

Universidad Carlos III de Madrid

# IMPLEMENTATION OF A NEURAL NETWORK-BASED ELECTROMYOGRAPHIC CONTROL SYSTEM FOR A PRINTED ROBOTIC HAND

BACHELOR IN BIOMEDICAL ENGINEERING

BACHELOR THESIS

AUTHOR: Irene Méndez Guerra

TUTOR: Álvaro Villoslada Peciña

DIRECTOR: Luis Enrique Moreno Lorente



Universidad  
Carlos III de Madrid

2016



# BACHELOR THESIS

## IMPLEMENTATION OF A NEURAL NETWORK-BASED ELECTROMYOGRAPHIC CONTROL SYSTEM FOR A PRINTED ROBOTIC HAND

AUTHOR: IRENE MÉNDEZ GUERRA

TUTOR: ÁLVARO VILLOSLADA PECIÑA

DIRECTOR: LUIS ENRIQUE MORENO LORENTE

## REVIEW COMMITTEE

CHAIR: ESTEBAN MORO EGIDO \_\_\_\_\_

MEMBER: ALEJANDRO RIVERA LAVADO \_\_\_\_\_

SECRETARY: JORGE RIPOLL LORENZO \_\_\_\_\_

SUBSTITUTE: JAVIER PASCAU GONZÁLEZ GARZÓN \_\_\_\_\_

Performed the presentation and after reading the bachelor thesis on the day 8th of March of 2016 in Leganés, in the Escuela Politécnica Superior of Universidad Carlos III Madrid, the review committee agrees to grant the qualification of:



## ACKNOWLEDGEMENTS

I would like to express my gratitude to the following people, without whom, this project could not have been possible.

First of all, I have to thank the director of my thesis Luis Enrique Moreno Lorente, for giving me this incredible opportunity and for his guidance throughout the bachelor thesis development.

Foremost, I wish to express my most sincere thanks to Álvaro Villoslada Peciña, tutor of the present thesis, for his inestimable help and for passing me on his enthusiasm and perspective on this field. It has been a pleasure working with him.

I must also thank my colleague Esperanza Marín Conde for her patience, dedication and team work during the development of the project. We finally made it!

To conclude, getting through this dissertation has required more than academic support, so I cannot begin to express my gratitude to my family and friends for holding me and not letting me give up.

# TABLE OF CONTENTS

TABLE OF CONTENTS.....	iv
INDEX OF FIGURES.....	vii
INDEX OF TABLES.....	ix
ABSTRACT.....	1
1. INTRODUCTION.....	3
1.1. BACKGROUND.....	3
1.2. MOTIVATION.....	3
1.3. OBJECTIVES.....	4
1.4. STRUCTURE.....	4
2. EMG SIGNAL.....	5
2.1. PHYSIOLOGY.....	5
2.2. ACQUISITION.....	7
2.2.1. Target muscles.....	7
2.2.2. Electrodes.....	8
2.2.3. EMG electronics and preprocessing.....	9
3. EMG SIGNAL PROCESSING AND CLASSIFICATION.....	11
3.1. DATA SEGMENTATION.....	12
3.2. FEATURE EXTRACTION.....	13
3.2.1. Time domain.....	14
3.2.2. Frequency domain.....	14
3.2.3. Time-frequency domain.....	15
3.3. DIMENSIONALITY REDUCTION.....	17
3.3.1. Feature subset selection (FSS).....	17
3.3.2. Feature projection.....	18
3.4. CLASSIFICATION.....	19
3.4.1. Bayesian classifier (BC).....	20
3.4.2. Neural networks (NN).....	20
3.4.3. Fuzzy inference systems (FIS).....	23
3.4.4. Linear discriminant analysis (LDA).....	23
3.4.5. Support vector machine (SMV) classifiers.....	24
3.4.6. Hidden markov models (HMM).....	25
3.4.7. K-nearest neighbor (KNN).....	26
3.4.8. Classifier comparison.....	26

3.4.9.	Combination of classifiers .....	27
4.	EMG ACQUISITION SYSTEM .....	30
4.1.	EMG ELECTRONIC CIRCUIT .....	30
4.1.1.	Fully-differential measurement .....	31
4.1.2.	Common mode active feedback .....	31
4.1.3.	Signal digitalization.....	32
4.2.	STM32F4 MICROCONTROLLER.....	33
4.3.	ACQUISITION SOFTWARE .....	34
4.3.1.	Simulink rcp methodology .....	34
4.3.2.	Simulink acquisition models.....	34
5.	NEURAL NETWORK-BASED MYOELECTRIC CONTROL SYSTEM.....	42
5.1.	GESTURES.....	42
5.2.	ACQUISITION PROCEDURE .....	42
5.3.	IMPLEMENTATION OF THE MYOELECTRIC CONTROL SYSTEM .....	43
5.3.1.	Data segmentation and feature extraction.....	44
5.3.2.	Dimensionality reduction .....	45
5.3.3.	Neural network generation and training .....	46
5.4.	SIMULINK CONTROL MODEL OF THE NNMCS.....	49
6.	TESTS AND RESULTS .....	53
6.1.	OPTIMAL NEURAL NETWORK CONFIGURATION .....	53
6.1.1.	Mmotion repetitions of the neural network training set.....	53
6.1.2.	LDA reduced dimensions.....	54
6.1.3.	Training algorithm .....	54
6.1.4.	Number of hidden neurons.....	55
6.2.	DIFFERENT SESSION TESTING.....	55
6.3.	EVALUATION FOR REAL-TIME APPLICATION .....	55
6.4.	RESULTS.....	56
6.4.1.	Results of optimal neural network configuration .....	56
6.4.2.	Results of different session testing .....	59
6.4.3.	Results of real-time application evaluation .....	60
7.	CONCLUSIONS AND FUTURE WORK.....	61
7.1.	CONCLUSIONS .....	61
7.2.	FUTURE WORK .....	62
	REFERENCES .....	65
	ANNEX .....	69
A.1.	WORK BREAKDOWN STRUCTURE .....	69

A.2.	LIST OF MATERIALS .....	70
A.3.	PROJECT BUDGET .....	70
A.3.1.	Materials .....	70
A.3.2.	Informatics .....	70
A.3.3.	Labor cost.....	71
A.3.4.	Total budget.....	71



## INDEX OF FIGURES

Figure 2.1: Motor unit arrangement .....	5
Figure 2.2: EMG signal composition.....	6
Figure 2.3: EMG signal time and frequency domains .....	6
Figure 2.4: Forearm superficial muscles anterior and posterior view .....	7
Figure 2.5: Differential amplifier configuration .....	9
Figure 3.1: Stages for developing myoelectric control systems.....	11
Figure 3.2: a) Disjoint segmentation and b) Overlapped segmentation.....	12
Figure 3.3: Time-frequency tiling of a) STFT, b) WT and c) WPT.....	15
Figure 3.4: Comparison of the classification performance of MRF selection and ULDA transformation .....	19
Figure 3.5: Artificial neuron model .....	20
Figure 3.6: Fully connected MLP with one input layer, one hidden and one output layer .....	21
Figure 3.7: Back-propagation neural network (BPNN).....	22
Figure 3.8: Fuzzy interface system .....	23
Figure 3.9: Representation of the optimal hyperplane as class boundary and its relation with support vectors. ....	24
Figure 3.10: a) OAA and b) OAO configurations.....	25
Figure 3.11: Components of HMM.....	25
Figure 3.12: Comparison of the classification error among the tested classifiers.....	27
Figure 3.13: Steps of a myoelectric control system for pattern recognition, with the employed techniques in bold .....	29
Figure 4.1: Main parts of the emg acquisition circuit .....	30
Figure 4.2: Fully-differential amplifier vs standard operational amplifier .....	31
Figure 4.3: PCB of the emg acquisition circuit .....	32
Figure 4.4: A) Hardware block diagram and B) STM32F4 board.....	33
Figure 4.5: SPI master setup and selected configuration .....	35
Figure 4.6: ADCx block and sub blocks.....	36
Figure 4.7: Data composer internal block diagram .....	36
Figure 4.8: Simulink target model .....	37
Figure 4.9: Comparison of the stability performance of different filter types .....	38
Figure 4.10: High order butterworth bandpass filter configuration .....	39
Figure 4.11: Simulink's host model .....	40
Figure 4.12: EMG acquisition system .....	41
Figure 5.1: Target muscles and electrode location .....	42
Figure 5.2: Flowchart of the feature segmentation and feature extraction matlab script.....	44
Figure 5.3: Distribution of the feature matrix.....	45
Figure 5.4: Target matrix layout.....	47
Figure 5.5: Neural network GUI training window .....	48
Figure 5.6: Training results of the neural network.....	49
Figure 5.7: Steps to generate a trained neural network Simulink model .....	49
Figure 5.8: Simulink host model of the NNMCS.....	50

Figure 5.9: EMG acquisition block and internal diagram .....	50
Figure 5.10: Segmentation block and its internal diagram .....	51
Figure 5.11: Trained neural network classifier block and internal models.....	51
Figure 5.12: Hand actuation block and internal diagrams .....	52

## INDEX OF TABLES

Table 3.1: EMG time domain features .....	14
Table 3.2: Frequency and time-frequency domain features.....	16
Table 6.1: Classification accuracy and generalization of training set 5 for <i>trainscg</i> and <i>trainbr</i>	56
Table 6.2: Classification accuracy and generalization of training set 10 for <i>trainscg</i> and <i>trainbr</i> .....	56
Table 6.3: Classification accuracy and generalization of training set 15 for <i>trainscg</i> and <i>trainbr</i> .....	57
Table 6.4: Classification accuracy and generalization of training set 20 for <i>trainscg</i> and <i>trainbr</i> .....	57
Table 6.5: <i>trainscg</i> and <i>trainbr</i> comparison for training set 15 reduced to 5 dimensions varying the number of hidden neurons .....	58
Table 6.6: <i>trainscg</i> and <i>trainbr</i> comparison for training set 15 reduced to 6 dimensions varying the number of hidden neurons .....	59
Table 6.7: NN generalization evaluation for testing sets of different sessions .....	59
Table 6.8: Execution time of each processing step of the nnmcs for one iteration .....	60
Table A.1: Breakdown of work structure and duration .....	69
Table A.2: Gantt chart .....	69
Table A.3: Cost of the material.....	70
Table A.4: Cost of the computing tools.....	70
Table A.5: Labor cost.....	71
Table A.6: Unified total budget .....	71



## ABSTRACT

3D printing has revolutionized the manufacturing process reducing costs and time, but only when combined with robotics and electronics, these structures could develop their full potential. In order to improve the available printable hand designs, a control system based on electromyographic (EMG) signals has been implemented, so that different movement patterns can be recognized and replicated in the bionic hand in real time. This control system has been developed in Matlab/ Simulink comprising EMG signal acquisition, feature extraction, dimensionality reduction and pattern recognition through a trained neural-network. Pattern recognition depends on the features used, their dimensions and the time spent in signal processing. Finding balance between this execution time and the input features of the neural network is a crucial step for an optimal classification.

*KEY WORDS: EMG, control system, neural networks, pattern recognition, real-time, printable robotic hand.*



# CHAPTER 1

## 1. INTRODUCTION

### 1.1. BACKGROUND

Human-assisting robots and rehabilitation have benefited from 3D printing in the last years. The main advantage of 3D printing, is the creation of personalized designs that do not compromise the fabrication cost. However, this improvement in design does not provide or restore any dexterity without the combination of robotics, electronics and control systems. This present study is the second part of a bigger project that comprises both robotic arm manufacturing and control system design.

An efficient human motion tracking system is electromyographic (EMG) that has been widely used in human-machine interfaces since the 1960's. EMG measures the electrical activity of the muscles due to neural activation, and thus, desire of movement can be detected. The principal advantage of EMG control system with respect to other biosignal-based control systems, such as electroencephalography (EEG), is the low invasiveness during signal acquisition.

The prosthesis and human-assisting robots developed so far, focused on high precision control of user's motion. This design allows complex movements with several degrees of freedom, but is expensive and requires the user to concentrate in the task. The cost and difficult control drawbacks, have opened a gap between research and practical world.

The aim of printable robotic limbs is to bring closer both fields, offering accessible and functional devices that adapt to and learn from the costumer. Providing an effective control system that is able to detect and identify user's intentions based on EMG signals to actuate a printable robotic hand, is the main goal of this work.

### 1.2. MOTIVATION

The prosthetic technology developed so far has a severe trade-off between control accuracy and cost. The motivation of this project, including both robotic arm fabrication and control system development, is to provide a low cost alternative for the present commercial prosthesis. To achieve this, new technologies like 3D printing, open source designs or software and hardware developed in the University, have been used to minimize the prize.

Once the prosthesis has been assembled, an interface between the user and the device is needed to actuate it. This study focuses in the implementation of a neural network-based myoelectric control system for the printable robotic hand. However, this control system must be consistent with the global motivation of the project, and thus, adjust to a low budget. In order to fulfill this requirements, the system has been developed using the minimum number of physical components, focusing on the digital domain. Following this idea, several tools like a low cost EMG acquisition circuit developed in the University, Rapid

Control Prototyping (RCP) methodology and their integration in MATLAB/ Simulink processing environment, have been used to implement the neural network classification.

### 1.3.OBJECTIVES

As previously mentioned, the main objective of this study is to develop a control system, based on EMG signals, able to detect and interpret different user hand movements to operate a printable bionic hand. In order to achieve this, the program must fulfill the following aspects:

1. Acquire and process EMG signals.
2. Identify the specific movement underneath the processed EMG data.
3. Be sensitive to changes in position.
4. Real-time actuation of the five-fingered robotic hand.
5. Integration of the different elements: robotic hand, EMG circuit and control system.
6. Low cost

### 1.4. STRUCTURE

The dissertation starts with an overview of the EMG theory and related work done in EMG control systems. EMG theory focuses on the physiology (2.1) and acquisition (2.2) of the signal. In chapter 3, the state of the art of EMG signal processing is analyzed, including data segmentation (3.1), feature extraction (3.2), dimensionality reduction (3.3) and data classification (3.4).

Next chapters describe the materials and methods used in program development. The EMG acquisition system is assessed in chapter 4, describing the employed circuit (4.1), microcontroller (4.2) and acquisition software (4.3). A detail explanation of the control system functioning can be followed step by step through:

- Gesture programming (5.1)
- EMG acquisition method (5.2)
- Segmentation and feature extraction (5.3.1)
- Dimensionality reduction (5.3.2)
- Generation and training of the Neural Network (5.3.3)
- Integration of the control system with the printed robotic hand (5.4)

The thesis concludes with the results of the performed tests (chapter 6) and the conclusions and future work extracted from their analysis (chapter 7).



# CHAPTER 2

## 2. EMG SIGNAL

The aim of this chapter is to give the reader a theoretical background about the biology of the EMG signal, from its generation in the human body to its acquisition.

### 2.1. PHYSIOLOGY

To understand how movement is generated in the body, it is important to know the basic structure responsible for it first. Motor units are the smallest functional modules that explain muscle contraction through neural control. They are formed by a motor neuron and all the skeletal muscle fibers it stimulates. These fibers are intermingled with other motor unit muscle fibers without following any distribution pattern. Motor neurons propagate electrical stimuli (action potentials) from the central nervous system to the muscle fibers. [1][2]

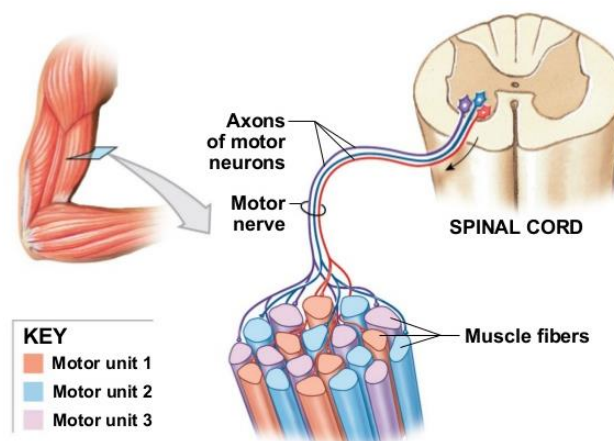


FIGURE 2.1: MOTOR UNIT ARRANGEMENT [3]

If muscle fibers are not activated, there is an ionic equilibrium (resting potential) between the inner and outer spaces, established at -80 to -90 mV. However, when an action potential reaches the axon terminal of a motor neuron, it triggers the release of a neurotransmitter at the neuromuscular junction. This neurotransmitter opens  $\text{Na}^+$  channels, favoring the inflow of these cations, increasing the membrane potential up to +30 mV and thus, causing the depolarization of the membrane. This condition is immediately compensated by  $\text{Na}^+/\text{K}^+$  ion pumps that repolarize the membrane to the original value. The effect of the action potential is muscle contraction, being an electro-mechanical coupled process [1]. The electromagnetic field produced by ion movement, can be detected and recorded with electrodes placed near the muscles.

So far, how a single muscle fiber is stimulated by an action potential has been explained. However, a single motor neuron innervates several muscle fibers that contract in unison. The spatial-temporal summation of all the individual action potentials that reach these muscles, is referred as Motor Unit Action Potential (MAUP) [4]. To maintain muscle

contraction, successive activation of motor units is required. This sequence of MUAP form the Motor Unit Action Potential Train (MUAPT). As previously mentioned, different motor unit fibers assemble muscles. Therefore, movement involves the activation of multiple motor units. The recorded EMG signal is the superposition of MUAPTs corresponding to the activated motor units.

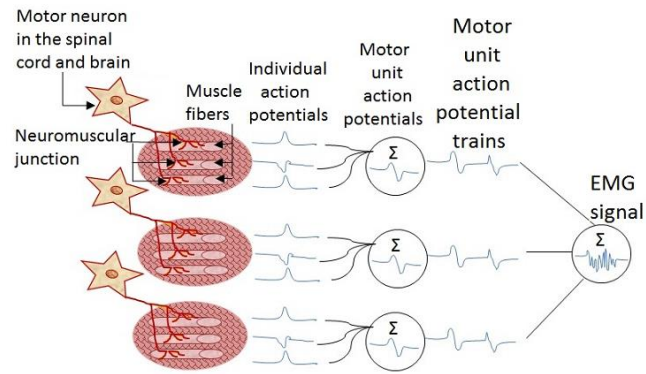


FIGURE 2.2: EMG SIGNAL COMPOSITION [2]

According to Luca [5] the resulting EMG signal has the following properties:

1. Stochastic amplitude nature depicted by a Gaussian distribution.
2. Amplitude range: 0 – 10 mV peak-to-peak or 0-1.5 mV rms
3. Frequency range: 0-500 Hz with maximum energy at the dominant interval 50-150 Hz

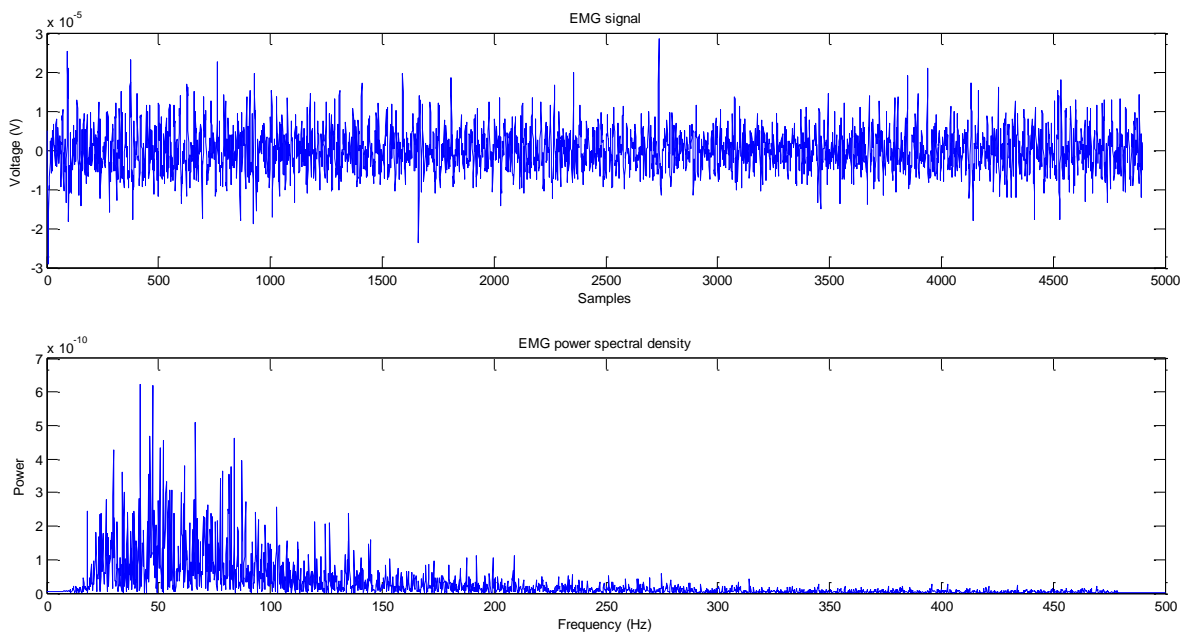


FIGURE 2.3: EMG SIGNAL TIME AND FREQUENCY DOMAINS

## 2.2. ACQUISITION

EMG acquisition is a multistage process comprising target muscle identification, electrode placement and signal preprocessing. Although there are different acquisition modalities, the previously mentioned steps are described focusing on noninvasive forearm-muscle EMG acquisition techniques.

