rest_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 0;
elseif(g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes && g_votes>f_votes)
    correct_g = correct_g + 1;
    last_mv = 1;
elseif(r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes && r_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 2;
elseif(p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>p_votes && p_votes>e_votes && p_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 3;
elseif(s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes && s_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 4;
elseif(e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes && e_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 5;
elseif(f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>p_votes && f_votes>s_votes && f_votes>e_votes && f_votes>f_votes)
    wrong_g = wrong_g + 1;
    last_mv = 6;
elseif(last_mv == 1)
correct_g = correct_g + 1;
    last_mv = 0;
else
wrong_g = wrong_g + 1;
end
results(7,2) = correct_g / (correct_g + wrong_g);
% Testing Releasing
correct_r = 0;
wrong_r = 0;
last_mv = 0;
for a = 5:length(mv_r)
    rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;
    if(mv_r(a) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_r(a) == 1)
        g_votes = g_votes + 1;
    elseif(mv_r(a) == 2)
        r_votes = r_votes + 1;
    elseif(mv_r(a) == 3)
        p_votes = p_votes + 1;
    elseif(mv_r(a) == 4)
        s_votes = s_votes + 1;
    elseif(mv_r(a) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
    if(mv_r(a-1) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_r(a-1) == 1)
        g_votes = g_votes + 1;
    elseif(mv_r(a-1) == 2)
        r_votes = r_votes + 1;
    elseif(mv_r(a-1) == 3)
        p_votes = p_votes + 1;
    elseif(mv_r(a-1) == 4)
        s_votes = s_votes + 1;
    elseif(mv_r(a-1) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
    if(mv_r(a-2) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_r(a-2) == 1)
        g_votes = g_votes + 1;
    elseif(mv_r(a-2) == 2)
        r_votes = r_votes + 1;
    elseif(mv_r(a-2) == 3)
        p_votes = p_votes + 1;
    elseif(mv_r(a-2) == 4)
        s_votes = s_votes + 1;
    elseif(mv_r(a-2) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
    if(mv_r(a-3) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_r(a-3) == 1)
        g_votes = g_votes + 1;
    elseif(mv_r(a-3) == 2)
r_votes = r_votes + 1;
elseif(mv_r(a-3) == 3)
p_votes = p_votes + 1;
elseif(mv_r(a-3) == 4)
s_votes = s_votes + 1;
elseif(mv_r(a-3) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end
if(mv_r(a-4) == 0)
rest_votes = rest_votes + 1;
elseif(mv_r(a-4) == 1)
g_votes = g_votes + 1;
elseif(mv_r(a-4) == 2)
r_votes = r_votes + 1;
elseif(mv_r(a-4) == 3)
p_votes = p_votes + 1;
elseif(mv_r(a-4) == 4)
s_votes = s_votes + 1;
elseif(mv_r(a-4) == 5)
e_votes = e_votes + 1;
else
f_votes = f_votes + 1;
end
if(rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes && rest_votes>f_votes)
wrong_r = wrong_r + 1;
last_mv = 0;
elseif(g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>f_votes)
wrong_r = wrong_r + 1;
last_mv = 1;
elseif(r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes && r_votes>f_votes)
correct_r = correct_r + 1;
last_mv = 2;
elseif(p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes && p_votes>f_votes)
wrong_r = wrong_r + 1;
last_mv = 3;
elseif(s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes && s_votes>f_votes)
wrong_r = wrong_r + 1;
last_mv = 4;
elseif(e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes && e_votes>f_votes)
wrong_r = wrong_r + 1;
last_mv = 5;
elseif(f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>p_votes && f_votes>s_votes && f_votes>e_votes)
wrong_r = wrong_r + 1;
last_mv = 6;
elseif(last_mv == 2)
correct_r = correct_r + 1;
last_mv = 0;
else
wrong_r = wrong_r + 1;
end
end
results(7,3) = correct_r / (correct_r + wrong_r);
% Testing Pronation
correct_p = 0;
wrong_p = 0;
last_mv = 0;
for a = 5:length(mv_p)
    rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;
    if(mv_p(a) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_p(a) == 1)
        g_votes = g_votes + 1;
    elseif(mv_p(a) == 2)
        r_votes = r_votes + 1;
    elseif(mv_p(a) == 3)
        p_votes = p_votes + 1;
    elseif(mv_p(a) == 4)
        s_votes = s_votes + 1;
    elseif(mv_p(a) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
    if(mv_p(a-1) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_p(a-1) == 1)
        g_votes = g_votes + 1;
    elseif(mv_p(a-1) == 2)
        r_votes = r_votes + 1;
    elseif(mv_p(a-1) == 3)
        p_votes = p_votes + 1;
    elseif(mv_p(a-1) == 4)
        s_votes = s_votes + 1;
    elseif(mv_p(a-1) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
    if(mv_p(a-2) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_p(a-2) == 1)
        g_votes = g_votes + 1;
    elseif(mv_p(a-2) == 2)
        r_votes = r_votes + 1;
    elseif(mv_p(a-2) == 3)
        p_votes = p_votes + 1;
    elseif(mv_p(a-2) == 4)
        s_votes = s_votes + 1;
    elseif(mv_p(a-2) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
    if(mv_p(a-3) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_p(a-3) == 1)
        g_votes = g_votes + 1;
    elseif(mv_p(a-3) == 2)
        r_votes = r_votes + 1;
    elseif(mv_p(a-3) == 3)
        p_votes = p_votes + 1;
    elseif(mv_p(a-3) == 4)
        s_votes = s_votes + 1;
    elseif(mv_p(a-3) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
end
258
g_votes = g_votes + 1;
elseif (mv_p(a-3) == 2)
    r_votes = r_votes + 1;
elseif (mv_p(a-3) == 3)
    p_votes = p_votes + 1;
elseif (mv_p(a-3) == 4)
    s_votes = s_votes + 1;
elseif (mv_p(a-3) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if (mv_p(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif (mv_p(a-4) == 1)
    g_votes = g_votes + 1;
elseif (mv_p(a-4) == 2)
    r_votes = r_votes + 1;
elseif (mv_p(a-4) == 3)
    p_votes = p_votes + 1;
elseif (mv_p(a-4) == 4)
    s_votes = s_votes + 1;
elseif (mv_p(a-4) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if (rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes && rest_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 0;
elseif (g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes && g_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 1;
elseif (r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes && r_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 2;
elseif (p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes && p_votes>f_votes)
    correct_p = correct_p + 1;
    last_mv = 3;
elseif (s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes && s_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 4;
elseif (e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>p_votes && e_votes>s_votes && e_votes>f_votes)
    wrong_p = wrong_p + 1;
    last_mv = 5;
elseif (f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>p_votes && f_votes>s_votes && f_votes>e_votes)
    wrong_p = wrong_p + 1;
    last_mv = 6;
elseif (last_mv == 3)
    correct_p = correct_p + 1;
    last_mv = 0;
else
    wrong_r = wrong_r + 1;
end
end
results(7,4) = correct_p / (correct_p + wrong_p);

% Testing Supination

correct_s = 0;
wrong_s = 0;
lst_mv = 0;
for a = 5:length(mv_s)
    rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;

if(mv_s(a) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_s(a) == 1)
    g_votes = g_votes + 1;
elseif(mv_s(a) == 2)
    r_votes = r_votes + 1;
elseif(mv_s(a) == 3)
    p_votes = p_votes + 1;
elseif(mv_s(a) == 4)
    s_votes = s_votes + 1;
elseif(mv_s(a) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_s(a-1) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_s(a-1) == 1)
    g_votes = g_votes + 1;
elseif(mv_s(a-1) == 2)
    r_votes = r_votes + 1;
elseif(mv_s(a-1) == 3)
    p_votes = p_votes + 1;
elseif(mv_s(a-1) == 4)
    s_votes = s_votes + 1;
elseif(mv_s(a-1) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_s(a-2) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_s(a-2) == 1)
    g_votes = g_votes + 1;
elseif(mv_s(a-2) == 2)
    r_votes = r_votes + 1;
elseif(mv_s(a-2) == 3)
    p_votes = p_votes + 1;
elseif(mv_s(a-2) == 4)
    s_votes = s_votes + 1;
elseif(mv_s(a-2) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_s(a-3) == 0)
rest_votes = rest_votes + 1;
elseif(mv_s(a-3) == 1)
g_votes = g_votes + 1;
elseif(mv_s(a-3) == 2)
r_votes = r_votes + 1;
elseif(mv_s(a-3) == 3)
p_votes = p_votes + 1;
elseif(mv_s(a-3) == 4)
s_votes = s_votes + 1;
elseif(mv_s(a-3) == 5)
e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_s(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_s(a-4) == 1)
g_votes = g_votes + 1;
elseif(mv_s(a-4) == 2)
r_votes = r_votes + 1;
elseif(mv_s(a-4) == 3)
p_votes = p_votes + 1;
elseif(mv_s(a-4) == 4)
s_votes = s_votes + 1;
elseif(mv_s(a-4) == 5)
e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(rest_votes >= g_votes && rest_votes >= r_votes && rest_votes >= p_votes && rest_votes >= s_votes && rest_votes >= e_votes && rest_votes >= f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 0;
elseif(g_votes >= rest_votes && g_votes >= r_votes && g_votes >= p_votes && g_votes >= s_votes && g_votes >= e_votes && g_votes >= f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 1;
elseif(r_votes >= rest_votes && r_votes >= g_votes && r_votes >= p_votes && r_votes >= s_votes && r_votes >= e_votes && r_votes >= f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 2;
elseif(p_votes >= rest_votes && p_votes >= g_votes && p_votes >= r_votes && p_votes >= s_votes && p_votes >= e_votes && p_votes >= f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 3;
elseif(s_votes >= rest_votes && s_votes >= g_votes && s_votes >= r_votes && s_votes >= p_votes && s_votes >= e_votes && s_votes >= f_votes)
    correct_s = correct_s + 1;
    last_mv = 4;
elseif(e_votes >= rest_votes && e_votes >= g_votes && e_votes >= r_votes && e_votes >= p_votes && e_votes >= s_votes && e_votes >= f_votes)
    wrong_s = wrong_s + 1;
    last_mv = 5;
elseif(f_votes >= rest_votes && f_votes >= g_votes && f_votes >= r_votes && f_votes >= p_votes && f_votes >= s_votes && f_votes >= e_votes)
    wrong_s = wrong_s + 1;
    last_mv = 6;
else
    last_mv = 4;
    correct_s = correct_s + 1;
    last_mv = 0;
else
wrong_r = wrong_r + 1;
end
end
results(7,5) = correct_s / (correct_s + wrong_s);

% Testing Extension
correct_e = 0;
wrong_e = 0;
last_mv = 0;
for a = 5:length(mv_e)
    rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;

    if(mv_e(a) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_e(a) == 1)
        g_votes = g_votes + 1;
    elseif(mv_e(a) == 2)
        r_votes = r_votes + 1;
    elseif(mv_e(a) == 3)
        p_votes = p_votes + 1;
    elseif(mv_e(a) == 4)
        s_votes = s_votes + 1;
    elseif(mv_e(a) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end

    if(mv_e(a-1) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_e(a-1) == 1)
        g_votes = g_votes + 1;
    elseif(mv_e(a-1) == 2)
        r_votes = r_votes + 1;
    elseif(mv_e(a-1) == 3)
        p_votes = p_votes + 1;
    elseif(mv_e(a-1) == 4)
        s_votes = s_votes + 1;
    elseif(mv_e(a-1) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end