### 2.2.1. TARGET MUSCLES

Most of the muscles responsible for hand movement are placed in the forearm [6][7]. For the robotic hand programmed actions: open/close hand, extended index, pincer, supination/ pronation and a rest position; the muscles of interest are those that contribute more to digits, thumb and wrist movement.

This work has focused on forearm superficial muscles because their accessibility allows noninvasive recordings by placing two electrodes longitudinally on the skin covering each target muscle.

Therefore, the EMG signals have been obtained from:

1. Flexor digitorum superficialis: Superficial anterior muscle in charge of phalanx bending.
2. Flexor carpi ulnaris: superficial anterior muscle responsible for hand flexing.
3. Brachioradialis: superficial anterior muscle that supinates/pronates the forearm.
4. Extensor digitorum: Superficial posterior muscle that extends the phalanxes.

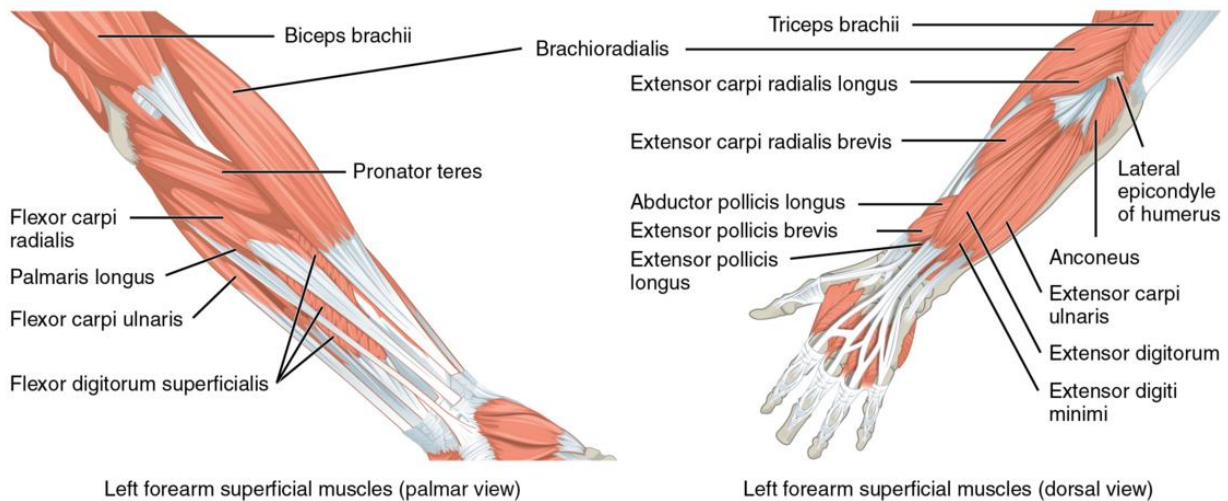


FIGURE 2.4: FOREARM SUPERFICIAL MUSCLES ANTERIOR AND POSTERIOR VIEW [7]

### 2.2.2. ELECTRODES

Electrodes are transducers that convert the ion current generated by contracting muscles into electric current. Depending on the invasiveness, there are two types of electrodes: surface or intramuscular (fine-wire or needle electrodes) [1],[2]. Intramuscular electrodes provide a high-resolution recording of the precise target muscle, but are relatively painful so surface disposable electrodes have been used for EMG acquisition.

#### 2.2.2.1. SKIN PREPARATION

Surface electrodes require special skin preparation to create a stable contact with low skin impedance. For dry skin, the impedance ranges from several kilo to megaohms due to movement artifacts, sweat or body fat percentage. These artifacts distort the signal so the following arrangements should be taken into account before placing the electrodes [2]:

1. Hair removal: This favors electron adhesion.
2. Skin cleaning: Dead skin cells, dirt and sweat that might be present at the recording site, increase the impedance. To remove them, special abrasive, fine sand paper or alcohol can be applied to the surface. Harm should be avoided, just a light red color of the treated skin indicates good impedance.

#### 2.2.2.2. ELECTRODE TYPES

Current transduction is produced by an oxidation-reduction chemical reaction inside the electrode. Depending on the material used, there are two types of electrodes [8]:

1. Gelled Electrodes: The most common material for the metallic part of the electrode is Silver-Silver Chloride (Ag-AgCl). These electrodes improve their performance with an electrolytic gel between the skin and the metal electrode by increasing the conductivity. Their main advantages are low impedance, good adhesion and cheap cost. They are usually disposable electrodes.
2. Dry electrodes: These type of electrodes do not require a gel interface at the skin-electrode junction. The preferred material is gold (Au), despite its high cost, do to their very low impedance. They are very resistant and can be reused.

In this study the electrodes that have been used are Ag-AgCl ECG surface electrodes, model FSTC1 from Skintact, due to their multifunction applications and low cost.

#### 2.2.2.3. ELECTRODE PLACEMENT

According to Luca, De [5] electrodes should be placed in the belly of the target muscle, along the longitudinal middle line of the muscle. This arrangement maximizes the number of muscle fibers that can be recorded with the electrodes.

It is important to avoid tendon insertions because muscle fibers become thinner and closer, reducing signal amplitude and making it more susceptible to crosstalk or non-target muscle activity interferences. Motor points or innervation zones, should be avoided too. These areas have the highest neural density so minimal electrical activity causes a perceptible twitch on the muscle surface, altering the stability.

An extra reference electrode should be located on electrically neutral tissue, like a bony prominence, nearby the target area.

In this case, two Ag-AgCl electrodes have been placed on top of the belly of the mentioned target muscles to measure the differential signal, and a reference electrode, in the elbow.

### 2.2.3. EMG ELECTRONICS AND PREPROCESSING

Raw EMG signals range from several  $\mu\text{V}$  to  $\text{mV}$  and are subjected to various noise sources, therefore they need to be preprocessed (amplification and filtering) so that a reliable analysis could be obtained. The main issues in EMG acquisition circuits are [5]:

1. Differential amplification. The idea of differential amplification is to record EMG signals at two close sites, subtract both signals and amplify the difference. Powerline noise or cross-talk cannot be filtered as they contribute to the dominant energy range of the EMG signal. However, differential amplification cancels this common noise while amplifies the changes in the superficial muscles. The accuracy of this noise elimination is measured by the Common Mode Rejection Ratio, being 90dB the recommended value. The reference electrode is the common reference for the differential inputs.

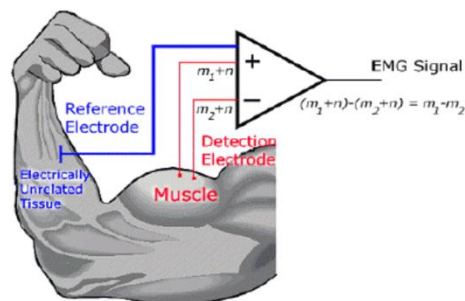


FIGURE 2.5: DIFFERENTIAL AMPLIFIER CONFIGURATION [3]

2. High input impedance of the differential amplifier. Input impedance of the pre-amplifier should be at least 10 times the electrode-skin impedance to avoid signal attenuation and distortion. Impedances of both inputs must be balanced.
3. Band pass filtering. The Signal to Noise Ratio (SNR) can be increased by filtering the range 20-500 Hz with a roll-off of 40dB/dec.
4. Electrical safety concerns. As the subject is directly connected to the circuit through the electrodes, failure of any electrical component may produce harmful current passing through the skin into the tissues. To avoid this and

guarantee user's safety, the subject must be isolated from power source associated connections via optical isolators or transformers. However, damaging current hazard is reduced for low-voltage powered circuits (3-15V).

This linear preprocessing stage aims to preserve maximum amount of information, reduce noise contamination and avoid any signal distortion. After this process, the EMG signal can be processed to extract relevant features for pattern recognition.

# CHAPTER 3

## 3. EMG SIGNAL PROCESSING AND CLASSIFICATION

The advances in robotics and prosthetics have developed more complex artificial limbs with several degrees of freedom. In the past, basic estimation of the amplitude and rate of change of the EMG signal was sufficient to control one function. However, new devices require more intricate control systems that aim to provide [2], [9]:

- Accuracy  
It is essential to reproduce user’s intentions faithfully. To improve it more information should be extracted from the EMG signal by increasing the number of channels of the recording and by building a feature set that makes the most of the obtained data.
- Intuitive interface  
The main problem of the low acceptance rate of myoelectric control systems, is the difference between subject’s instinctive motion commands and the required knowledge to actually perform that action. The solution is either training the user to gain the needed control skills, or develop intelligent user interfaces able to learn muscle activation patterns during changing operation conditions.
- Short time response  
The time response of the control system is limited to real-time constrains, so that the user does not perceive a delay during manipulation. To achieve this, processing the input signals in short segments for further analysis is the most efficient approach.

From these objectives, a classification of the EMG control systems can be derived into: patter recognition- and non-pattern recognition-based. In the last type, the mentioned old-fashioned control systems based on thresholding are included. This section will focus on pattern recognition-based approach in which several functions can be identified from signal patterns by classifiers.

Figure 3.1 depicts the workflow of pattern recognition based-control systems is divided in: data segmentation, feature extraction, dimensionality reduction, classification and controller. Each step will be discussed in the next subsections.



FIGURE 3.1: STAGES FOR DEVELOPING MYOELECTRIC CONTROL SYSTEMS

### 3.1. DATA SEGMENTATION

Data segmentation is the process of dividing input signals into time slots for further processing. There are several segmentation approaches based on window length, signal features in the window or signal state.

One of the main concerns of control systems is a fast time response. Real-time considerations have established a maximum lag of 300 ms between muscle contraction onset and the corresponding device movement. The larger the segment, the smaller bias and feature variance, but the higher the computational load. Thus, segment length must be chosen carefully so that all processing steps can be computed within that 300 ms interval.

According to Oskoei and Hu [10], there are two main windowing techniques: adjacent or disjoint segmentation and overlapped segmentation. Disjoint windowing segments data by a predefined length, whereas in overlapping windowing, segments slide over each other with an increment greater than the processing time and shorter than the segment length, as shown in Figure 3.2. Results showed that disjoint segmentation with length of 200 ms, yields in high EMG classification performance and time response for real-time applications. Overlapped segments of 200 ms with an increment of 50 ms provide a semi-redundant classification that also reduce time response without a significant lost in accuracy.

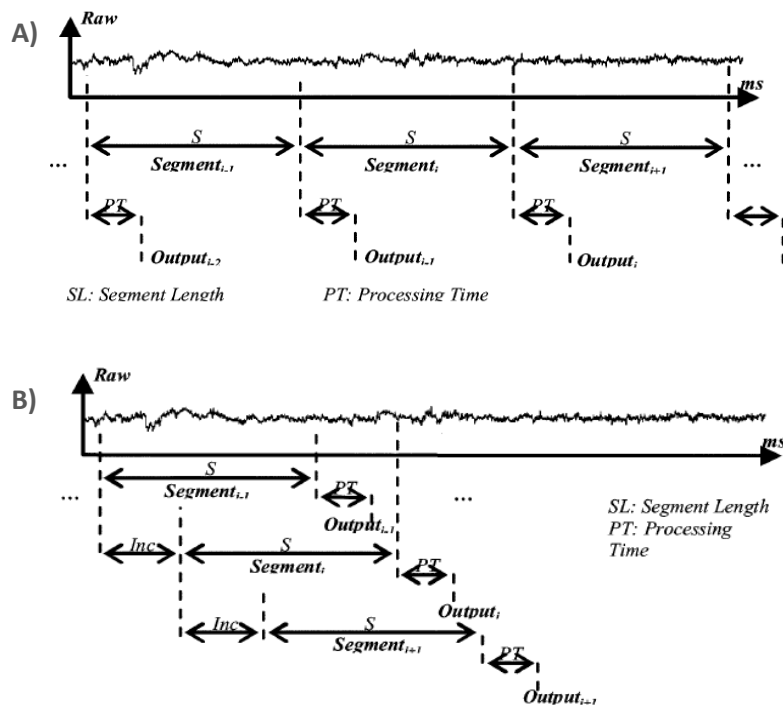


FIGURE 3.2: A) DISJOINT SEGMENTATION AND B) OVERLAPPED SEGMENTATION [10]

However, not all segmentation procedures are based on length itself; other approaches involve thresholding. Christodoulou and Pattichis [11] developed a segmentation algorithm, applied over a fixed length window, to calculate a threshold based on maximum and mean maximum value of the whole signal. Peaks above the threshold are considered candidate segments. On the other hand, Gut and Moschytz [12] employed thresholding over a sliding



window to detect onset and offset depending on the mean slope and total variation of the segment.

Even more complex segmentation techniques like MAUPs peak identification, determining MAUP's beginning and ending extraction points or Daubechies Wavelet Transform have been analyzed by Kaur *et al.* [13]. These methods process the data for diagnosis of neuromuscular disorders, rather than for control systems, by detecting and classifying different MAUPs shapes. MAUPs peak identification by thresholding showed the best delineation and classification performance (95.90%).

Apart from segment length and signal characteristic-based windowing, another important subject in data segmentation is the state of the signal. EMG signals can be divided into a transient state and steady state [14]. Movement initiation from rest is considered as transient state, whereas steady state is associated with maintained muscle contractions during the action.

Transient states used to be considered more significant for pattern recognition based-control systems as they show user's intention of movement. Hudgins *et al.* [15] employed windows of 100 ms after muscle onset to successfully classify muscle functions. The major drawback of this approach is its inability to switch classes as muscle contraction must be initiated from rest.

Englehart *et al.* [16] proved that steady-state signal classification is more accurate and less sensible to short segment length degradation. Although class transitions are allowed with this method, most errors occur during these periods due to an undefined intermediate state between classes. To solve this, detection and elimination of transition states is a promising technique to create a reliable training data set.

### 3.2. FEATURE EXTRACTION

Feeding a classifier directly with the segmented data is still inefficient. Feature extraction is a technique that transforms raw input data into a reduced set of features that highlight the encoded relevant information. Success in pattern recognition depends almost exclusively in feature selection and extraction.

As claimed by Zecca *et al.* [17] features can be classified as: time domain, frequency (spectral) domain and time-frequency (time-scale) domain.

EMG signals should be considered non-stationary due to asynchronous MAUPs summation, even during constant contractions where there is no voluntary change. However, for short-time low-level contractions (20-30% of maximal voluntary contraction, MVC; during 20-40s), the signal can be assumed to be wide-sense stationary. This presumption can be held for higher contraction levels (50-80% of MVC) during shorter periods (0.5-1.5s) to avoid muscle fatigue [18]. Therefore, as stated by Oskoei and Hu [14] the EMG signal can be considered stationary in real-time applications. These concepts are significant to assess the main limitations of each feature type.

### 3.2.1. TIME DOMAIN

Time domain features have been widely used in research do to its easy and quick calculation. These features are based on signal amplitude, which can be considered as the time-varying standard deviation (STD) of the signal, proportional to the number of active MUs and their firing rate. The computed features provide information about waveform energy, amplitude, duration and frequency. Time domain features are shown in Table 3.1.

**TABLE 3.1: EMG TIME DOMAIN FEATURES [19]**

Time domain features			
Integrated EMG (IEMG)	$IEMG_k = \sum_{i=1}^N  x_i $	Root Mean Square (RMS)	$RMS_k = \sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2}$
Mean Absolute Value (MAV)	$MAV_k = \frac{1}{N} \sum_{i=1}^N  x_i $	Zero Crossings (ZC)	ZC is incremented if $\{x_i > 0 \text{ and } x_{i+1} < 0\}$ or $\{x_i < 0 \text{ and } x_{i+1} > 0\}$ and $ x_i - x_{i-1}  \geq \epsilon$
Modified Mean Absolute Value 1 (MAV1)	$MMAV1_k = \frac{1}{N} \sum_{i=1}^N w_i  x_i $ $w(i) = \begin{cases} 1, & 0.25N \leq i \leq 0.75N \\ 0.5, & \text{otherwise} \end{cases}$	Slope Sign Changes (SSC)	SSC incremented if $\{x_i > x_{i-1} \text{ and } x_i > x_{i+1}\}$ or $\{x_i < x_{i-1} \text{ and } x_i < x_{i+1}\}$ and $ x_i - x_{i+1}  \geq \epsilon$ or $ x_i - x_{i-1}  \geq \epsilon$
Modified Mean Absolute Value 2 (MAV2)	$MMAV2_k = \frac{1}{N} \sum_{i=1}^N w_i  x_i $ $w(i) = \begin{cases} 1, & 0.25N \leq i \leq 0.75N \\ 4i/N, & 0.25N > i \\ 4(i-N)/N, & 0.75N < i \end{cases}$	Willison Amplitude (WAMP)	$WAMP_k = \sum_{i=1}^{N-1} f( x_i - x_{i+1} )$ $f(x) = \begin{cases} 1, & x > \epsilon \\ 0, & \text{otherwise} \end{cases}$
Mean Absolute Value Slope (MAVS)	$MAVS_k = MAV_{k+1} - MAV_k$	Simple Square Integral (SSI)	$SSI_k = \sum_{i=1}^N ( x_i ^2)$
Variance (VAR)	$VAR_k = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$	Histogram of EMG (HEMG)	HEMG divides the elements in EMG signal into b equally spaced segments and returns the number of elements in each segment
Waveform Length (WL)	$WL_k = \sum_{i=1}^{N-1}  x_{i+1} - x_i $		
Variables of the time domain features $x_i$ : value of each part of segment k N : length of the segment $\bar{x}$ : mean value of segment k $\epsilon$ : a threshold (not the same for all equations)			

### 3.2.2. FREQUENCY DOMAIN

Frequency domain features are related to the estimated power spectrum density (PSD) of the signal, defined for wide-sense stochastic signals, as the Fourier transform of the autocorrelation function. Primary information about the spectrum and its change in time is represented by frequency mean (FMN) and frequency median (FMD).