    if(mv_e(a-2) == 0)
        rest_votes = rest_votes + 1;
    elseif(mv_e(a-2) == 1)
        g_votes = g_votes + 1;
    elseif(mv_e(a-2) == 2)
        r_votes = r_votes + 1;
    elseif(mv_e(a-2) == 3)
        p_votes = p_votes + 1;
    elseif(mv_e(a-2) == 4)
        s_votes = s_votes + 1;
    elseif(mv_e(a-2) == 5)
        e_votes = e_votes + 1;
    else
        f_votes = f_votes + 1;
    end
if(mv_e(a-3) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_e(a-3) == 1)
    g_votes = g_votes + 1;
elseif(mv_e(a-3) == 2)
    r_votes = r_votes + 1;
elseif(mv_e(a-3) == 3)
    p_votes = p_votes + 1;
elseif(mv_e(a-3) == 4)
    s_votes = s_votes + 1;
elseif(mv_e(a-3) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_e(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_e(a-4) == 1)
    g_votes = g_votes + 1;
elseif(mv_e(a-4) == 2)
    r_votes = r_votes + 1;
elseif(mv_e(a-4) == 3)
    p_votes = p_votes + 1;
elseif(mv_e(a-4) == 4)
    s_votes = s_votes + 1;
elseif(mv_e(a-4) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(rest_votes>g_votes && rest_votes>r_votes && rest_votes>p_votes && rest_votes>s_votes && rest_votes>e_votes && rest_votes>f_votes)
    wrong_e = wrong_e + 1;
    last_mv = 0;
elseif(g_votes>rest_votes && g_votes>r_votes && g_votes>p_votes && g_votes>s_votes && g_votes>e_votes && g_votes>f_votes)
    wrong_e = wrong_e + 1;
    last_mv = 1;
elseif(r_votes>rest_votes && r_votes>g_votes && r_votes>p_votes && r_votes>s_votes && r_votes>e_votes && r_votes>f_votes)
    wrong_e = wrong_e + 1;
    last_mv = 2;
elseif(p_votes>rest_votes && p_votes>g_votes && p_votes>r_votes && p_votes>s_votes && p_votes>e_votes && p_votes>f_votes)
    wrong_e = wrong_e + 1;
    last_mv = 3;
elseif(s_votes>rest_votes && s_votes>g_votes && s_votes>r_votes && s_votes>p_votes && s_votes>e_votes && s_votes>f_votes)
    wrong_e = wrong_e + 1;
    last_mv = 4;
elseif(e_votes>rest_votes && e_votes>g_votes && e_votes>r_votes && e_votes>s_votes && e_votes>f_votes)
    correct_e = correct_e + 1;
    last_mv = 5;
elseif(f_votes>rest_votes && f_votes>g_votes && f_votes>r_votes && f_votes>s_votes && f_votes>e_votes && f_votes>f_votes)
    wrong_e = wrong_e + 1;
    last_mv = 6;
elseif(last_mv == 5)
    correct_e = correct_e + 1;
last_mv = 0;
else
    wrong_e = wrong_e + 1;
end
end
results(7,6) = correct_e / (correct_e + wrong_e);

% Testing Flexion

correct_f = 0;
wrong_f = 0;
last_mv = 0;
for a = 5:length(mv_f)
   rest_votes = 0;
g_votes = 0;
r_votes = 0;
p_votes = 0;
s_votes = 0;
e_votes = 0;
f_votes = 0;

if(mv_f(a) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a) == 4)
    s_votes = s_votes + 1;
elseif(mv_f(a) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_f(a-1) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-1) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-1) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-1) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-1) == 4)
    s_votes = s_votes + 1;
elseif(mv_f(a-1) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end
if(mv_f(a-2) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-2) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-2) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-2) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-2) == 4)
    s_votes = s_votes + 1;
elseif(mv_f(a-2) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_f(a-3) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-3) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-3) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-3) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-3) == 4)
    s_votes = s_votes + 1;
elseif(mv_f(a-3) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(mv_f(a-4) == 0)
    rest_votes = rest_votes + 1;
elseif(mv_f(a-4) == 1)
    g_votes = g_votes + 1;
elseif(mv_f(a-4) == 2)
    r_votes = r_votes + 1;
elseif(mv_f(a-4) == 3)
    p_votes = p_votes + 1;
elseif(mv_f(a-4) == 4)
    s_votes = s_votes + 1;
elseif(mv_f(a-4) == 5)
    e_votes = e_votes + 1;
else
    f_votes = f_votes + 1;
end

if(rest_votes > g_votes && rest_votes > r_votes && rest_votes > p_votes && rest_votes > s_votes && rest_votes > e_votes && rest_votes > f_votes)
    wrong_f = wrong_f + 1;
    last_mv = 0;
else( g_votes > rest_votes && g_votes > r_votes && g_votes > p_votes && g_votes > s_votes && g_votes > e_votes && g_votes > f_votes)
    wrong_f = wrong_f + 1;
    last_mv = 1;
else(r_votes > rest_votes && r_votes > g_votes && r_votes > p_votes && r_votes > s_votes && r_votes > e_votes && r_votes > f_votes)
    wrong_f = wrong_f + 1;
    last_mv = 2;
else(p_votes > rest_votes && p_votes > g_votes && p_votes > r_votes && p_votes > s_votes && p_votes > e_votes && p_votes > f_votes)
    wrong_f = wrong_f + 1;
    last_mv = 3;
else(s_votes > rest_votes && s_votes > g_votes && s_votes > r_votes && s_votes > p_votes && s_votes > e_votes && s_votes > f_votes)
    wrong_f = wrong_f + 1;
    last_mv = 4;
else(e_votes > rest_votes && e_votes > g_votes && e_votes > r_votes && e_votes > s_votes && e_votes > p_votes)
    e_votes = e_votes + 1;
    last_mv = 5;
else(f_votes > rest_votes && f_votes > g_votes && f_votes > r_votes && f_votes > s_votes && f_votes > e_votes)
    correct_f = correct_f + 1;
    last_mv = 6;
elseif(last_mv == 6)
    correct_f = correct_f + 1;
    last_mv = 0;
else
    wrong_f = wrong_f + 1;
end
end
results(7,7) = correct_f / (correct_f + wrong_f);
results(7,8) =
(correct_g+correct_r+correct_p+correct_s+correct_rest)/(wrong_g+wrong_r+wrong_p+wrong_s+wrong_rest+correct_g+correct_
 r+correct_p+correct_s+correct_rest);
Appendix E: Matlab Code for TDNN Preprocessing of EMG Data

% This file performs all of the preprocessing on data that has been
% collected by the upper extremity data acquisition system using Bio-Radio
% Capture Lite. Specifically the file low pass filters all EMG data and
% transforms the position data from voltages to position data.

% This code loads the data for each
load elbow90.csv
load elbow180.csv
load shoulder0.csv
load shoulder90.csv
load train_slow_elbow.csv
load test_slow_elbow.csv
load train_fast_elbow.csv
load test_fast_elbow.csv
load train_slow_shoulder.csv
load test_slow_shoulder.csv
load train_fast_shoulder.csv
load test_fast_shoulder.csv
load train_slow_reaching.csv
load test_slow_reaching.csv
load train_fast_reaching.csv
load test_fast_reaching.csv

% Obtaining y = mx + b for elbow and shoulder data. This is later used to
% transform the data into degrees.
elb_pt1 = zeros(1,2);
elb_pt1 = [mean(elbow180(:,1)),180];
elb_pt2 = zeros(1,2);
elb_pt2 = [mean(elbow90(:,1)),90];
elb_m = (elb_pt1(2)-elb_pt2(2))/(elb_pt1(1)-elb_pt2(1));
elb_b = -elb_m*elb_pt1(1) + elb_pt1(2);
shd_pt1 = zeros(1,2);
shd_pt1 = [mean(shoulder0(:,2)),0];
shd_pt2 = zeros(1,2);
shd_pt2 = [mean(shoulder90(:,2)),90];
shd_m = (shd_pt1(2)-shd_pt2(2))/(shd_pt1(1)-shd_pt2(1));
shd_b = -shd_m*shd_pt1(1) + shd_pt1(2);

% The neural net files and testing files use these variable names
train_es = train_slow_elbow;
train_ef = train_fast_elbow;
train_ss = train_slow_shoulder;
train_sf = train_fast_shoulder;
train_rs = train_slow_reaching;
train_rf = train_fast_reaching;
test_es = test_slow_elbow;
test_ef = test_fast_elbow;
test_ss = test_slow_shoulder;
test_sf = test_fast_shoulder;
test_rs = test_slow_reaching;
test_rf = test_fast_reaching;

% Transforming All Position Data from Voltage to Degrees
% Also Normalizes Position Data Between 0 and 1
train_es(:,2) = shd_m*train_slow_elbow(:,2)+shd_b;
train_es(:,2) = (train_es(:,2)+30)/185;
train_es(:,1) = elb_m*train_slow_elbow(:,1)+elb_b;
train_es(:,1) = (train_es(:,1))/195;
train_ef(:,2) = shd_m*train_fast_elbow(:,2)+shd_b;
train_ef(:,2) = (train_ef(:,2)+30)/185;
train_ef(:,1) = elb_m*train_fast_elbow(:,1)+elb_b;
train_ef(:,1) = (train_ef(:,1))/195;
train_ss(:,2) = shd_m*train_slow_shoulder(:,2)+shd_b;
train_ss(:,2) = (train_ss(:,2)+30)/185;
train_ss(:,1) = elb_m*train_slow_shoulder(:,1)+elb_b;
train_ss(:,1) = (train_ss(:,1))/195;
train_sf(:,2) = shd_m*train_fast_shoulder(:,2)+shd_b;
train_sf(:,2) = (train_sf(:,2)+30)/185;
train_sf(:,1) = elb_m*train_fast_shoulder(:,1)+elb_b;
train_sf(:,1) = (train_sf(:,1))/195;
train_rs(:,2) = shd_m*train_slow_reaching(:,2)+shd_b;
train_rs(:,2) = (train_rs(:,2)+30)/185;
train_rs(:,1) = elb_m*train_slow_reaching(:,1)+elb_b;
train_rs(:,1) = (train_rs(:,1))/195;
train_rf(:,2) = shd_m*train_fast_reaching(:,2)+shd_b;
train_rf(:,2) = (train_rf(:,2)+30)/185;
train_rf(:,1) = elb_m*train_fast_reaching(:,1)+elb_b;
train_rf(:,1) = (train_rf(:,1))/195;
test_es(:,2) = shd_m*test_slow_elbow(:,2)+shd_b;
test_es(:,2) = (test_es(:,2)+30)/185;
test_es(:,1) = elb_m*test_slow_elbow(:,1)+elb_b;
test_es(:,1) = (test_es(:,1))/195;
test_ef(:,2) = shd_m*test_fast_elbow(:,2)+shd_b;
test_ef(:,2) = (test_ef(:,2)+30)/185;
test_ef(:,1) = elb_m*test_fast_elbow(:,1)+elb_b;
test_ef(:,1) = (test_ef(:,1))/195;
test_ss(:,2) = shd_m*test_slow_shoulder(:,2)+shd_b;
test_ss(:,2) = (test_ss(:,2)+30)/185;
test_ss(:,1) = elb_m*test_slow_shoulder(:,1)+elb_b;
test_ss(:,1) = (test_ss(:,1))/195;
test_sf(:,2) = shd_m*test_fast_shoulder(:,2)+shd_b;
test_sf(:,2) = (test_sf(:,2)+30)/185;
test_sf(:,1) = elb_m*test_fast_shoulder(:,1)+elb_b;
test_sf(:,1) = (test_sf(:,1))/195;
test_rs(:,2) = shd_m*test_slow_reaching(:,2)+shd_b;
test_rs(:,2) = (test_rs(:,2)+30)/185;
test_rs(:,1) = elb_m*test_slow_reaching(:,1)+elb_b;
test_rs(:,1) = (test_rs(:,1))/195;
test_rf(:,2) = shd_m*test_fast_reaching(:,2)+shd_b;
test_rf(:,2) = (test_rf(:,2)+30)/185;
test_rf(:,1) = elb_m*test_fast_reaching(:,1)+elb_b;
test_rf(:,1) = (test_rf(:,1))/195;