Frequency domain features can be computed by periodogram or parametric methods. In the former method, PSD is calculated by the square of the Fourier transform divided by the signal length. According to Oskoei and Hu [14] the weaknesses of periodogram are: 1) pre-windowing frequency leakage, 2) frequency resolution of short-time segments, 3) large estimation variance and 4) periodicity assumption outside the analyzed segment. On the other hand, in parametric methods, autoregressive coefficients (AR) are used to estimate PSD. The problems of this procedure are order determination and unwanted modelled noise.

The main disadvantage of frequency domain features compared to time domain features is the higher computational load.

### 3.2.3. TIME-FREQUENCY DOMAIN

Previous features were computed under the assumption that EMG signals are stationary. Nevertheless, these signals are composed of several non-stationary or transitory features. Therefore, spectral analysis lose information in time domain as it cannot detect when a specific event occurs. Providing energy measurements in both time and frequency domains, describes more precisely the physical phenomenon, but requires heavy computation.

The most studied time-frequency features are: Short-time Fourier Transform (STFT), Wavelet Transformation (WT) and Wavelet Packet Transformation. Each method divides differently the time-frequency plane. STFT has a fixed partitioning ratio in which each cell has the same aspect ratio. WT has a variable tiling with an aspect ratio that changes frequency resolution proportionally to the center frequency. Although the partitioning pattern is not constant, it is still fixed. Finally, WPT provides an adaptive tiling that selects the optimal partition pattern among a given set, depending on the application.

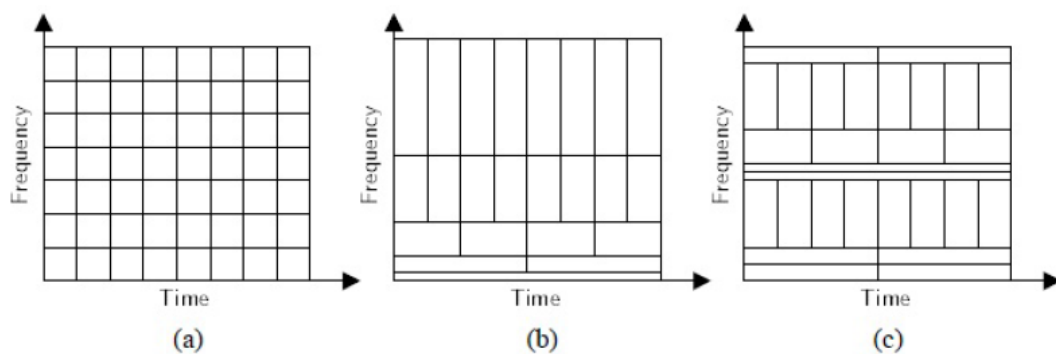


FIGURE 3.3: TIME-FREQUENCY TILING OF A) STFT, B) WT AND C) WPT [20]

Table 3.2 summarizes frequency and time-frequency features.

TABLE 3.2: FREQUENCY AND TIME-FREQUENCY DOMAIN FEATURES [19]

Frequency domain features		Time-frequency domain features	
Autoregressive Coefficients (AR)	$x_k = \sum_{i=1}^N a_i x_{k-i} + e_k$	Short Time Fourier Transform (STFT)	$STFT_x(t, w) = \int W^*(\tau - t)x(\tau)e^{-j\omega t} d\tau$
Frequency Median (FMD)	$F_{MD} = \frac{1}{2} \sum_{i=1}^M PSD_i$	Wavelet Transform (WT)	$W_x(a, b) = \int x(t) \left(\frac{1}{\sqrt{a}}\right) \psi^* \left(\frac{t-b}{a}\right) dt$
Frequency Mean (FMN)	$F_{MN} = \frac{\sum_{i=1}^M f_i PSD_i}{\sum_{i=1}^M PSD_i}$ $f_i = \frac{i \cdot sampling\ rate}{2M}$	Wavelet Packet Transform (WPT)	WPT is a generalized version of the continuous wavelet transform and the discrete wavelet transform. The basis of the WPT is chosen using an entropy-based cost function
Modified Frequency Median (MFMD)	$MFMD = \frac{1}{2} \sum_{j=1}^M A_j$		
Modified Frequency Mean (MFMN)	$MFMN = \frac{\sum_{j=1}^M f_j A_j}{\sum_{j=1}^M A_j}$		
Frequency Ratio (FR)	$FR_j = \frac{ F(\cdot) _{j\ lowfreq}}{ F(\cdot) _{j\ highfreq}}$		
<u>Variables of frequency domain features:</u>  $a_i$ : AR coefficients $e_i$ : White noise or error sequence $M$ : length of the power spectrum density $PSD_i$ : $i$ th line of the power spectrum density $A_j$ : EMG amplitude spectrum at frequency bin $j$ $f_i$ : frequency of the spectrum at frequency bin $j$ $ F(\cdot) _j$ : Fast Fourier transform of EMG signal in channel $j$ . <i>lowfreq</i> is the low frequency band and <i>highfreq</i> is the high frequency band		<u>Variables of time-frequency domain features:</u>  $W(t)$ : Window function $*$ : Complex conjugate $\tau$ : Time $w$ : Frequency $x(t)$ : Function of the input signal $\psi^*$ : Complex conjugate of the mother wavelet function $\psi^* \left(\frac{t-b}{a}\right)$ : Shifted and scaled version of the wavelet at time $b$ and scale $a$	

Several studies have been conducted to compare combinations of time and frequency domain features and test their performance in pattern recognition. Phinyomark *et al.* [21] examined the tolerance to white Gaussian noise of time domain (IEMG, MAV, MMAV1, MMAV2, MAVS, SSI, VAR, RMS, WL, ZC, SSC, WAMP and HEMG) and frequency domain features (AR, FMN, FMD, MFMD and MFMN). The latest proved to be more robust and tolerant to noise. Also, MFMN, WAMP and HEMG were tested as pattern recognition inputs, resulting in excellent candidates for multi-source feature vector.

Most research used special techniques to test the efficiency of each feature set. For instance, Huang and Chen [22] evaluated time and frequency domain features with Davies Boulding index that measures the overlapping degree between nearest neighboring clusters. Features with highest cluster separability (VAR, IEMG and WL in this case) contain more relevant information, and were fed with other features into a classifier for pattern recognition analysis. Two sets were examined: IEMG, VAR, WL and WAMP against IEMG, VAR, WL, WAMP, BZ and AR. The last combination showed better performance in pattern recognition.

### 3.3. DIMENSIONALITY REDUCTION

The previous example introduced the Bellman's so called "curse of dimensionality" [23]. Computational efficiency is essential for real-time applications, but the number of calculations and training data required for pattern recognition, rise exponentially with the increase of feature vector dimensions.

Dimensionality reduction, also called phenomenological analysis, aims to improve signal classification, reducing the dimensions of the feature vector by removing redundant information without disrupting the relevant data. According to [2], [17] and [19], there are two main strategies: feature subset selection (or just feature selection) and feature projection.

#### 3.3.1. FEATURE SUBSET SELECTION (FSS)

Feature Subset Selection (FSS) seeks for the optimal subset, with highest classification rate, among all existing features. This approach requires a search strategy to select candidate subsets, and an objective function that test them and provide feedback to new candidate selection. Depending on the evaluation process, objective functions can be classified into *filters* and *wrappers*. The former assess the candidate subset based on their information content (interclass distance, statistical dependence or information-theoretic measures); whereas the latest are classifiers that evaluate the candidates according to their classification accuracy.

Oskoei and Hu [24] developed a complex algorithm based on FSS. For this study, the selected search strategy was Genetic Algorithm that simulates "survival of the fittest" evolutionary process. Both classes of objective functions were analyzed in the research: Davies Boulding index and Fisher Linear Discriminant Analysis (represent clusters' dispersion comparing to their scatter) as filters, and Linear Discriminant Analysis (LDA) as wrapper. LDA employs a discriminant function to divide the feature space into different labeled subspaces by the hyperplane decision surface, using a training data set. More details about LDA are provided in the next subsection. Time domain (MAV, MAVS, RMS, VAR, WL, ZC, SSC and WAMP) and frequency domain features (AR, FMN, FMD and FR) were analyzed by FSS and then classified by an Artificial Neural Network. WL showed the highest discriminating information for classification, followed by MAV and RMS, and AR in the third place. However, all this process requires complex calculations that increase the computation time. This approach was therefore presented, as a pre-offline training of the classifier to upgrade its performance.

### 3.3.2. FEATURE PROJECTION

Feature projection creates a new and smaller feature set by identifying the optimal combination of the original features. The simplest method employs a linear transformation matrix to map the original features into the reduced-dimension space. The most popular linear mapping functions are Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA).

#### 3.3.2.1. PRINCIPAL COMPONENT ANALYSIS (PCA)

PCA, which is also known as Karhunen-Loève transform or Singular Value Decomposition [17] and [25], creates an uncorrelated reduced-dimension set of variables from the original ones. To achieve this, the linear transformation matrix is calculated by searching for the directions with higher variations. This implies the consideration of those features that provide more variation, so low-order principal components are kept while the higher-order ones are discarded. Thus, the main limitation of this method is the preliminary assumption that maximum variance directions provide maximum discrimination, as no class information like between-class or within-class scatter is taken into account. In addition, PCA is scale-sensitive and some principal components might be obscured by elements with higher variances.

Englehart *et al.* [9] compared feature projection using PCA, and Euclidean distance class separability (CS) as FSS. The tested features were Hudgins' time domain features (TD): MAV, MAVS, ZC, SSC and WL; and time-frequency domain features: STFT, WT and WPT. After reducing the dimensions of the previous feature sets, classification performance was evaluated by LDA classifier and Multi-Layer Perceptron (MLP) classifier. Results show that for dimensionality reduction PCA is more effective than CS, especially for time-frequency domain features.

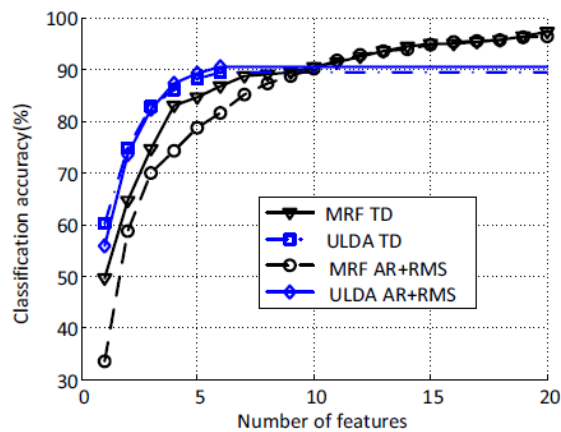
#### 3.3.2.2. LINEAR DISCRIMINANT ANALYSIS (LDA)

LDA mapping is based on a classification criterion that maximizes the ratio of between-class and within-class variance. As previously seen, LDA can be used as a wrapper objective function (FSS) or as a classifier itself, however in this subsection, the focus will be its application in dimensionality reduction.

According to Wang [23] LDA is a twofold process that first defines the discriminant functions and then solves the criterion function. Discriminant functions are a series of input-output functions generated by the classifier, which relate the input features to the parameter set of the class. In dimensionality reduction, this is a prior estimation of the feasible reduced features, calculated from a training data set. In the second step, these discriminant functions are introduced into the criterion function that is based on between-class and within-class covariance. Depending on the decision rule implied in the criterion function, the optimal solution will minimize (Generalized LDA) or maximize (Fisher's linear discriminants) the function.

Liu [26] performed a comparative analysis between feature selection and feature projection methods for myoelectric pattern recognition applications. Two feature sets were tested: time domain (TD) features including MAV, ZC, WL and SSC; and the combination of AR and RMS (AR+RMS). Dimensionality reduction was carried out by PCA and ULDA (Uncorrelated Linear Discrimination Analysis) as feature projection methods; whereas feature selection

approach employed two filter objective functions: Markov Random Field (MRF) and Forward Orthogonal Search algorithm (FOS-MOD). MRF selects the candidate subset that maximizes between-class distance while minimizing the within-class distance; whereas, FOS-MOD maximizes the overall dependency to identify relevant features. Results show that ULDA is more efficient than PCA in dimensionality reduction as the projected features condense better the relevant information. On the other hand, for feature selection performance, MRF and FOS-MOD achieve similar classification accuracy. When comparing feature selection (MRF) against feature projection (ULDA), MRF reaches higher accuracies as the number of selected features increases. However, for low feature sets (up to eight) ULDA projected features provide more significant information yielding in better classification accuracy than MRF original feature set, as depicted in Figure 3.4. Finally the study concludes that dimensionality reduction is able to preserve classification performance while decreasing the number of features. This reduces the computational load and processing time of the classifier, allowing real-time applications.



**FIGURE 3.4: COMPARISON OF THE CLASSIFICATION PERFORMANCE OF MRF SELECTION AND ULDA TRANSFORMATION [26]**

### 3.4. CLASSIFICATION

Once the acquired data have been properly processed to enhance relevant information while minimizing the redundant one, the next step in pattern recognition is classification. According to Oskoei and Hu [14], the ideal classifier should fulfill the following requirements:

- Be able to deal with varying patterns optimally (efficient classification of novel patterns).
- Prevent over fitting (noise or random error modelling)
- Fast classification to meet real-time constrains

There are several techniques to categorize the reduced feature vector: Bayesian Classifier (BC), Neural Networks (NN), Fuzzy Inference Systems (FIS), Support Vector Machines (SVM), Hidden Markov Models (HMM) and K-Nearest Neighbor (KNN). Next subsections describe in detail each classification procedure.



### 3.4.1. BAYESIAN CLASSIFIER (BC)

Bayesian Classifiers (BC) is a standard statistical classification method that applies Bayes' to sort input data based on a given hypothesis. Therefore, training data creates a probabilistic model to classify each input with a probability associated for each feasible class [27]:

$$P(c|D) = \frac{P(D|c)P(c)}{P(D)}$$

where  $P(c)$  is the prior probability that the input data corresponds to that class,  $P(D)$  is the prior probability that the training data  $D$  will be observed (independently of the class hypothesis that is being analyzed), and  $P(D|c)$  is the probability of observing data  $D$  assuming that the hypothesis that  $D$  belongs to class  $c$ , holds.  $P(c|D)$  is the posterior probability or confidence classification level that relates input  $D$  to class  $c$ .

Bu *et al.* [28] proposed a Bayesian Network (BN) for motion prediction using EMG signals. The presented model comprised BC to forecast motions, in parallel with a probabilistic NN called Log-Linearized Gaussian Mixture Network (LLGMN) to calculate the probability density functions of the input signals. In BC, motion prediction hypothesis was based on hand position and previous motion. The final command was determined combining both BC and LLGMN outputs. To demonstrate the feasibility of this technique, upper limb motions were analyzed during a cooking task yielding in classification rates of 85.1% for LLGMN alone, and 92.9% for the proposed BN.

### 3.4.2. NEURAL NETWORKS (NN)

Neural Networks (NN) are inspired in neurons' ability to collect, process and disseminate action potentials. NN are composed of artificial neurons, information-processing units of a set of inputs and weights.

In the first step of artificial neuron processing, the net is calculated by the arithmetic sum of the inputs and weights, as depicted in Figure 3.5, where  $n$  is the total number of input units,  $x_i$  is the input of one unit and  $w_i$ , its associated weight. The next step evaluates the obtained net with an activation function ( $f_{act}$ ). This function works as a logic gate that is activated (output value near +1) or inactivated (output value near 0) depending on the input net value. This activation must be nonlinear being the proposed activation functions: threshold linear, step, arbitrary step, exponential or sigmoid [29].

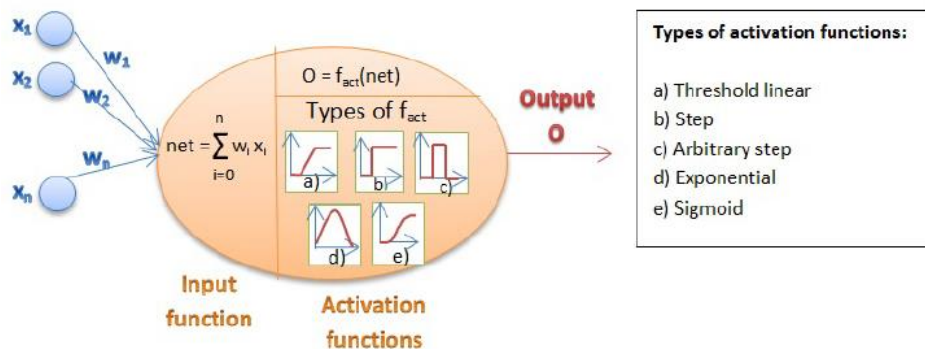


FIGURE 3.5: ARTIFICIAL NEURON MODEL [2]



The most common NN is the Multilayer Perceptron (MLP) formed by several input units, several hidden layers (usually one or two) and a level of output units, as represented in Figure 3.6. Each input unit is connected to each unit in the hidden layer with an associated weight. Similarly, that hidden layer is connected to the artificial neurons of the following hidden or output layer. This layout corresponds to a fully connected network, but there are also partially connected networks.

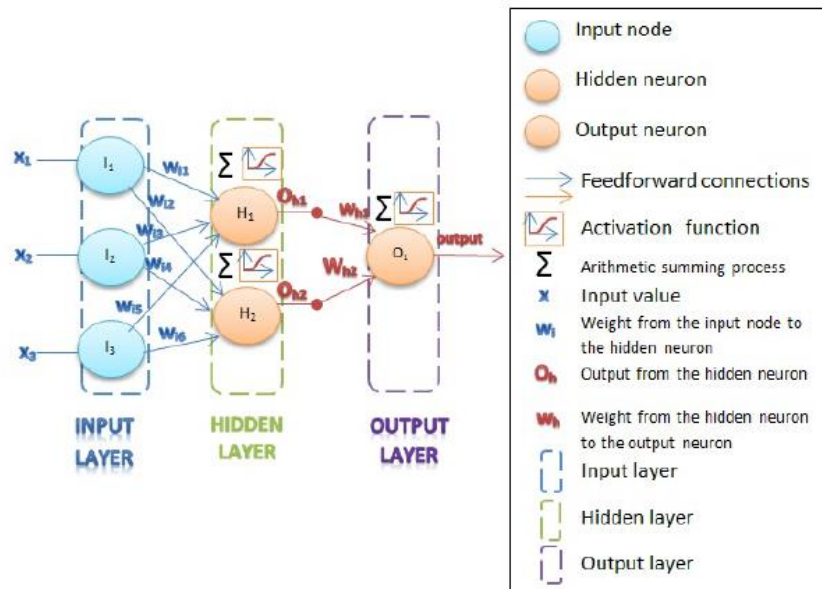


FIGURE 3.6: FULLY CONNECTED MLP WITH ONE INPUT LAYER, ONE HIDDEN AND ONE OUTPUT LAYER [2]

According to [17] and [29], the advantages of NN are based on artificial intelligence learning skills, being able to:

- Learn both linear and non-linear relationships directly from the input data without an a priori mathematical model.
- Adapt to changing conditions.
- Process corrupted or partial data sets.

Nevertheless, its learning performance is biased by the provided number of examples.