% Visually inspect all EMG data
figure,subplot(6,1,1),plot(train_es(:,1))
figure,subplot(6,1,2),plot(train_es(:,2))
figure,subplot(6,1,3),plot(train_es(:,3))
figure,subplot(6,1,4),plot(train_es(:,4))
figure,subplot(6,1,5),plot(train_es(:,5))
figure,subplot(6,1,6),plot(train_es(:,6))
figure, subplot(6,1,1), plot(train_ss(:,1))
subplot(6,1,2), plot(train_ss(:,2))
subplot(6,1,3), plot(train_ss(:,3))
subplot(6,1,4), plot(train_ss(:,4))
subplot(6,1,5), plot(train_ss(:,5))
subplot(6,1,6), plot(train_ss(:,6))
figure, subplot(6,1,1), plot(train_sf(:,1))
subplot(6,1,2), plot(train_sf(:,2))
subplot(6,1,3), plot(train_sf(:,3))
subplot(6,1,4), plot(train_sf(:,4))
subplot(6,1,5), plot(train_sf(:,5))
subplot(6,1,6), plot(train_sf(:,6))
figure, subplot(6,1,1), plot(train_rs(:,1))
subplot(6,1,2), plot(train_rs(:,2))
subplot(6,1,3), plot(train_rs(:,3))
subplot(6,1,4), plot(train_rs(:,4))
subplot(6,1,5), plot(train_rs(:,5))
subplot(6,1,6), plot(train_rs(:,6))
figure, subplot(6,1,1), plot(test_es(:,1))
subplot(6,1,2), plot(test_es(:,2))
subplot(6,1,3), plot(test_es(:,3))
subplot(6,1,4), plot(test_es(:,4))
subplot(6,1,5), plot(test_es(:,5))
subplot(6,1,6), plot(test_es(:,6))
figure, subplot(6,1,1), plot(test_ef(:,1))
subplot(6,1,2), plot(test_ef(:,2))
subplot(6,1,3), plot(test_ef(:,3))
subplot(6,1,4), plot(test_ef(:,4))
subplot(6,1,5), plot(test_ef(:,5))
subplot(6,1,6), plot(test_ef(:,6))
figure, subplot(6,1,1), plot(test_ss(:,1))
subplot(6,1,2), plot(test_ss(:,2))
subplot(6,1,3), plot(test_ss(:,3))
subplot(6,1,4), plot(test_ss(:,4))
subplot(6,1,5), plot(test_ss(:,5))
subplot(6,1,6), plot(test_ss(:,6))
figure, subplot(6,1,1), plot(test_sf(:,1))
subplot(6,1,2), plot(test_sf(:,2))
subplot(6,1,3), plot(test_sf(:,3))
subplot(6,1,4), plot(test_sf(:,4))
subplot(6,1,5), plot(test_sf(:,5))
subplot(6,1,6), plot(test_sf(:,6))
figure, subplot(6,1,1), plot(test_rs(:,1))
subplot(6,1,2), plot(test_rs(:,2))
subplot(6,1,3), plot(test_rs(:,3))
subplot(6,1,4), plot(test_rs(:,4))
subplot(6,1,5), plot(test_rs(:,5))
subplot(6,1,6), plot(test_rs(:,6))
figure, subplot(6,1,1), plot(test_rf(:,1))
subplot(6,1,2), plot(test_rf(:,2))
subplot(6,1,3), plot(test_rf(:,3))
subplot(6,1,4), plot(test_rf(:,4))
subplot(6,1,5), plot(test_rf(:,5))
subplot(6,1,6), plot(test_rf(:,6))
% Low pass filter, Order 4, Cutoff at 4Hz, Butterworth
n = 4;
fs = 960;
f3db = 4;
d = fdesign.lowpass('n,f3db',n,f3db,fs);
lpfilt = design(d,'butter');
freqz(lpfilt);

% Low Pass Filter All EMG Data
train_es(:,3) = filter(lpfilt, abs(train_es(:,3)));
train_es(:,4) = filter(lpfilt, abs(train_es(:,4)));
train_es(:,5) = filter(lpfilt, abs(train_es(:,5)));
train_es(:,6) = filter(lpfilt, abs(train_es(:,6)));
train_ef(:,3) = filter(lpfilt, abs(train_ef(:,3)));
train_ef(:,4) = filter(lpfilt, abs(train_ef(:,4)));
train_ef(:,5) = filter(lpfilt, abs(train_ef(:,5)));
train_ef(:,6) = filter(lpfilt, abs(train_ef(:,6)));
train_ss(:,3) = filter(lpfilt, abs(train_ss(:,3)));
train_ss(:,4) = filter(lpfilt, abs(train_ss(:,4)));
train_ss(:,5) = filter(lpfilt, abs(train_ss(:,5)));
train_ss(:,6) = filter(lpfilt, abs(train_ss(:,6)));
train_sf(:,3) = filter(lpfilt, abs(train_sf(:,3)));
train_sf(:,4) = filter(lpfilt, abs(train_sf(:,4)));
train_sf(:,5) = filter(lpfilt, abs(train_sf(:,5)));
train_sf(:,6) = filter(lpfilt, abs(train_sf(:,6)));
train_rf(:,3) = filter(lpfilt, abs(train_rf(:,3)));
train_rf(:,4) = filter(lpfilt, abs(train_rf(:,4)));
train_rf(:,5) = filter(lpfilt, abs(train_rf(:,5)));
train_rf(:,6) = filter(lpfilt, abs(train_rf(:,6)));
train_es(:,3) = filter(lpfilt, abs(test_es(:,3)));
train_es(:,4) = filter(lpfilt, abs(test_es(:,4)));
train_es(:,5) = filter(lpfilt, abs(test_es(:,5)));
train_es(:,6) = filter(lpfilt, abs(test_es(:,6)));
train_ef(:,3) = filter(lpfilt, abs(test_ef(:,3)));
train_ef(:,4) = filter(lpfilt, abs(test_ef(:,4)));
train_ef(:,5) = filter(lpfilt, abs(test_ef(:,5)));
train_ef(:,6) = filter(lpfilt, abs(test_ef(:,6)));
train_ss(:,3) = filter(lpfilt, abs(test_ss(:,3)));
train_ss(:,4) = filter(lpfilt, abs(test_ss(:,4)));
train_ss(:,5) = filter(lpfilt, abs(test_ss(:,5)));
train_ss(:,6) = filter(lpfilt, abs(test_ss(:,6)));
train_sf(:,3) = filter(lpfilt, abs(test_sf(:,3)));
train_sf(:,4) = filter(lpfilt, abs(test_sf(:,4)));
train_sf(:,5) = filter(lpfilt, abs(test_sf(:,5)));
train_sf(:,6) = filter(lpfilt, abs(test_sf(:,6)));
train_rf(:,3) = filter(lpfilt, abs(test_rf(:,3)));
train_rf(:,4) = filter(lpfilt, abs(test_rf(:,4)));
train_rf(:,5) = filter(lpfilt, abs(test_rf(:,5)));
train_rf(:,6) = filter(lpfilt, abs(test_rf(:,6)));