This workflow implies a training step to adjust network weights. This introduces the concept of back-propagation neural networks (BPNN) in which each unit or neuron receives feedback from the following layer to adapt their weights and improve the classification performance; see Figure 3.7. Units might even have self-connections. This recurrence supplies information about past events working as a short-term memory.

Ahsan *et al.* [30] proposed an optimized design of artificial NN to classify four predefined hand motions (left, right, up and down) using a back-propagation algorithm. The foremost BPNN employed 7 inputs, 10 neurons in the hidden layer and 4 outputs along with a back propagation training algorithm called Levenberg-Marquardt (LM) that is used to solve non-linear least squares problems [31]. To evaluate NN classification performance 70% of the input data was employed for training, 15% for validation and 15% for testing. The designed NN structure obtained an average classification rate during training of 88.4%.

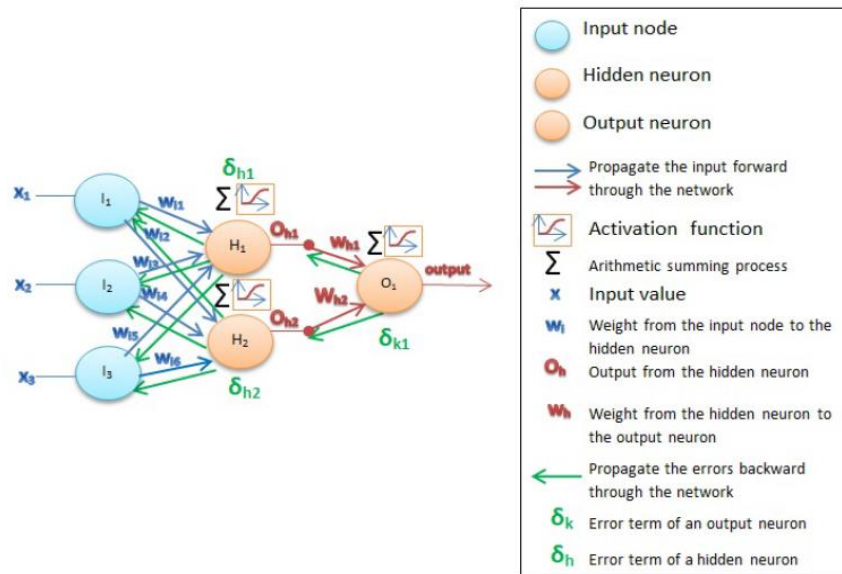


FIGURE 3.7: BACK-PROPAGATION NEURAL NETWORK (BPNN) [2]

There are two main learning paradigms [29]: supervised or unsupervised learning. In supervised learning, the user provides a series of input-output examples that the BPNN uses to compare the output signal to the desired or target output. Therefore, this approach propagates the input forward through the network, calculates the error of each step until a termination condition is satisfied and finally propagates the calculated errors backward to optimize the model. On the other hand, in unsupervised learning, there is no target output so the NN must extract statistical information from the input and develop new classes automatically.

Subasi *et al.* [32] designed a Wavelet Neural Network (WNN) to classify neuromuscular disorders based on EMG signals. AR coefficients were calculated from the input data representing the target classes: normal, myopathic and neurogenic disorder. The proposed WNN design was built by a mono-hidden-layer forward neural network whose node activation function was based on dyadic discrete Morlet wavelet basic function. In order to test the implemented WNN classification performance, it was compared with a BPNN composed by an AR input layer, a fifty-neuron hidden layer and a three-neuron output layer. Results showed a success rate of 90.7% for WNN and 88% for BPNN.

Chu *et al.* [33] presented a real-time EMG pattern recognition system based on linear-nonlinear feature projection for a multifunction myoelectric hand, using MLP classification. Nine hand motions were recorded with a four channel EMG from the forearm. The proposed method processed the data by extracting WPT features, reduced feature dimensions with PCA and applied Self-Organizing Feature Map (SOFM) to transform the PCA-reduced feature set into a new feature space with higher class separability. Finally data was classified with a MLP composed of: a) an input layer built from the eight outputs of the SOFM for the four EMG channels, b) two nine-neuron hidden layers and b) a nine-neuron output layer. Experimental results proved the suitability of this technique for real-time applications with an overall processing time (including virtual hand control) of 125ms. The obtained MLP average classification success rates were 95.795% when only PCA was applied, 97.024% for PCA+SOFM combination and 97.785% for SOFM algorithm alone.

### 3.4.3. FUZZY INFERENCE SYSTEMS (FIS)

Fuzzy logic presents several advantages in bio-signal processing as it is able to handle uncertain, imprecise and even contradictory data; detect complex hidden patterns and allow experience and empirical information integration in the system to improve classification performance [14]. Fuzzy inference systems (FIS) are excellent simulators of human decision-making process.

As explained by Sivanandam *et al.* [34], fuzzy logic works with fuzzy sets or series of linguistic variables that model problems' uncertainty (i.e. qualitatively: low, medium, high; quantitatively: few, several, many; etc.). Therefore numerical inputs must be transformed into these fuzzy values by a membership function, whose values range from the continuous interval [0, 1] following a trapezoidal, triangular or Gaussian waveform. This process is called fuzzification. Once in the fuzzy domain, classification is performed according to a predefined set of IF THEN fuzzy rules. The output of this step is another fuzzy variable that needs to be converted into a crisp (not fuzzy) quantity again by defuzzification.

Thus FIS comprises fuzzification interface, membership functions database, fuzzy rules base, decision-making unit and a defuzzification interface. See Figure 3.8.

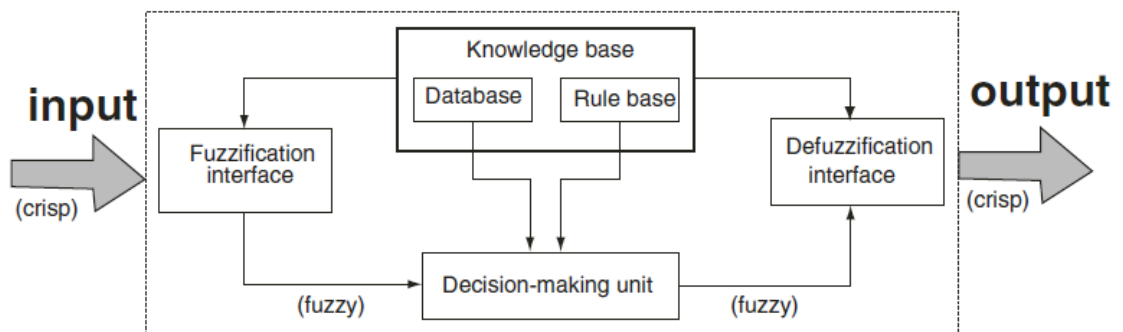


FIGURE 3.8: FUZZY INTERFACE SYSTEM [34]

Ajiboye and Weir [35] designed a pattern recognition algorithm based on heuristic fuzzy logic for multifunctional prosthesis EMG control. The multiinput-single-output fuzzy system included: 1) Input membership functions based on the mean and standard deviation of the signal to fuzzify the input into OFF, LOW, MED and HIGH; 2) an inference rule base to classify the data by processing the linguistic inputs and 3) a membership function to defuzzify the inference rule base linguistic outputs. In addition, a fuzzy c-mean clustering method was employed for data reduction, and inference rule base generation so that clustering performance was optimized. The obtained overall classification rates ranged from 94% to 99%.

### 3.4.4. LINEAR DISCRIMINANT ANALYSIS (LDA)

As it has been previously explained, Linear Discriminant Analysis (LDA) is an algorithm able to reduce feature dimensions and classify them. Xiong and Cherkassky [36] reported that LDA's primary attractiveness is its ability to interpret "global" characteristics of the data to predict the decision boundary. LDA classification searches

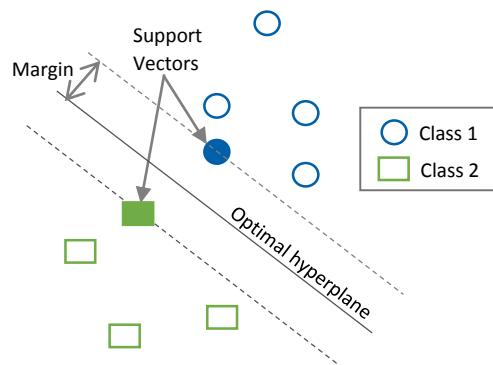
for linear combinations that maximize the distance between classes while minimizing the within-class variance. Hence, LDA does not employ any predefined assumption of the class distributions, rather estimates them from the data.

Balakrishnama and Ganapathiraju [37] described two mapping transformations: *Class-dependent transformation* that maximizes the ratio of between-class variance to within-class variance; and *class-independent transformation* which maximizes the ratio of the overall variance to within-class variance.

Phinyomark *et al.* [38] studied the behavior of fifty time-domain and frequency-domain features to improve myoelectric pattern recognition robustness for ten upper limb motions. Feature classification was performed with LDA, obtaining 93.37% average classification accuracy with sample entropy, and reaching 98.87% when increasing the number of features including sample entropy, fourth order cepstrum coefficients, RMS and WL.

### 3.4.5. SUPPORT VECTOR MACHINE (SMV) CLASSIFIERS

Support Vector Machine (SVM) is a kernel approach that builds an optimal separating hyperplane that maximizes the possible margin between points of  $k$  different classes. The margin is the perpendicular distance from the hyperplane to a class sample. Therefore, this hyperplane can be defined as a linear combination of the informative samples of the training data or support vectors that constitute the boundary of the class, as illustrated in Figure 3.9. Training data is mapped or classified by a nonlinear kernel function like a linear, polynomial, sigmoid or radial basis function [10].



**FIGURE 3.9: REPRESENTATION OF THE OPTIMAL HYPERPLANE AS CLASS BOUNDARY AND ITS RELATION WITH SUPPORT VECTORS.**

Originally, SVM is binary classifier that divides the data into two classes. However, it can be implemented for multiclass problems via: one-against-all (OAA) or one-against-one (OAO). OAA approach employs  $k$  binary classifiers to evaluate the training set for one class with respect to the other ones. This method is simple, relatively fast and accurate. On the other hand, OAO employs  $k(k-1)$  SMV to test for all possible pair of class combinations. At the end, the final output is the class that obtains the highest contributions of the overall tests. This procedure has the advantage of retrieving a probability for each class, yielding in better generalization and certainty [10],[23].

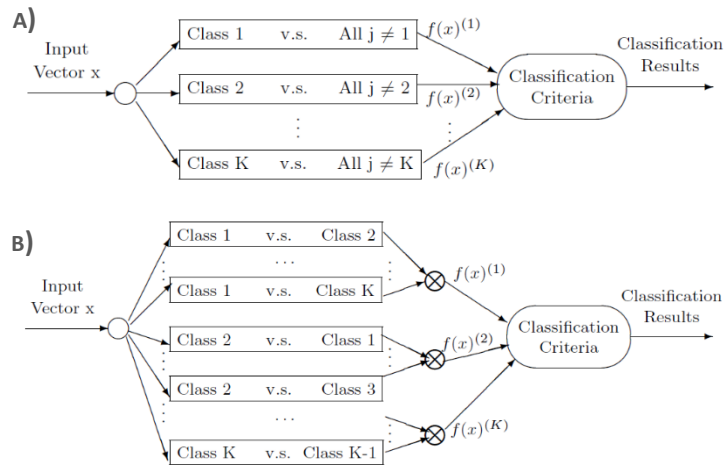


FIGURE 3.10: A) OAA AND B) OAO CONFIGURATIONS

Oskoei and Hu [10] assessed the application of SVM to classify upper limb motions using EMG signals. Four common kernels (radial-basis function, linear, polynomial and sigmoid) were evaluated against LDA, MLP with one hidden layer (MLP1) and MLP with two hidden layers (MLP2). Signals were collected from the forearm of eleven healthy individuals with four-channel EMG to classify six different limb motions. For the multifeature set composed of MAV, WL, SSC and ZC, results indicate a similar performance of all SVM-based classifiers with an average accuracy of all kernels of  $95.5 \pm 3.8\%$ , followed by LDA with  $94.5 \pm 4.9\%$ . Finally, MLP2 performed similarly to SMV and LDA with approximately 95% average accuracy, whereas MLP1's dropped around 91%.

### 3.4.6. HIDDEN MARKOV MODELS (HMM)

Hidden Markov Model (HMM) is a probability-based classifier. It is formed by a network of states related to each other by state transition probabilities. Each state is associated with a probability density function (modelled as a Gaussian mixture) that depicts the observed data probabilistic behavior (Observation symbol probability). HMM's outputs account for the probability of each state, being the highest one the final class or pattern. Usually, the initial-state occupancy probability and state transition matrix are predefined for each model. Therefore training goal is to model the Gaussian probability density function of each state via mean vector and covariance matrix calculation [14].

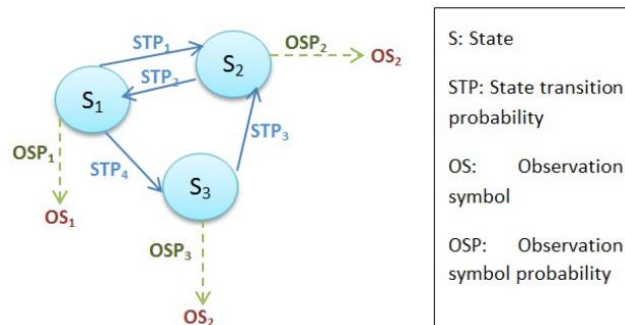


FIGURE 3.11: COMPONENTS OF HMM [2]

HMM benefits include the detection of time-varying signal characteristics, ability to model signal dynamics statically, high classification accuracy and low computational cost, suitable for real-time applications [2].

Chan and Englehart [39] applied HMM to classify six upper limb motions acquired by four EMG channels placed at the wrist, forearm and elbow. The proposed HMM was a state-driven method in which every motion pattern was associated to a state in the model. State transition probabilities were optimized empirically; assuming uniform distribution of the initial state and state-switching probabilities, and fixing a high probability to remain in a given state. The EMG signal classification of the fully connected HMM reached an average accuracy of 94.63%.

#### 3.4.7. K-NEAREST NEIGHBOR (KNN)

The basis of this approach is that each sample is classified based on the class of their  $k$  Nearest Neighbors (KNN). This technique is a twofold process that first identifies the nearest neighbors and then classifies the input by majority voting or by distance weighted voting [40]. The last method gives more relevance to the nearest neighbors by using the inverse or negative exponential of the Euclidean distance as weights. The main advantage of KNN is thus, its easy implementation.

However, KNN require training data during run time to compare the testing samples, and large training data sets may diminish real-time performance. Indeed it, is quite vulnerable to irrelevant or redundant features, but dimensionality reduction and neighbor weighting have efficiently solve this problem.

Purushothaman and Ray [41] proposed a prosthetic hand motion control using continuous EMG signals. Six hand and wrist movements were recorded with four EMG channels. Both TD and fourth order AR features were calculated and dimensionally reduced by PCA. The study compared pattern recognition performance of KNN and NN for both sets. The multilayer NN was built with five input and six output neurons and implemented for back-propagation training algorithm. Results revealed a higher average classification accuracy for TD features with KNN (84.3%) than with NN (80.8%). On the other hand, AR features were not quite accurately classified with neither KNN (59.1%) nor NN (63.5%).

#### 3.4.8. CLASSIFIER COMPARISON

Several studies have been conducted to compare the performance of different classifiers. For instance, Kaufmann *et al.* [42] compared conventional classifiers with Evolvable Hardware (EHW) to recognize eight to eleven hand movement patterns based on EMG signals from forearm muscles. The tested classifiers were: KNN, MLP, SVM, Decision Trees (DT, based on fuzzy logic) and EHW. The latest is a new adaptable classification method, based on a programmable logic array like-structure that is optimized with evolutionary or genetic algorithms. Results of the experiment in which five recording sets were used for training and one for testing, show that SVM is the most accurate classifier with an average misclassification error of 9%, closely followed by MLP and KNN with 10.44% and 10.45%, respectively.

Radman *et al.* [43] evaluated various feature combinations and classifiers for pattern recognition methods in dynamic myoelectric control systems. The classifiers employed

were: KNN, SVM, NN, Fuzzy Clustering (FC or FIS according to the nomenclature of this chapter), LDA and Mahalanobis Distance (MD), similar to the Euclidean distance but taking into account correlation in the data. Eight hand motions were recorded changing position, orientation and load to recreate more realistic conditions. Results of the classification error taking into account each feature set, show an increase in performance from FC, MD, KNN, NN, SVM and LDA, see Figure 3.12. Hence, due to its high accuracy and easy implementation, LDA was considered ideal for real-time applications.

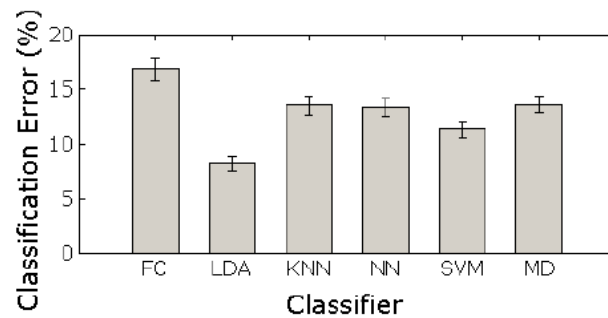


FIGURE 3.12: COMPARISON OF THE CLASSIFICATION ERROR AMONG THE TESTED CLASSIFIERS [43]

#### 3.4.9. COMBINATION OF CLASSIFIERS

The previous methods are the main classifier techniques employed in myoelectric pattern recognition. However, some researchers have studied the performance of combined classifiers in order to compensate the weaknesses of each method and refine their performance. The major approaches in classifier combination are according to [2], [44]:

1. Parallel architecture: individual classifiers are applied independently and their outputs are then combined by a combiner to retrieve the final class, based on major voting for instance. Fariman *et al.* [45] designed a hybrid Adaptive Resonance Theory (ART)-based neural network to classify upper limb myoelectric signals. ART is a supervised learning method of the NN. The proposed hybrid classifier (Best ART) employed four ART-based classification methods in parallel. The combiner algorithm is based on the classification accuracy of each independent classifier and the maximum rule to select the output with highest estimated confidence as final class. [46] Best-ART approach classification performance was tested against the four ART-based classifiers independently, KNN and LDA. Results show that Best-ART has the highest average classification accuracy (89.09%) and best computation time requiring less than half of LDA or KNN training and processing time.
2. Cascading architecture: individual classifiers are employed in a linear sequence to compensate the errors and refine the accuracy of the preceding classifier. Karlik *et al.* [47] applied this architecture to design a neuro-fuzzy classifier called fuzzy clustering network (FCNN). Firstly, this method employs fuzzy clustering to divide the data into six overlapping clusters representing six upper limb motions. This algorithm estimates the center of each class and assigns input

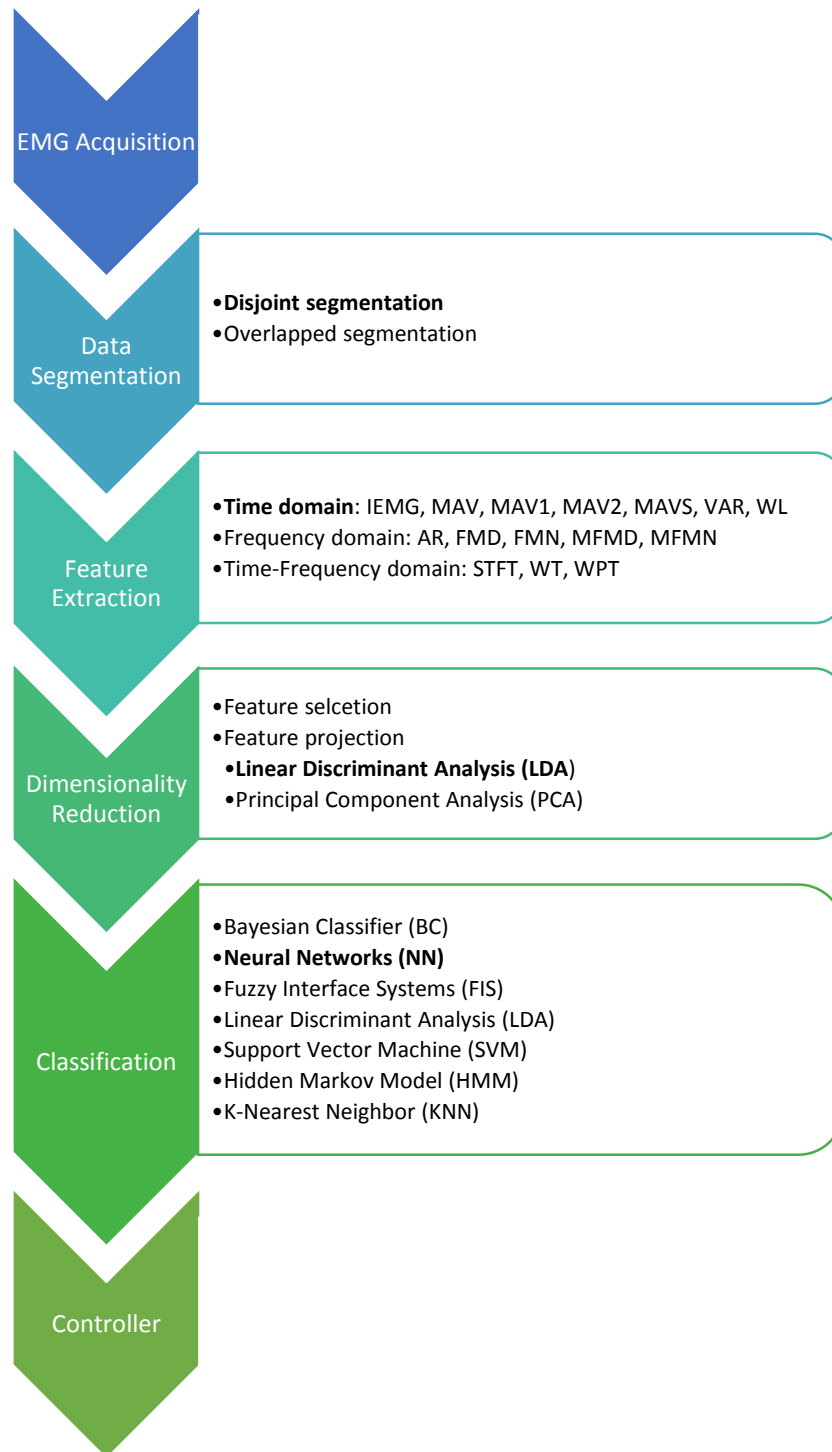


data a certain degree of membership. This clustered data of fuzzy cluster centers is used as input to the NN to retrieve a final classification value. The study compared classification performance of FCNN, MLP and Conic Section Functional Neural Network (CSFNN, variation of MLP that not only takes into account open boundaries but also class centers) with four AR parameters and signal power as features. Results show a maximum classification accuracy of 88% for CSFNN, 97% for MLP and 98% for FCNN. In addition, it was proved that FCNN required less training time than the other classifiers to obtain reliable results.

3. Stacked architecture: Several classifiers are combined so that their outputs estimate the possible clusters and then, they are used as training set of the stacked classifier. Final classification takes into account a combination of the outputs of stacked classifier and individual ones. Although this architecture offers high efficiency and flexibility in class separation, it is seldom used for EMG signal classification. An example of this architecture is presented in Xiong and Cherkassky [36] who developed a combined SVM with LDA (SVM/LDA). This approach can be considered a generalization of SVM that takes into account both local and global properties (characteristic of LDA). The SVM/LDA classification performance was evaluated against SVM and LDA alone, with a synthetically generated twonorm data set (20 dimension for 2 classes with normal distribution and unit covariance matrix) [48]. 90% of this data set was used for training, while the remaining 10% was employed in testing. Results demonstrate that SVM/LDA has the highest performance with an average classification accuracy of 98.3%, compared to SVM (95.8%) and LDA (97.4%).

To conclude, Figure 3.13 summarizes the content of this chapter, showing the different steps involved in pattern recognition of a myoelectric control system, and the associated techniques for each step. The present thesis has been developed taking into account these theoretical bases to classify user's motions in order to actuate a robotic hand. The employed methods are in bold in Figure 3.13.





**FIGURE 3.13: STEPS OF A MYOELECTRIC CONTROL SYSTEM FOR PATTERN RECOGNITION, WITH THE EMPLOYED TECHNIQUES IN BOLD**

# CHAPTER 4

## 4. EMG ACQUISITION SYSTEM

The goal of the present acquisition system is to use the minimum possible number of components to collect accurately the EMG signal. Therefore, the system is integrated into MATLAB/ Simulink, to benefit from high performance and easy implementation of digital signal processing. This chapter describes and explains the advantages of the materials used to acquire the EMG signals, focusing on the electronic circuit, microcontroller and acquisition software.

### 4.1. EMG ELECTRONIC CIRCUIT

The employed EMG acquisition circuit has been developed by Ángel García Martín-Engeños [50] during his bachelor thesis in this Department of Signal Engineering and Automation. The circuit has been implemented according to De Lucas' specifications [5], minimizing electromagnetic noise by maximizing the signal-to-noise ratio (SNR) and common-mode rejection ratio (CMRR) [51].

The main advantages of this circuit are its effective design and integration in the MATLAB/ Simulink environment. This four-channel assembly allows accurate EMG signal acquisition using the minimum number of components and thus, reduces the cost. Its compatibility with MATLAB/ Simulink offer real-time processing of the obtained signals, including digital filtering. All this features imply a significant improvement from commercial acquisition systems that have a trade-off between accuracy and cost. Therefore, the utilization of this circuit is consistent with the motivation of the present study.

The circuit can be divided in three stages represented in Figure 4.1.: fully-differential measurement, common mode active feedback and signal digitalization.

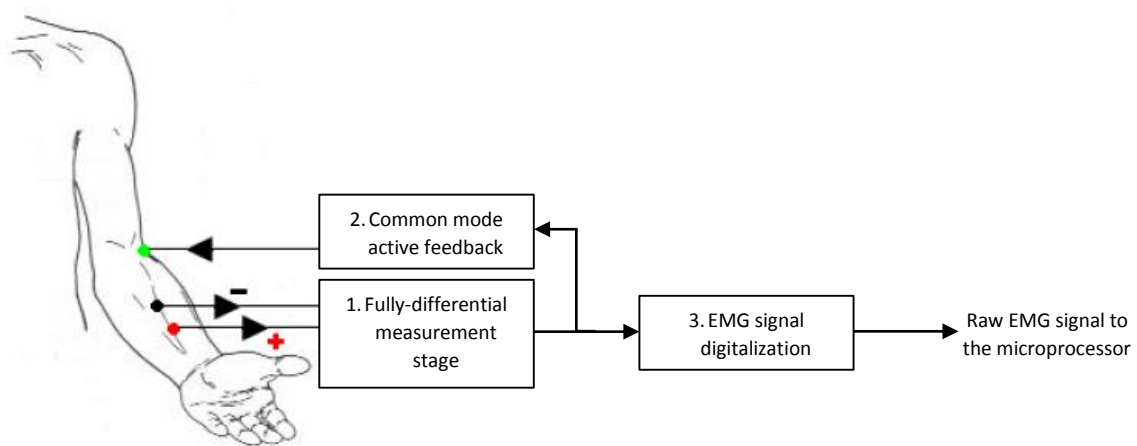
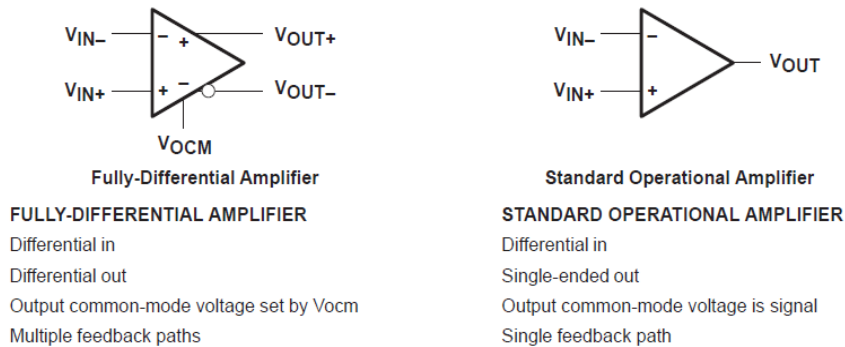


FIGURE 4.1: MAIN PARTS OF THE EMG ACQUISITION CIRCUIT [MODIFIED FROM 50]

#### 4.1.1. FULLY-DIFFERENTIAL MEASUREMENT

This is the first step in EMG signal acquisition, where the signal is measured and amplified. Fully-differential amplifiers work effectively with low-voltage systems like this one, increasing the immunity to external noise, the dynamic range of the output voltage and reducing the even-order harmonics (a measure of signal distortion) [52]. Its structure is quite similar to the standard amplifier, but instead of a single-ended output, fully-differential amplifiers have a differential output, as shows Figure 4.1. This configuration provides an independent control of the common mode signal that will be used in the next step to cancel noise.



**FIGURE 4.2: FULLY-DIFFERENTIAL AMPLIFIER VS STANDARD OPERATIONAL AMPLIFIER [52]**

In addition, a radiofrequency passive low pass filter has been included in this stage with a cut-off frequency of 160 kHz, to improve circuit's noise tolerance. Circuit's performance is stabilized with another filter that adds a pole at approximately 1600 Hz. It is essential that this process is applied to both differential outputs so as to preserve the symmetry of the signal.

The amplification of the input signal is controlled with a multi-turn potentiometer to optimize the gain specifically for each user. Finally, a common mode signal collector is placed in parallel to the differential output.

#### 4.1.2. COMMON MODE ACTIVE FEEDBACK

Common mode signals are noise and artifacts that affect both inputs. Variations of electron input impedances, cables, input electronic components or external powerlines are some sources of these alterations. The idea of this stage is to collect the common mode signal from the user, and return its inverse back to the subject. This noise cancellation method is called Driven Right Leg (DRL) and has been widely used in ECG recordings. An extra reference electrode is placed along the DRL circuit, to provide a low impedance path between the subject and the reference of the amplifier. Therefore, DRL reduce several orders of magnitude electrode's impedance, so that only a small non-hazarding current flows through the subject while eliminating noise artifacts [53]. In this case, the reference electrode is placed in the elbow instead of the right leg, being an electrically neutral area near the measured muscles. This technique cancels 50/ 60 Hz powerline noise without applying a notch filter, and thus, avoids distortion of the EMG bandwidth, which ranges from 20-500 Hz.

#### 4.1.3. SIGNAL DIGITALIZATION

The measured signals must be sent to the microcontroller for the final processing stage. Hence an ADC is required to convert the analog signals into digital ones that can be handled by the microcontroller and computer.

The employed ADC has high delta-sigma resolution with differential input and low noise signal. This model uses high number of bits to map the dynamic range of the analog signal, and thus is able to accurately digitalize small analog signals, like the measured EMG. Delta-sigma ADC architecture also reduces the size and complexity of the circuit, as additional filtering and amplification steps can be substituted by a first order anti-aliasing (low pass) filter.

Sampling frequency is adjusted with an external oscillator to fulfill Nyquist theorem, which states that sampling frequency must be at least two-times original signal's upper frequency bound to be properly recovered after digitalization. In this case, sampling frequency must be at least 1 kHz, but is set to 1190 Hz due to commercially available components.

The circuit layout has been implemented in a PCB using Through-Hole Technology. Electrodes are linked to Jack 3.5 connectors by shielded wires to increase noise resistance. The PCB has five female Jack 3.5 plugs for: the four input signals (one from each channel) and the elbow reference electrode. On the other hand, a ten pin male connector is used for power supply and output-microcontroller communication. Figure 4.3 depicts the employed PCB.

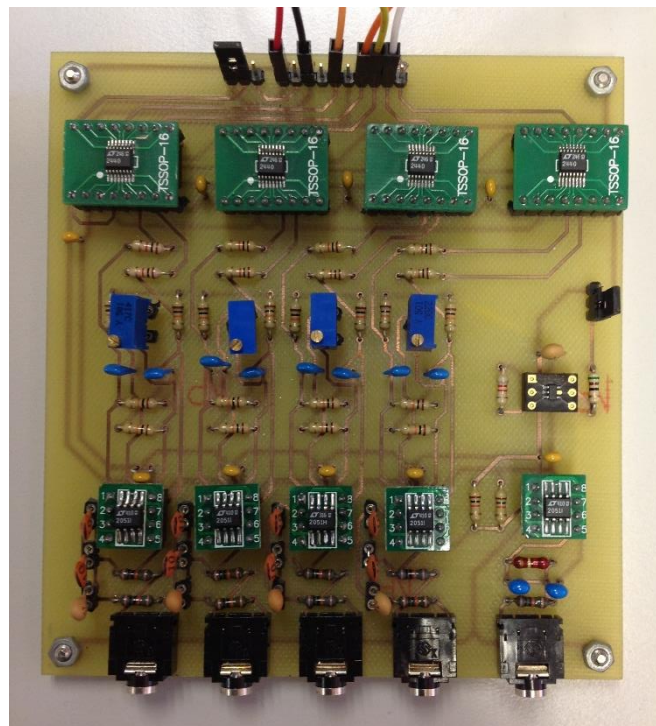


FIGURE 4.3: PCB OF THE EMG ACQUISITION CIRCUIT

## 4.2. STM32F4 MICROCONTROLLER

The microcontroller board is the intermediary between the acquisition circuit and the MATLAB/ Simulink environment, where the acquired signals are finally processed. Therefore, STM32F4-Discovery has been chosen due to its high compatibility with MATLAB/ Simulink and its low cost.

STM32F4 Discovery is built around STM32F407VGT6 microcontroller with 1 MB of Flash memory and 192 KB of RAM. This microcontroller is able to work at 168 MHz/ 210 DIMPS (Dhrystone Million Instructions per Second, measure of computer performance) with floating numbers; providing an edge in control algorithm execution, code efficiency and support for meta-language tools, among other features. Furthermore, STM32F407VGT6 is designed for high performance and ultra-fast data transfers, significantly relevant for real-time control applications.

The embedded microcontroller is connected to several peripherals (Figure 4.4) that include:

- ST-LINK/V2 programming and debugging tool
- 2 Pushbuttons (user and reset)
- LED (LD3 – LD6)
- CS43L22 audio DAC
- USB
- LIS302DL or LIS3DSH accelerometer
- MP45DT02 microphone
- Connectors (80 pins)

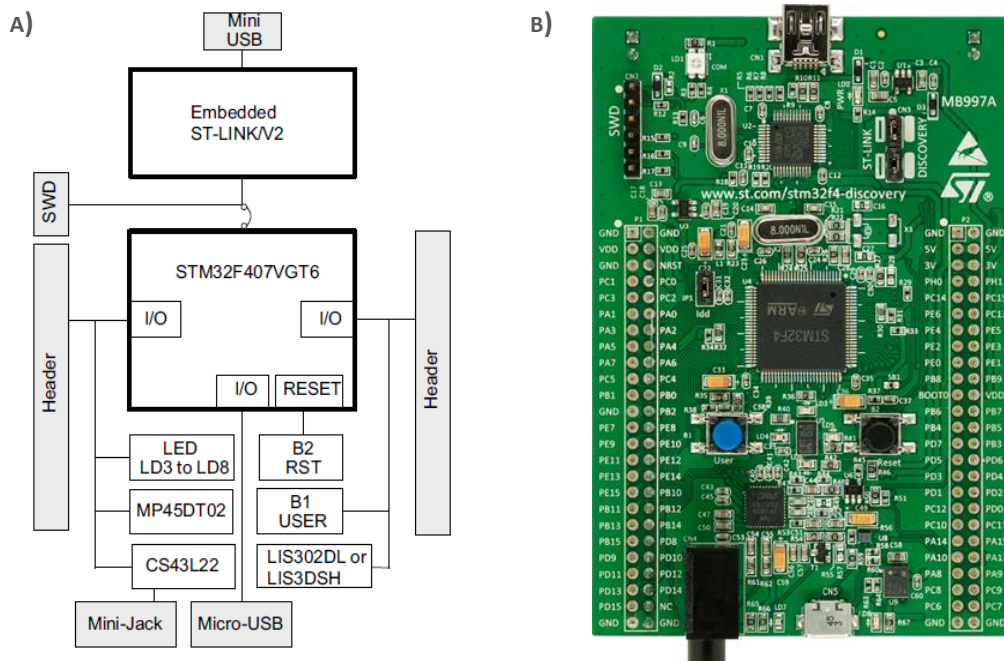


FIGURE 4.4: A) HARDWARE BLOCK DIAGRAM AND B) STM32F4 BOARD [54]

The board can be powered by the host PC through the USB cable or by an external 5V source. In addition, STM32F4 can be used as a power source delivering 3V and 5V.

More information can be found in the user manual of the manufacturer [54] (including datasheet and schematics).

### 4.3. ACQUISITION SOFTWARE

Along this chapter, great importance has been given to digital signal processing so as to develop a robust, flexible and fast acquisition system. In this step, the EMG signals are filtered to eliminate undesired frequencies below 20 Hz and above 500 Hz. To achieve this, the acquisition system employs a MATLAB/ Simulink Advanced Rapid Control Prototyping (ARCP) system implemented by the PhD. Antonio Flores Caballero [55], member of the Department of Systems Engineering and Automation.

#### 4.3.1. SIMULINK RCP METHODOLOGY

As stated in [55], RCP comprises all software and hardware techniques that shorten control systems development and practical application. To achieve this, a high level of abstraction is required to handle programming complexity; yielding in an automatic code generation, compilation and program loading in the hardware. Visual or graphical programming languages like Simulink's, provide this abstraction level as well as a Model Based Design that allows both simulations and RCP systems' programming, among other applications. Therefore, multidisciplinary users without a deep background in programming, can benefit from this intuitive technology to avoid low-level configuration and work directly with real hardware of control.

Usually, RCP systems have been utilized for development stages and laboratory experimentation; however, the present ARCP, also covers the final implementation and validation stages, providing more accurate results. ARCP technology focuses on real-time applications and digital signal processing, using just one microcontroller for the different stages mentioned before.

Consequently, this combination of Simulink and ARCP, constitute a powerful environment to process signals immediately after acquisition for analysis, control or simulation purposes. In addition, MATLAB and Simulink integration, enable the exportation of Simulink results for further processing with MATLAB algorithms [56].

#### 4.3.2. SIMULINK ACQUISITION MODELS

According to García Martín-Engeños [50], two block-models have been developed with Simulink and ARCP software. The first one, called Target, is implemented in STM32F4 Discovery board to acquire and send the data to the second model. The latest is the so called Host model, which is executed in the PC, and is in charge of communication with the workspace where EMG signals are processed. These models have been developed using Simulink own libraries as well as "Waijung blockset" implemented by Aimagin [57] and "UC3M ADDONS STM32F4", by Antonio Flores Caballero.