270
Appendix F: Matlab Code for Creating TDNN

load train_data.mat
load test_data.mat

delay = 96; % 100ms intervals at 960Hz
total = 288; % 300ms total delay

train_input = zeros(length(train_es)+length(train_ef)+length(train_ss)+length(train_sf)+length(train_rs)+length(train_rf)-
(6*delay),(total/delay+1)*4);
train_output = zeros(length(train_input),2);
for a = total+1:length(train_es)
    offset = a-total;
    train_input(offset,1:1*(total/delay)+1) = [train_es(a,3), train_es(a-delay,3), train_es(a-2*delay,3), train_es(a-3*delay,3)];
    train_input(offset,1*(total/delay)+2:2*(total/delay)+2) = [train_es(a,4), train_es(a-delay,4), train_es(a-2*delay,4), train_es(a-3*delay,4)];
    train_input(offset,2*(total/delay)+3:3*(total/delay)+3) = [train_es(a,5), train_es(a-delay,5), train_es(a-2*delay,5), train_es(a-3*delay,5)];
    train_input(offset,3*(total/delay)+4:4*(total/delay)+4) = [train_es(a,6), train_es(a-delay,6), train_es(a-2*delay,6), train_es(a-3*delay,6)];
    train_output(offset,1) = train_es(a,1);
    train_output(offset,2) = train_es(a,2);
end
for a = total+1:length(train_ef)
    offset = a-total;
    train_input(offset,1:1*(total/delay)+1) = [train_ef(a,3), train_ef(a-delay,3), train_ef(a-2*delay,3), train_ef(a-3*delay,3)];
    train_input(offset,1*(total/delay)+2:2*(total/delay)+2) = [train_ef(a,4), train_ef(a-delay,4), train_ef(a-2*delay,4), train_ef(a-3*delay,4)];
    train_input(offset,2*(total/delay)+3:3*(total/delay)+3) = [train_ef(a,5), train_ef(a-delay,5), train_ef(a-2*delay,5), train_ef(a-3*delay,5)];
    train_input(offset,3*(total/delay)+4:4*(total/delay)+4) = [train_ef(a,6), train_ef(a-delay,6), train_ef(a-2*delay,6), train_ef(a-3*delay,6)];
    train_output(offset,1) = train_ef(a,1);
    train_output(offset,2) = train_ef(a,2);
end
for a = total+1:length(train_ss)
    offset = a-total;
    train_input(offset,1:1*(total/delay)+1) = [train_ss(a,3), train_ss(a-delay,3), train_ss(a-2*delay,3), train_ss(a-3*delay,3)];
    train_input(offset,1*(total/delay)+2:2*(total/delay)+2) = [train_ss(a,4), train_ss(a-delay,4), train_ss(a-2*delay,4), train_ss(a-3*delay,4)];
    train_input(offset,2*(total/delay)+3:3*(total/delay)+3) = [train_ss(a,5), train_ss(a-delay,5), train_ss(a-2*delay,5), train_ss(a-3*delay,5)];
    train_input(offset,3*(total/delay)+4:4*(total/delay)+4) = [train_ss(a,6), train_ss(a-delay,6), train_ss(a-2*delay,6), train_ss(a-3*delay,6)];
    train_output(offset,1) = train_ss(a,1);
    train_output(offset,2) = train_ss(a,2);
end
for a = total+1:length(train_sf)
    offset = a-total;
    train_input(offset,1:1*(total/delay)+1) = [train_sf(a,3), train_sf(a-delay,3), train_sf(a-2*delay,3), train_sf(a-3*delay,3)];
    train_input(offset,1*(total/delay)+2:2*(total/delay)+2) = [train_sf(a,4), train_sf(a-delay,4), train_sf(a-2*delay,4), train_sf(a-3*delay,4)];
    train_input(offset,2*(total/delay)+3:3*(total/delay)+3) = [train_sf(a,5), train_sf(a-delay,5), train_sf(a-2*delay,5), train_sf(a-3*delay,5)];
    train_input(offset,3*(total/delay)+4:4*(total/delay)+4) = [train_sf(a,6), train_sf(a-delay,6), train_sf(a-2*delay,6), train_sf(a-3*delay,6)];
    train_output(offset,1) = train_sf(a,1);
    train_output(offset,2) = train_sf(a,2);
end
for a = total+1:length(train_rs)
offset = a-5*total+length(train_es)+length(train_ef)+length(train_ss)+length(train_sf);
train_input(offset,1:1*(total/delay)+1) = [train_rs(a,3), train_rs(a-delay,3), train_rs(a-2*delay,3), train_rs(a-3*delay,3)];
train_input(offset,1*(total/delay)+2:2*(total/delay)+2) = [train_rs(a,4), train_rs(a-delay,4), train_rs(a-2*delay,4), train_rs(a-3*delay,4)];
train_input(offset,2*(total/delay)+3:3*(total/delay)+3) = [train_rs(a,5), train_rs(a-delay,5), train_rs(a-2*delay,5), train_rs(a-3*delay,5)];
train_input(offset,3*(total/delay)+4:4*(total/delay)+4) = [train_rs(a,6), train_rs(a-delay,6), train_rs(a-2*delay,6), train_rs(a-3*delay,6)];
train_output(offset,1) = train_rs(a,1);
train_output(offset,2) = train_rs(a,2);
end
for a = total+1:length(train_rf)
offset = a-6*total+length(train_es)+length(train_ef)+length(train_ss)+length(train_rf)+length(train_rf);
train_input(offset,1:1*(total/delay)+1) = [train_rf(a,3), train_rf(a-delay,3), train_rf(a-2*delay,3), train_rf(a-3*delay,3)];
train_input(offset,1*(total/delay)+2:2*(total/delay)+2) = [train_rf(a,4), train_rf(a-delay,4), train_rf(a-2*delay,4), train_rf(a-3*delay,4)];
train_input(offset,2*(total/delay)+3:3*(total/delay)+3) = [train_rf(a,5), train_rf(a-delay,5), train_rf(a-2*delay,5), train_rf(a-3*delay,5)];
train_input(offset,3*(total/delay)+4:4*(total/delay)+4) = [train_rf(a,6), train_rf(a-delay,6), train_rf(a-2*delay,6), train_rf(a-3*delay,6)];
train_output(offset,1) = train_rf(a,1);
train_output(offset,2) = train_rf(a,2);
end
net = newff(train_input',train_output',[10],{'tansig'},'trainbfg')
net.trainParam.epochs = 250; %Maximum number of epochs to train
%net.trainParam.goal 0 %Performance goal
net.trainParam.lr = 3; %Learning rate
net.trainParam.max_fail = 10 %Maximum validation failures
%net.trainParam.min_grad %1e-10 %Minimum performance gradient
net.trainParam.show = 25; %Epochs between displays (NaN for no displays)
%net.trainParam.time %Maximum time to train in seconds