##### 4.3.2.1. TARGET MODEL

This model is implemented in STM32F4 Discovery board and is built with five major blocks: *Target Setup*, *SPI Master Setup*, *ADCx*, *Data Composer* and *USB VCP Send STM32F4*.

#### 4.3.2.1.1. TARGET SETUP

*Target Setup* configures the hardware, in this case STM32F4 Discovery. Several options can be modified like the selected compiler that translates Simulink’s block models into board code, or the sampling frequency at which the board works. This last parameter should be always equal or greater than the sampling frequency of the USB and SPI Simulink blocks, compiler model and microcontroller model.

#### 4.3.2.1.2. SPI MASTER SETUP

*SPI Master Setup* configures the parameters of the Serial Peripheral Interface (SPI) bus that transmits the digitalized signals from the ADCs of the acquisition circuit.

Some parameters depend on the slave device (the acquisition circuit), like data format, baud rate or clock polarity and phase; whereas others are specific for the application of the model. The employed configuration enables receiving data from the slave (Half-Duplex\_Rx direction) and reading the four channels simultaneously (Custom slave select mode). Custom parameter requires an extra Digital Output block to implement the desired selection control, which is integrated in the *ADCx* block in this model.

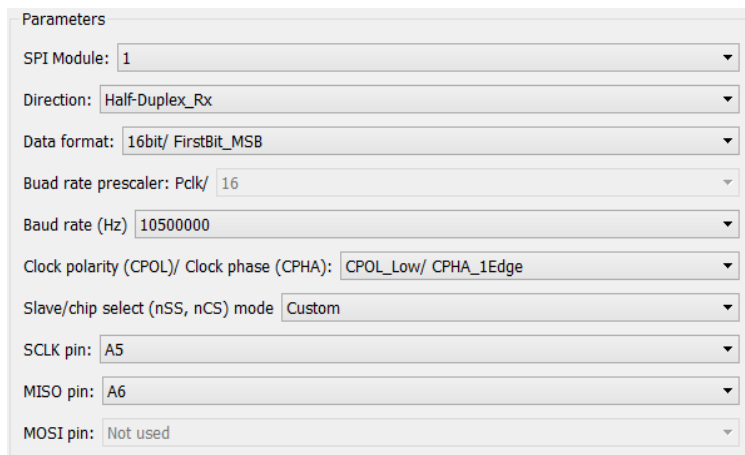


FIGURE 4.5: SPI MASTER SETUP AND SELECTED CONFIGURATION

#### 4.3.2.1.3. ADC<sub>x</sub>

*ADC<sub>x</sub>* controls the reception of signals from the circuit. It is formed by three sub blocks as illustrated in Figure 4.5, which perform specific functions. The model has four *ADC<sub>x</sub>* blocks, one for each channel.

The first sub block called *CS<sub>x</sub>* (blue in Figure 4.6) associates the physical channel *x* to the reading port of STM32F4 Discovery. Its internal Digital Output block enable the reading of the selected channel *x* by assigning a 0 to its pin during the reading time, whereas the non-selected channels have a fixed 1 in their corresponding pins.

Secondly, *SPI Device x* (green in Figure 4.6) actually reads the data of the enabled channel *x*. Finally the third sub block *CS-off* (gray in Figure 4.6), disconnects all channels digitally, assigning a 1 to their pins, and thus stopping the reading process.



Therefore, all ADCx must work in synchrony to read the four channels simultaneously.

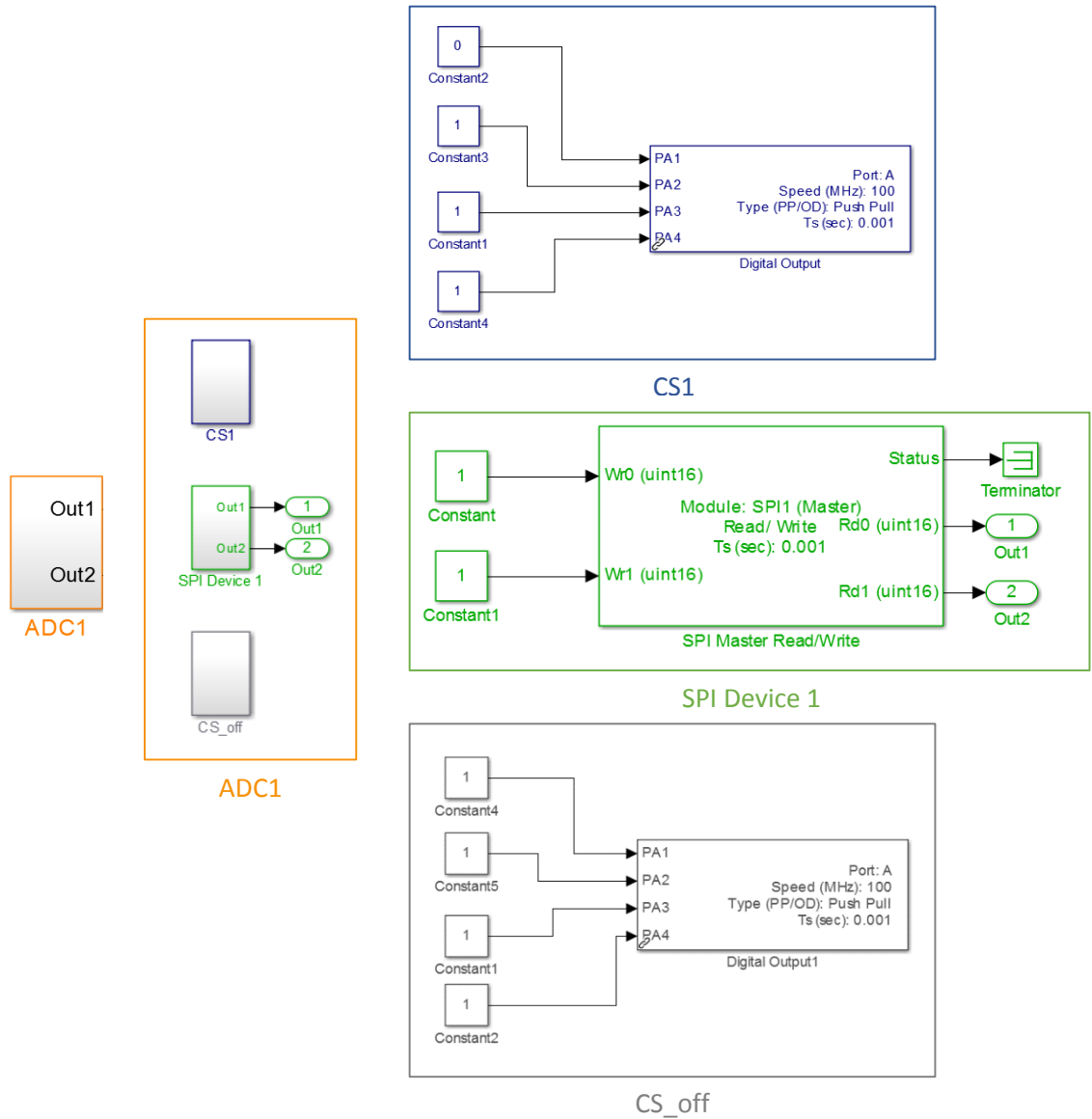


FIGURE 4.6: ADCx BLOCK AND SUB BLOCKS

#### 4.3.2.1.4. DATA COMPOSER

Data composer converts the train of digital pulses retrieved from the ADC, into int32 format with the algorithm of Figure 4.7.

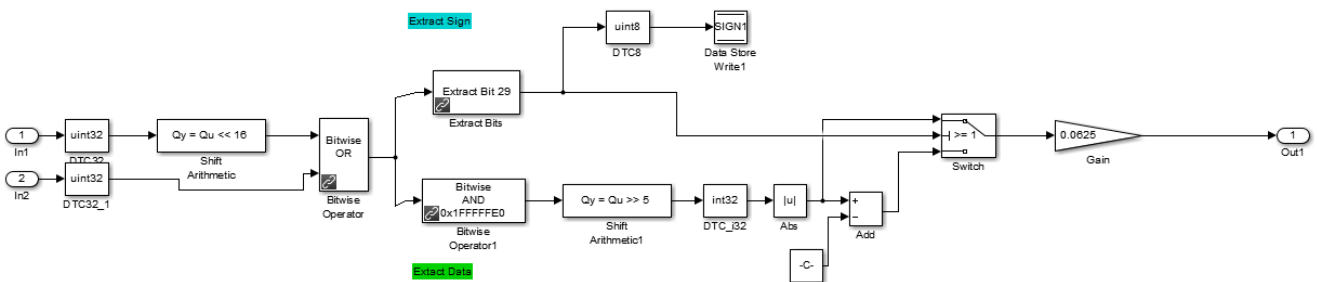


FIGURE 4.7: DATA COMPOSER INTERNAL BLOCK DIAGRAM



4.3.2.1.5. EXECUTION TIME PROFILER

This block measures the execution time of the model in microseconds for the STM32F4xx microcontroller family.

4.3.2.1.6. USB VCP SEND STM32F4

USB VCP Send STM32F4 block is responsible for the USB communication between STM32F4 and PC. It collects the pre-processed EMG data and sends them to the Host model for the final processing step. The configuration of this block sets the amount of data transferred per packet, format, heading, terminator and transfer speed. The sampling frequency that determines this last parameter, must be always equal or smaller than Target Setup's.

Figure 4.8 depicts the complete Target model with the blocks explained before and their connections.

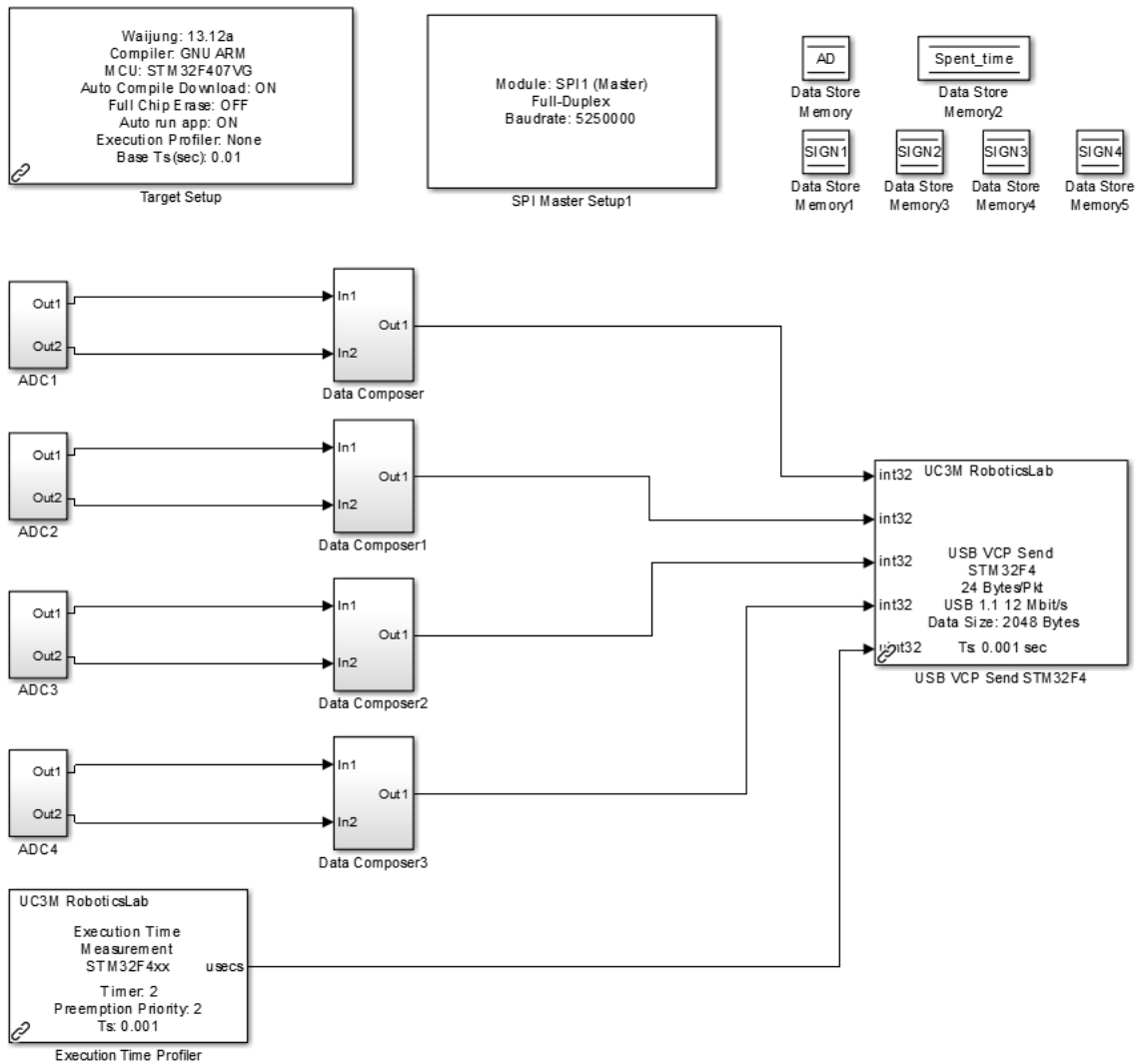


FIGURE 4.8: SIMULINK TARGET MODEL

#### 4.3.2.2. HOST MODEL

The Host model is implemented in the PC to filter, store and visualize the pre-processed EMG signals from the Target model. Host model comprises: *Host Serial Setup*, *Host Serial Rx*, *Data Type Conversion*, *Digital filter design*, *CHx* and *To Workspace* blocks.

##### 4.3.2.2.1. HOST SERIAL SETUP

This block configures the communication between STM32F4 Discovery and the PC through the COM port. Host serial setup determines the USB port for data transmission, reading speed (baud rate) and communication synchrony.

##### 4.3.2.2.2. HOST SERIAL RX

Host Serial Rx receives the data from Target’s USB VCP Send STM32F4 block, and therefore, their configuration must be identical in terms of amount of transferred data, format, header and termination. In addition the same COM port as in Host Serial Setup must be selected.

##### 4.3.2.2.3. DATA TYPE CONVERSION

This block converts the input signal data into the chosen format, applying the selected rounding method. In this case, Data Type Conversion transform the int32 input into single format using floor rounding.

##### 4.3.2.2.4. DIGITAL FILTER DESIGN

Regarding digital signal processing, the filtering block is the most important part of the model. Digital filters are more precise, smaller and cheaper to implement than analog filters, with the only drawback of increasing slightly the computing time.

The aim of this filter is to eliminate the frequency bands that do not provide myoelectric information. Therefore, a bandpass filter is required to eliminate the frequencies below 20 Hz and the ones above 500 Hz. A high order (8<sup>th</sup>) filter is implemented due to its resemblance to the ideal filter.

Furthermore, Butterworth filter architecture has been chosen due to its high performance compared to other filter types, as depicted in Figure 4.9

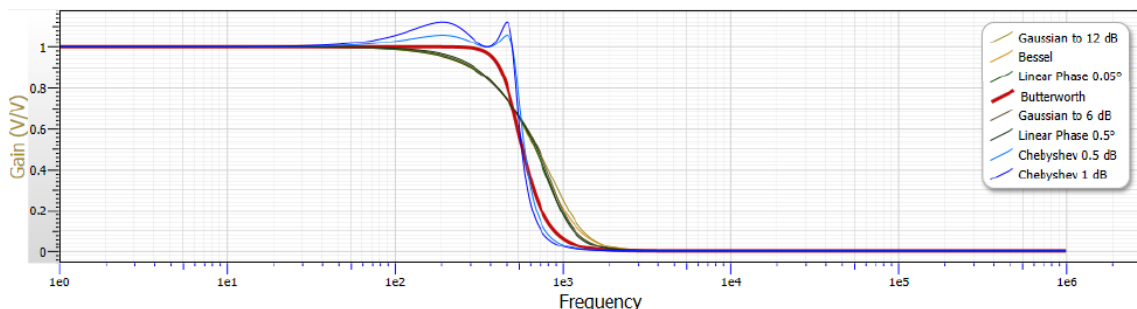


FIGURE 4.9: COMPARISON OF THE STABILITY PERFORMANCE OF DIFFERENT FILTER TYPES [50]

The final filter configuration is indicated in Figure 4.10

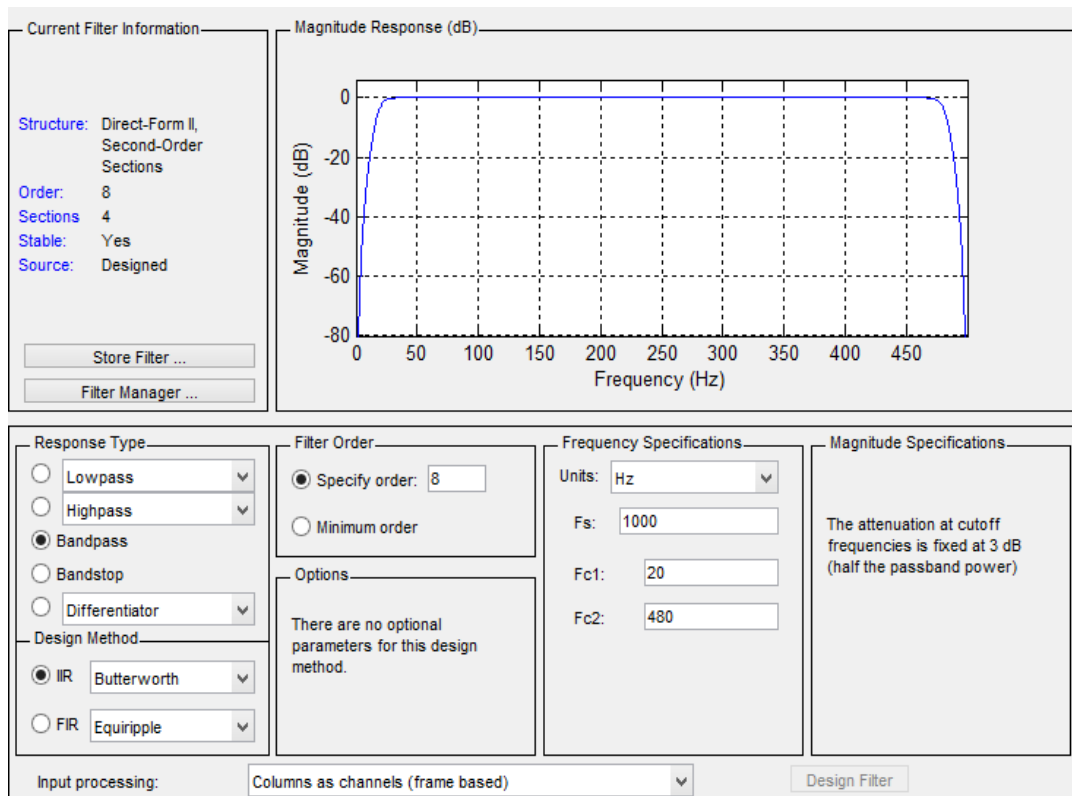


FIGURE 4.10: HIGH ORDER BUTTERWORTH BANDPASS FILTER CONFIGURATION

#### 4.3.2.2.5. TO WORKSPACE

To Workspace block saves the processed EMG signals into a MATLAB timeseries, array or structure which is sent to MATLAB workspace. In this model, the signals are stored in a struct with a time array.