[net, tr] = train(net, train_input', train_output');
Appendix G: Matlab Code for Testing the TDNN

```matlab
% This file uses the trained TDNN and uses the test trials of data to test
% the TDNN.

load test_data.mat

delay = 96;  % 100ms intervals at 960Hz
total = 288;  % 300ms total delay

test = test_es;
nn_input = zeros(1,(total/delay+1)*4);
nn_output_es = zeros(length(test)-total,2);
for a = total+1:length(test)
    nn_input(1:1*(total/delay)+1) = [test(a,3), test(a-delay,3), test(a-2*delay,3), test(a-3*delay,3)];
    nn_input(1*(total/delay)+2:2*(total/delay)+2) = [test(a,4), test(a-delay,4), test(a-2*delay,4), test(a-3*delay,4)];
    nn_input(2*(total/delay)+3:3*(total/delay)+3) = [test(a,5), test(a-delay,5), test(a-2*delay,5), test(a-3*delay,5)];
    nn_input(3*(total/delay)+4:4*(total/delay)+4) = [test(a,6), test(a-delay,6), test(a-2*delay,6), test(a-3*delay,6)];
    nn_output_es(a-total,1:2) = sim(net,[nn_input]');
end

figure,subplot(2,1,1),plot((1/960:1/960:1/960*length(nn_output_es(:,1))),(test(total+1:end,2)*185)-30,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_es(:,1))),(nn_output_es(:,2)*185)-30,'g')
title('Shoulder Position During Slow Elbow Movement')
xlabel('Time (s)')
ylabel('Shoulder Position (Degrees)')
legend('Actual','TDNN')
subplot(2,1,2),plot((1/960:1/960:1/960*length(nn_output_es(:,1))),(test(total+1:end,1)*195,'b'))
hold on
plot((1/960:1/960:1/960*length(nn_output_es(:,2))),(nn_output_es(:,1)*195,'g'))
title('Elbow Position During Slow Elbow Movement')
xlabel('Time (s)')
ylabel('Elbow Position (Degrees)')

end

figure,subplot(2,1,1),plot((1/960:1/960:1/960*length(nn_output_ef(:,1))),(test(total+1:end,2)*185)-30,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_ef(:,1))),(nn_output_ef(:,2)*185)-30,'g')
title('Shoulder Position During Fast Elbow Movement')
xlabel('Time (s)')
ylabel('Shoulder Position (Degrees)')
legend('Actual','TDNN')
subplot(2,1,2),plot((1/960:1/960:1/960*length(nn_output_ef(:,1))),(test(total+1:end,1)*195,'b'))
hold on
plot((1/960:1/960:1/960*length(nn_output_ef(:,2))),(nn_output_ef(:,1)*195,'g'))
title('Elbow Position During Fast Elbow Movement')
xlabel('Time (s)')
ylabel('Elbow Position (Degrees)')

end

```

```matlab
test = test_ss;
nn_input = zeros(1,(total/delay+1)*4);
```
nn_output_ss = zeros(length(test)-total,2);
for a = total+1:length(test)
    nn_input(1:1*(total/delay)+1) = [test(a,3), test(a-delay,3), test(a-2*delay,3), test(a-3*delay,3)];
    nn_input(1*(total/delay)+2:2*(total/delay)+2) = [test(a,4), test(a-delay,4), test(a-2*delay,4), test(a-3*delay,4)];
    nn_input(2*(total/delay)+3:3*(total/delay)+3) = [test(a,5), test(a-delay,5), test(a-2*delay,5), test(a-3*delay,5)];
    nn_input(3*(total/delay)+4:4*(total/delay)+4) = [test(a,6), test(a-delay,6), test(a-2*delay,6), test(a-3*delay,6)];
    nn_output_ss(a-total,1:2) = sim(net,[nn_input]');
end

figure,subplot(2,1,1),plot((1/960:1/960:1/960*length(nn_output_ss(:,1))),(test(total+1:end,2)*185)-30,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_ss(:,1))),(nn_output_ss(:,2)*185)-30,'g')
title('Shoulder Position During Slow Shoulder Movement')
xlabel('Time (s)')
ylabel('Shoulder Position (Degrees)')

subplot(2,1,2),plot((1/960:1/960:1/960*length(nn_output_ss(:,1))),test(total+1:end,1)*195,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_ss(:,2))),nn_output_ss(:,1)*195,'g')
title('Elbow Position During Slow Shoulder Movement')
xlabel('Time (s)')
ylabel('Elbow Position (Degrees)')
legend('Actual','TDNN')


nn_output_sf = zeros(length(test)-total,2);
for a = total+1:length(test)
    nn_input(1:1*(total/delay)+1) = [test(a,3), test(a-delay,3), test(a-2*delay,3), test(a-3*delay,3)];
    nn_input(1*(total/delay)+2:2*(total/delay)+2) = [test(a,4), test(a-delay,4), test(a-2*delay,4), test(a-3*delay,4)];
    nn_input(2*(total/delay)+3:3*(total/delay)+3) = [test(a,5), test(a-delay,5), test(a-2*delay,5), test(a-3*delay,5)];
    nn_input(3*(total/delay)+4:4*(total/delay)+4) = [test(a,6), test(a-delay,6), test(a-2*delay,6), test(a-3*delay,6)];
    nn_output_sf(a-total,1:2) = sim(net,[nn_input]');
end

figure,subplot(2,1,1),plot((1/960:1/960:1/960*length(nn_output_sf(:,1))),(test(total+1:end,2)*185)-30,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_sf(:,1))),(nn_output_sf(:,2)*185)-30,'g')
title('Shoulder Position During Fast Shoulder Movement')
xlabel('Time (s)')
ylabel('Shoulder Position (Degrees)')

subplot(2,1,2),plot((1/960:1/960:1/960*length(nn_output_sf(:,1))),test(total+1:end,1)*195,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_sf(:,2))),nn_output_sf(:,1)*195,'g')
title('Elbow Position During Fast Shoulder Movement')
xlabel('Time (s)')
ylabel('Elbow Position (Degrees)')
legend('Actual','TDNN')


nn_output_rs = zeros(length(test)-total,2);
for a = total+1:length(test)
    nn_input(1:1*(total/delay)+1) = [test(a,3), test(a-delay,3), test(a-2*delay,3), test(a-3*delay,3)];
    nn_input(1*(total/delay)+2:2*(total/delay)+2) = [test(a,4), test(a-delay,4), test(a-2*delay,4), test(a-3*delay,4)];
    nn_input(2*(total/delay)+3:3*(total/delay)+3) = [test(a,5), test(a-delay,5), test(a-2*delay,5), test(a-3*delay,5)];
    nn_input(3*(total/delay)+4:4*(total/delay)+4) = [test(a,6), test(a-delay,6), test(a-2*delay,6), test(a-3*delay,6)];
    nn_output_rs(a-total,1:2) = sim(net,[nn_input]');
end

figure,subplot(2,1,1),plot((1/960:1/960:1/960*length(nn_output_rs(:,1))),(test(total+1:end,2)*185)-30,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_rs(:,1))),(nn_output_rs(:,2)*185)-30,'g')
title('Shoulder Position During Slow Reaching Movement')
xlabel('Time (s)')
ylabel('Shoulder Position (Degrees)')
subplot(2,1,2), plot((1/960:1/960:1/960*length(nn_output_rs(:,1))),test(total+1:end,1)*195,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_rs(:,2))),nn_output_rs(:,1)*195,'g')
title('Elbow Position During Slow Reaching Movement')
xlabel('Time (s)')
ylabel('Elbow Position (Degrees)')
legend('Actual','TDNN')

(test = test_rf;
nn_input = zeros(1,(total/delay+1)*4);
nn_output_rf = zeros(length(test)-total,2);
for a = total+1:length(test)
    nn_input(1:1*(total/delay)+1) = [test(a,3), test(a-delay,3), test(a-2*delay,3), test(a-3*delay,3)];
    nn_input(1*(total/delay)+2:2*(total/delay)+2) = [test(a,4), test(a-delay,4), test(a-2*delay,4), test(a-3*delay,4)];
    nn_input(2*(total/delay)+3:3*(total/delay)+3) = [test(a,5), test(a-delay,5), test(a-2*delay,5), test(a-3*delay,5)];
    nn_input(3*(total/delay)+4:4*(total/delay)+4) = [test(a,6), test(a-delay,6), test(a-2*delay,6), test(a-3*delay,6)];
    nn_output_rf(a-total,1:2) = sim(net,[nn_input]');
end

figure, subplot(2,1,1), plot((1/960:1/960:1/960*length(nn_output_rf(:,1))),(test(total+1:end,2)*185)-30,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_rf(:,2))),(nn_output_rf(:,2)*185)-30,'g')
title('Shoulder Position During Fast Reaching Movement')
xlabel('Time (s)')
ylabel('Shoulder Position (Degrees)')
subplot(2,1,2), plot((1/960:1/960:1/960*length(nn_output_rf(:,1))),(test(total+1:end,1)*195,'b')
hold on
plot((1/960:1/960:1/960*length(nn_output_rf(:,2))),(nn_output_rf(:,1)*195,'g')
title('Elbow Position During Fast Reaching Movement')
xlabel('Time (s)')
ylabel('Elbow Position (Degrees)')
legend('Actual','TDNN')