#### 4.3.2.2.6. CH<sub>x</sub>

This block is a time scope that displays the processed signal that is saved in the workspace.

Figure 4.11 represent the complete Host model that is implemented in the PC, with the blocks explained previously.

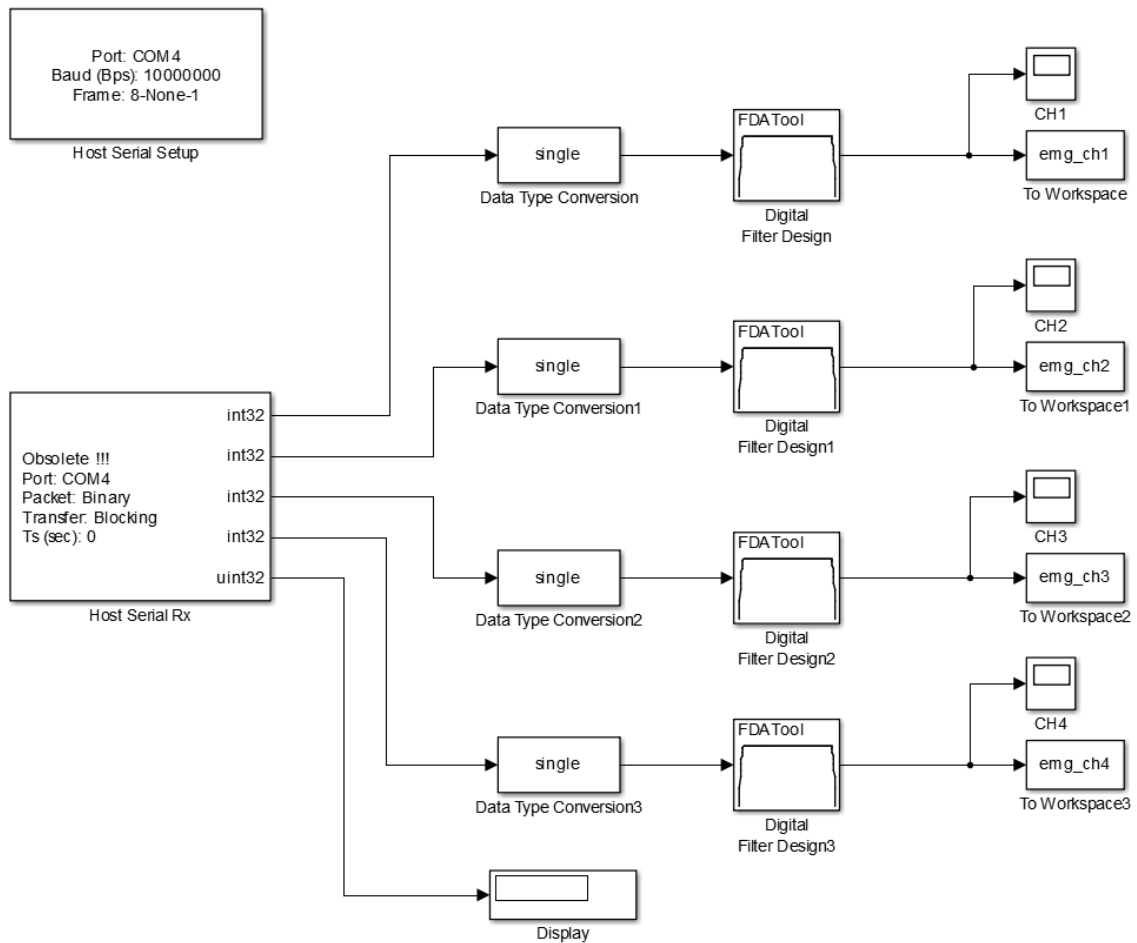
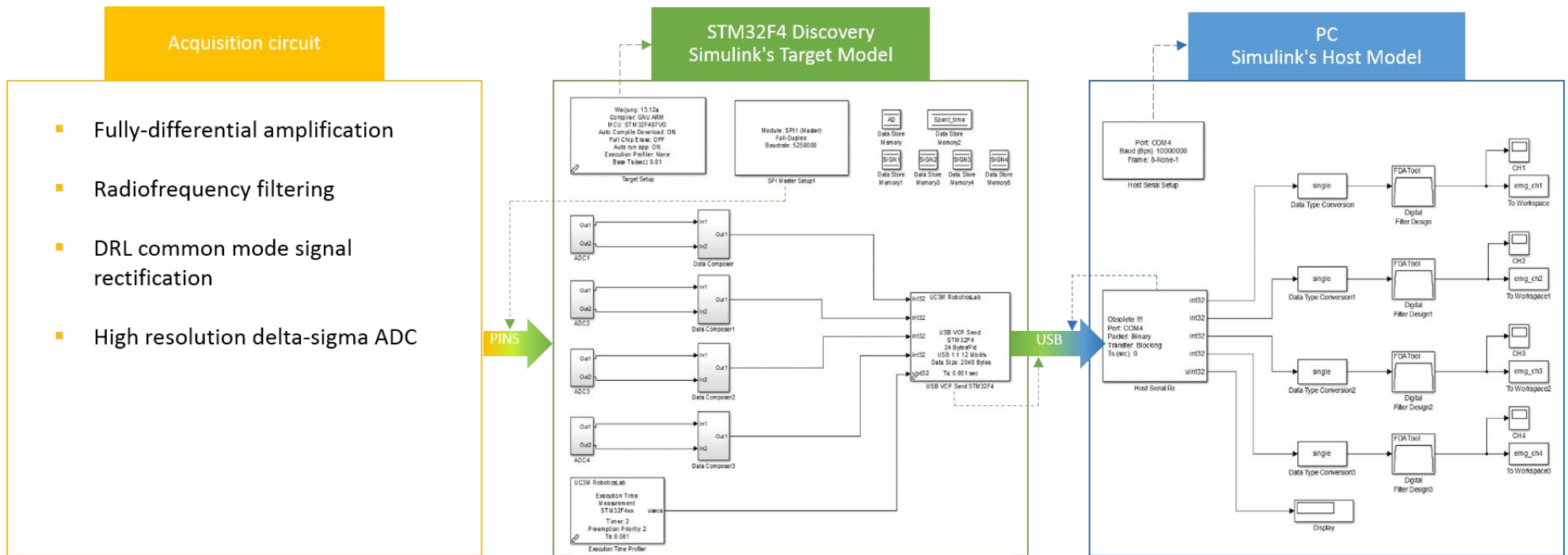


FIGURE 4.11: SIMULINK'S HOST MODEL

The following scheme, Figure 4.12, is a review of the content of this chapter, in which each device is coupled with its correspondent processing step. The diagram covers the acquisition process from the analog circuit up to the final processing step and storage in the computer. Dashed lines in Simulink's models represent the device or process controlled by that block.

Finally, the obtained signals have been used in this study to develop a training set for the employed NN classifier.



# CHAPTER 5

## 5. NEURAL NETWORK-BASED MYOELECTRIC CONTROL SYSTEM

This chapter describes the implementation process of the neural network-based myoelectric control system (NNMCS). The NNMCS works with a predefined set of gestures recorded with the acquisition circuit previously explained. Several features are extracted and dimensionality reduced to train the NN. Once the configuration of the NN has finished, it is implemented in a Simulink block in order to build the real-time control model.

### 5.1. GESTURES

The 3D printed robotic hand that has been used as a support for the present control system, has 7 pre-programmed movements at its low control level. More information is provided in the Bachelor Thesis of Esperanza Marín Conde: “Construction of a Robotic Hand for Myoelectric Control System Research Applied to Low-Cost Prostheses” [58], which corresponds to the first part of this project.

These gestures comprise: open/ close hand, extended index, pincer movement, pronation/ supination and hand’s neutral position, in which all fingers are slightly bent, called rest. The user must perform these motions to actuate the printed robotic hand. Therefore, the first step in NNMCS development is to record these gestures to train the NN classifier, so that it can interpret user’s movement intentions.

### 5.2.ACQUISITION PROCEDURE

To record these gestures, 4 EMG channels have been used (each of them with two Ag-AgCl surface electrodes) and a reference electrode in the elbow. As the programmed gestures involve mainly the movement of the fingers, two pairs of electrodes have been placed on top of Flexor digitorum superficialis and Extensor digitorum muscles. In addition, two more pairs of electrodes have been used to record the activity of Flexor carpi ulnaris and Brachioradialis responsible for hand flexion and pronation/supination, respectively. Figure 5.1 shows the electrode pairs location in the forearm.

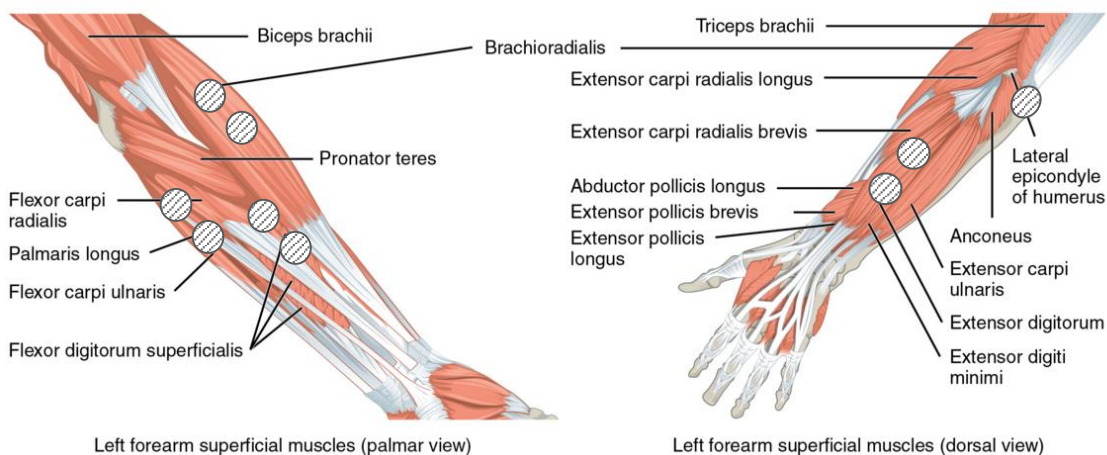


FIGURE 5.1: TARGET MUSCLES AND ELECTRODE LOCATION

Once the electrodes have been placed properly, they are connected to the acquisition system explained in Chapter 4, including the EMG acquisition circuit and STM32F4 Discovery microcontroller and PC Simulink models. In order to automatize the motion recording process, a MATLAB script has been implemented. This program requests the user to perform the desired motions in a certain order:

1. Motion 1: Close hand
2. Motion 2: Open hand
3. Motion 3: Extended index
4. Motion 4: Pincer movement
5. Motion 5: Pronation
6. Motion 6: Supination
7. Motion 7: Rest

Each motion is repeated five times, alternating with resting periods. The data acquisition program calls Simulink's Host model, which is connected to the Target model in STM32F4, to record signals of 5 s at 1 kHz sampling frequency. The idea of this methodology is to record only the static signals, therefore, the first 100 samples are removed to correct small transition between rest and the performed motion, as well as filter stabilization.

To save the obtained signals, four matrices (one for each channel) are allocated to each motion. Every repetition is stored in a column of its corresponding motion-channel matrix. The final outcome of the program are 28 motion-channel matrices of 4900x5. As this data set is quite large, the acquired signals are furtherly processed to extract their features and reduce their dimensions.

### 5.3. IMPLEMENTATION OF THE MYOELECTRIC CONTROL SYSTEM

The main concern of the present work is to provide a real-time control of the prosthetic arm; therefore, the most important issue regarding to signal processing is to reduce the computational load the maximum possible.

The first measure to achieve this and fulfill real-time constrains, has been the recording of the stationary part of the EMG signals. This approach enables precise transitions between the different gestures, dividing the data into shorter segment lengths without critically compromising classification accuracy, and reduce computational load, as no time-frequency domain features are required to describe transitory components.

These advantages have been taken into account to process the acquired data and develop the NN classifier. In this section, the generation and training of the NN classifier are explained along with the required data segmentation, feature extraction and dimensionality reduction.

### 5.3.1. DATA SEGMENTATION AND FEATURE EXTRACTION

After the acquisition process is completed, each motion-channel matrix is processed independently. Firstly, the analyzed matrix is split into disjoint segments of 150 samples, which correspond to 150 ms. As previously mentioned, steady-state signals are less sensible to misclassification due to short segment length; therefore in the present work, this length has been slightly reduced from 200 ms to 150 ms to leave more time for the pattern recognition process.

Once the signal has been segmented, nine time-domain features are computed. This feature domain choice is based again on the perspective of reducing the computational load. Time-domain features have proven a relatively high classification accuracy and computation efficiency when compared to frequency-domain features [21]. Hence, the chosen features are: IEMG, MAV, RMS, VAR, WL, ZC, SSC, WAMP and SSI, allocated to feature 1-9 respectively. The program in charge of feature extraction, calculates these parameters according to the equations of Table 3.1, for each segment of each repetition of the corresponding channel of the analyzed motion. A flowchart of the program is represented in Figure 5.2.

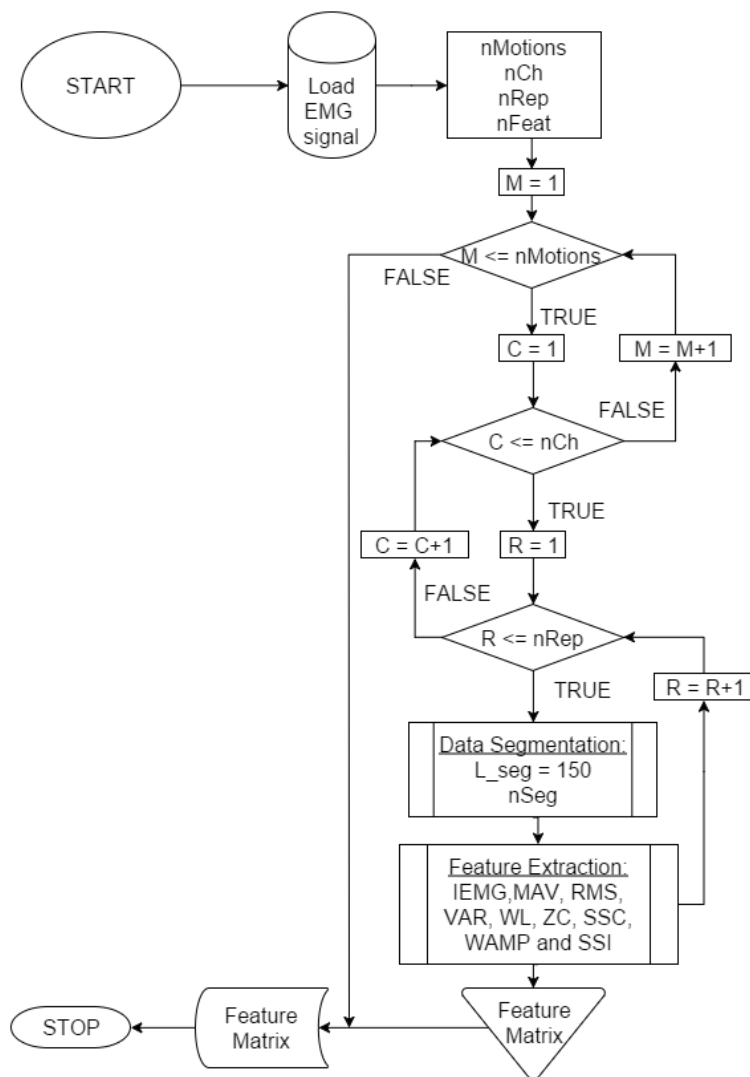


FIGURE 5.2: FLOWCHART OF THE FEATURE SEGMENTATION AND FEATURE EXTRACTION MATLAB SCRIPT



Finally, an m-by-n feature matrix is generated, arranged according the distribution scheme depicted in Figure 5.3.

$$features_{m \times n}$$

$$m = \text{number of motions} \cdot \text{number of repetitions} \cdot \text{number of segments}$$

$$n = \text{number of channels} \cdot \text{number of features}$$

		Feature 1			...			Feature 9		
		Ch1	...	Ch4				Ch1	...	Ch4
Motion 1	Rep. 1									
	...									
	Rep. 5									
...	...									
Motion 7	Rep. 1									
	...									
	Rep. 5									

FIGURE 5.3: DISTRIBUTION OF THE FEATURE MATRIX

### 5.3.2. DIMENSIONALITY REDUCTION

The obtained feature matrix needs to be reduced to accelerate motion classification. The selected technique has been feature projection using LDA, due to its robustness and higher classification accuracy of its reduced feature set compared to PCA's [26].

The integration of this technique into MATLAB environment, and thus in the present study, has been possible with *drtoolbox* (Dimensionality Reduction MATLAB toolbox) developed by Laurens van der Maaten of Delft University of Technology [59]. The LDA function provided by *drtoolbox* is based on Fisher's Linear Discriminant criterion function, which maximizes the ratio of between-class variance and within-class variance.

The function receives as inputs: the data to be reduced, a target matrix and the number of dimensions of the embedded feature space. The input data matrix must have variables (features) in columns and observations (movement repetitions) in rows, which is already fixed for the previously constructed feature matrix. The target matrix labels each instance of each motion with a specific value. The last input determines the dimensions of the embedded space in which the feature matrix is mapped, to create a smaller matrix with the optimal combination of original features. In practice, this parameter controls the size of the reduced feature matrix; and hence, the degree of dimensionality reduction.

Firstly, the script computes the mean ( $\mu$ ) of the original set and the total covariance ( $S_T$ ). With these values, between-class ( $S_B$ ) and within-class ( $S_W$ ) scatter are calculated to set out the generalized eigenvalues and eigenvectors problem:

$$A \cdot v = \lambda \cdot v$$

where  $\mathbf{v} = S_w^{-1} \cdot S_B$ , corresponding to Fisher's linear discriminants criterion function that needs to be maximized;  $\mathbf{v}$  are the generalized eigenvectors and  $\lambda$ , the eigenvalues. The importance of these parameters is that they determine the new LDA subspace. Basically, the eigenvectors represent the axis of the new subspace, and the associated eigenvalues dictate how informative the new axis are. The higher the eigenvalues, the more informative the eigenvectors are, so the ratio of between-class and within-class variance increases, yielding in better class separability. Therefore, the eigenvalues and eigenvectors are sorted in descending order to construct the linear mapping matrix denoted by  $M$ . Its column size is limited by the inputted number of dimensions.

Finally, the function outputs the mapped set (reduced features), as well as the mapping information including the mean, mapping matrix  $M$  formed by the eigenvectors, and the sorted  $\lambda$  eigenvalues.

The reduced feature set is used to train the NN; whereas the mapping information is fed into *out\_of\_sample* function of *drtoolbox*. This function reduces the dimensions of new data applying the transformation provided by mapping information. A trained LDA dimensionality reduction block is implemented in Simulink's control model, employing both mapping information and *out\_of\_sample* function, to reduce feature dimensions of the EMG signals in real-time.

### 5.3.3. NEURAL NETWORK GENERATION AND TRAINING

The generation of the NN classifier is the crucial step in the implementation of the myoelectric control system. MATLAB provides a Neural Network Toolbox with several functionalities: Function Approximation and Nonlinear Regression, Pattern Recognition Classification, Clustering, Time Series and Dynamic Systems, Neural Network Control Systems and Define Neural Network Architectures [60]. For the present work, the utility that has been used is Pattern Recognition Classification.

The workflow of the NN generation comprises:

1. Data collection
2. Network creation
3. Network configuration
4. Weights and biases initialization
5. Network training
6. Network validation (post-training analysis)
7. Network utilization

There are two approaches to generate the NN: Neural Pattern Recognition graphical user interface (GUI) and command-line functions. *nprtool* opens the Neural Pattern Recognition App, which guides the user through the NN generation steps. Once the NN has been created, trained and validated; the GUI can generate a MATLAB script (with command-line functions) to reproduce the NN generation steps and configuration employed by GUI. This script not only enables a faster NN creation and training with new data sets, but also provides advanced configuration parameters, which are not available in the GUI. However, it is advisable to use the GUI to firstly implement this script, as it is more intuitive than directly programming with NN Toolbox functions.

Firstly, the GUI requests the definition of the problem, where the user selects the input data that is presented to the NN and the target matrix that indicates the desired NN output. These variables must be in the workspace to be loaded. Depending on the element arrangement of the input data, it can be classified into matrix columns or row columns. Reduced features matrix follows the matrix row configuration, where the number of columns determines the number of elements, and the number of rows, the samples of the data set. This parameter is important to construct the target matrix, as it must have the same structure type as the input data. This matrix works as a switch in which 1 means belonging to that class (motion in this case) and 0, no fit in. The target matrix arrangement depicted in Figure 5.3, creates a class for each motion, with the same sample size as the reduced features matrix.

$$target\_matrix_{m \times n}$$

$$m = \text{number of motions} \cdot \text{number of repetitions} \cdot \text{number of segments}$$

$$n = \text{number of motions}$$

	M1	M2	M3	M4	M5	M6	M7
M1: (nRep·nSeg)x1	1	0	0	0	0	0	0
M2: (nRep·nSeg)x1	0	1	0	0	0	0	0
M3: (nRep·nSeg)x1	0	0	1	0	0	0	0
M4: (nRep·nSeg)x1	0	0	0	1	0	0	0
M5: (nRep·nSeg)x1	0	0	0	0	1	0	0
M6: (nRep·nSeg)x1	0	0	0	0	0	1	0
M7: (nRep·nSeg)x1	0	0	0	0	0	0	1

FIGURE 5.4: TARGET MATRIX LAYOUT

The next GUI window, divides the input data into training, validating and testing subsets. The training set is presented to the NN to adjust the weights and bias to improve classification. The validation step measures the generalization of the NN and halts training, when generalization no longer improves. The testing set does not affect training, rather provides an independent measure of the NN performance during and after training. The default values are 70% for training, 15% for validation and 15% for testing.

This tool also adjusts the NN architecture. Initially, the NN is a two-layer feed forward network with one sigmoid hidden layer and one softmax output layer. The number of neurons of the hidden layer is determined by the user, whereas the number of neurons in the output layer depends on the number of classes established by the target matrix (number of columns  $n$  in this case). The higher number of neurons in the hidden layer, the higher computation, and more tendency to overfitting the data; however, increasing this number might solve complex problems more efficiently. Therefore, several hidden layer settings are evaluated in this work to provide the best classification and generalization performance.

The remaining process involved in NN generation is NN training (*nntraintool*), in which weights and bias are adjusted so that the computed output data fit the target. The training procedure set by default is the scaled conjugate gradient back propagation (*trainscg*) that indicates how far and in which direction the currently computed outputs are with respect to the target. The goal of this training approach is to minimize the gradient with each epoch or NN processing cycle. Classification performance is measured by means of cross entropy (*crossentropy*). This method penalizes greatly misclassified outputs, whereas fairly correct outputs are assigned very little penalty. Thus, the smaller the cross entropy is, the better classification performance. Training method and performance measure can be modified in the MATLAB script but not in the GUI. During training, the GUI displays a new window summarizing the employed architecture and algorithms, as illustrated in Figure 5.5. In addition, training progress information and several plots measuring the performance are also available.

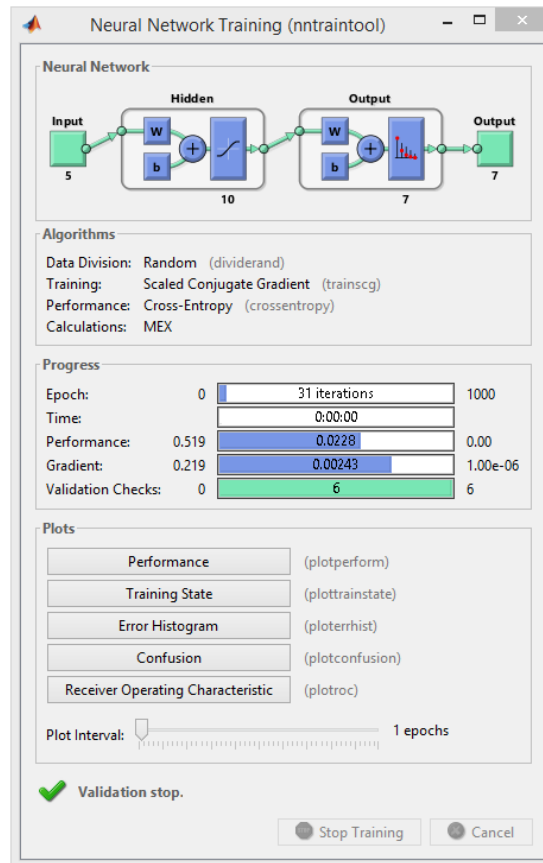


FIGURE 5.5: NEURAL NETWORK GUI TRAINING WINDOW

Results of the training process include the number of samples dedicated to training, validation and testing, along with cross entropy (CE) and Percent Error (%E) as indicated in Figure 5.6.

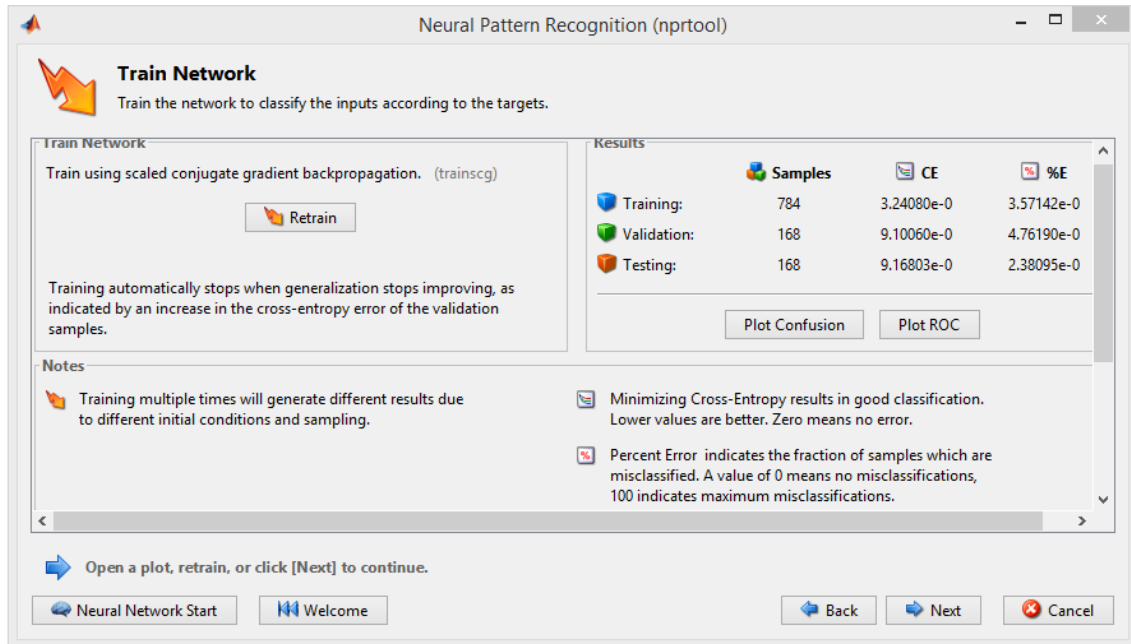


FIGURE 5.6: TRAINING RESULTS OF THE NEURAL NETWORK

If the obtained results are not satisfactory, the application enables retraining, modifying NN architecture, import larger data sets and perform additional tests. This last option allows the NN testing with new different data sets, calculating also their CE and %E.

The present work has benefited from the toolbox deployment alternatives to implement the generated trained NN into a Simulink diagram. Hence the NN classifier can be integrated into the Simulink control model.

As previously mentioned, the final step in the GUI application allows the creation of a MATLAB script implementing the generated NN, in addition to saving the obtained MATLAB network object, performance information, outputs and errors in the Workspace.

All these offline processing steps (Figure 5.7) generate a trained NN classifier, based on the reduced features extracted from the recorded signals, which can be incorporated into Simulink’s control environment.

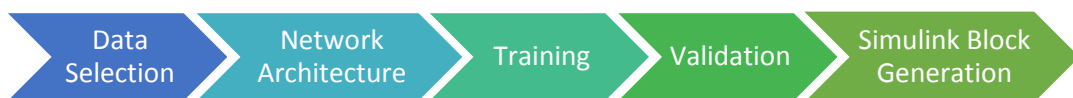


FIGURE 5.7: STEPS TO GENERATE A TRAINED NN SIMULINK MODEL

#### 5.4. SIMULINK CONTROL MODEL OF THE NNMCS

Simulink’s graphical programming is a powerful and intuitive tool to develop Model-Based design control systems. The integration of all previous steps: EMG acquisition, segmentation, feature extraction, LDA dimensionality reduction, NN classifier and hand

actuation control; not only improves the robustness of the system but also enable model simulation and testing. The implemented system can be considered as a more complex signal processing model than the one employed just for EMG signal acquisition. Therefore, the EMG Acquisition block of the Host model and the Target model will be the same. Figure 5.8 depicts the unification of all pattern recognition steps into the Host model.

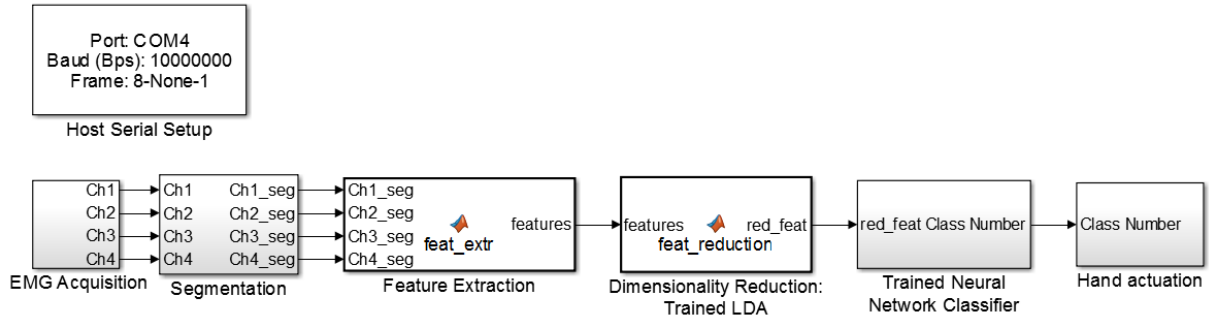


FIGURE 5.8: SIMULINK HOST MODEL OF THE NNMCS

Firstly, *Host Serial Setup* is configured to provide a communication port between the PC and the STM32F4 Discovery. In this case, the microcontroller sends the EMG signals to the PC Host model and receives the corresponding motion command for hand actuation, yielding in a two-direction communication flow.

As previously mentioned, the EMG Acquisition block has the same internal architecture as the model explained in Chapter 4, see Figure 5.9. Target model reads the EMG signals from the analog circuit and sends them to the Host model. This data transmission is controlled by *USB VCP Send STM32F4* and *Host Serial Rx*, in the Target and Host model respectively. The input signals are converted to single data type and filtered digitally with the high-order Butterworth bandpass filter implemented in *Digital Filter Design* block.

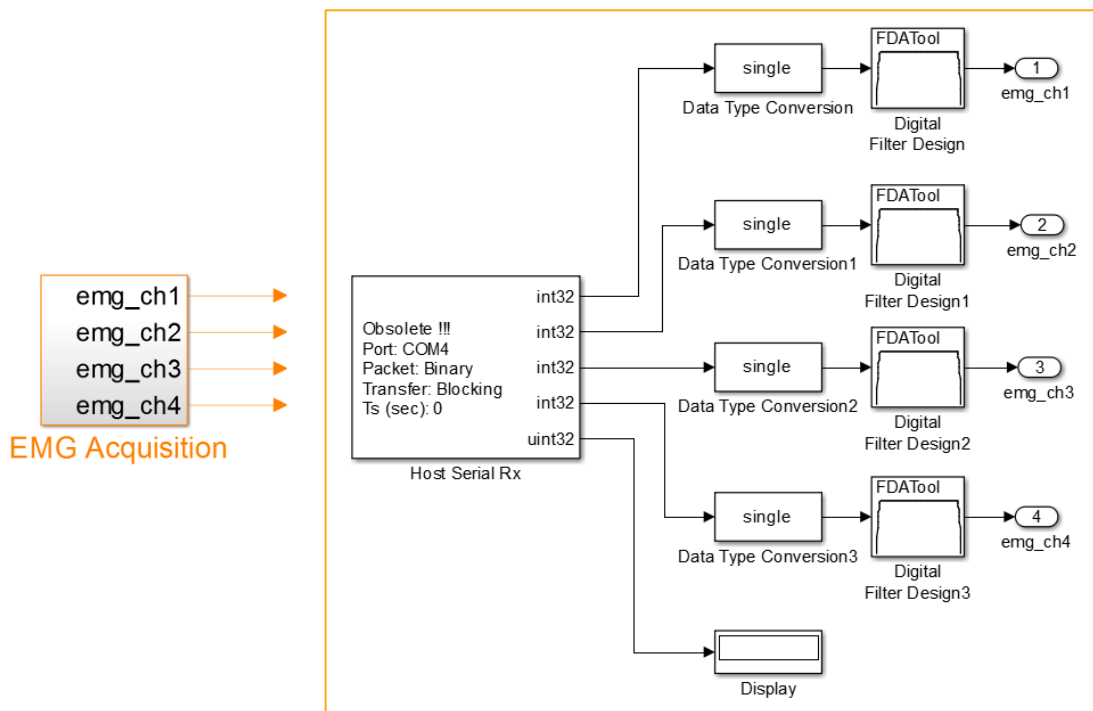


FIGURE 5.9: EMG ACQUISITION BLOCK AND INTERNAL DIAGRAM

The next step is signal segmentation; however, the MATLAB methodology used in offline training no longer applies for Simulink’s real-time signal processing. In the Simulink model, the EMG signals are segmented using a *Tapped Delay Line* block for each channel. These blocks do not output any signal until  $n$  number of samples (150, in this case) are collected. As MATLAB external functions (*feat\_extr* and *feat\_reduction*) output double precision data, a *Data Type Conversion* block has been added to shortcut this transformation, as represented in Figure 5.10.

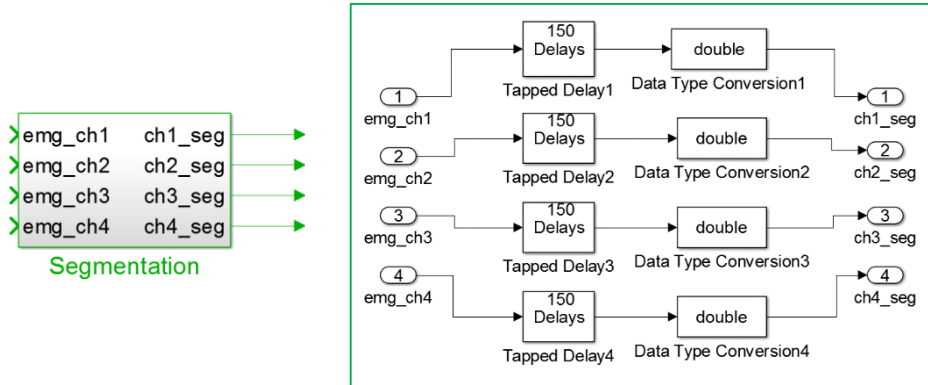


FIGURE 5.10: SEGMENTATION BLOCK AND ITS INTERNAL DIAGRAM

The following processing stages are *MATLAB function* blocks, which generate the corresponding MATLAB function embedded code for Simulink environment. Feature Extraction block creates a feature vector following the same procedure as in offline training, adapting to the new input data size. In Dimensionality Reduction, *feat\_reduction* employs the mapping configuration of the NN training data set to reduce the dimension of the feature vector, applying *out\_of\_sample drtoolbox* function.

The Trained Neural Network Classifier block is responsible for gesture identification and comprises two sub blocks: the trained NN itself and a class identifier; as illustrated in Figure 5.11. The trained NN is the Simulink block generated with the *nprtool* GUI/ script. It receives the reduced features and outputs a 1x7 vector representing the probability of the analyzed signal to be classified in each motion. Therefore, a class identifier is required to detect the maximum probability and send the associated gesture number (*Maximum* block index output) to the Hand actuation control.

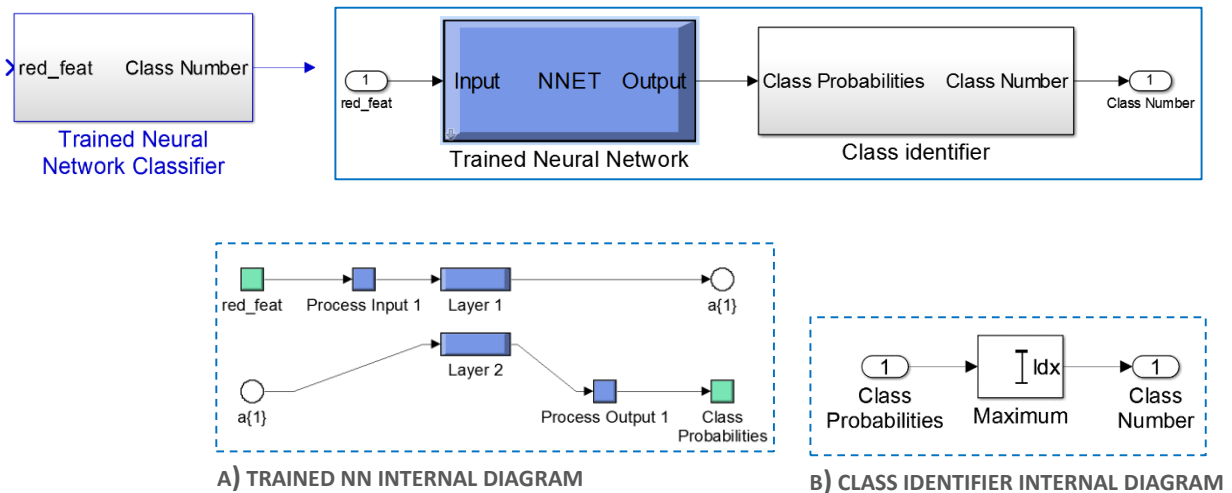


FIGURE 5.11: TRAINED NEURAL NETWORK CLASSIFIER BLOCK AND INTERNAL MODELS

The final Hand actuation block belongs to the low-level control of the robotic hand developed by E. Marín [58]. The internal diagram (Figure 5.12) is based on a *Multiport Switch* with the same number of data ports as feasible movements. An additional control input receives the motion number generated by the Class Identifier of the Trained NN Classifier, and outputs the corresponding motion information. The asterisk in Rest motion input, not only indicates that it is the default gesture, but also the selected position if an error in motion selection occurs. The outputted data comprises the angular position of each servomotor of the robotic hand, which is sent to the STM32F4 for hand actuation with another *Host Serial Tx* block.

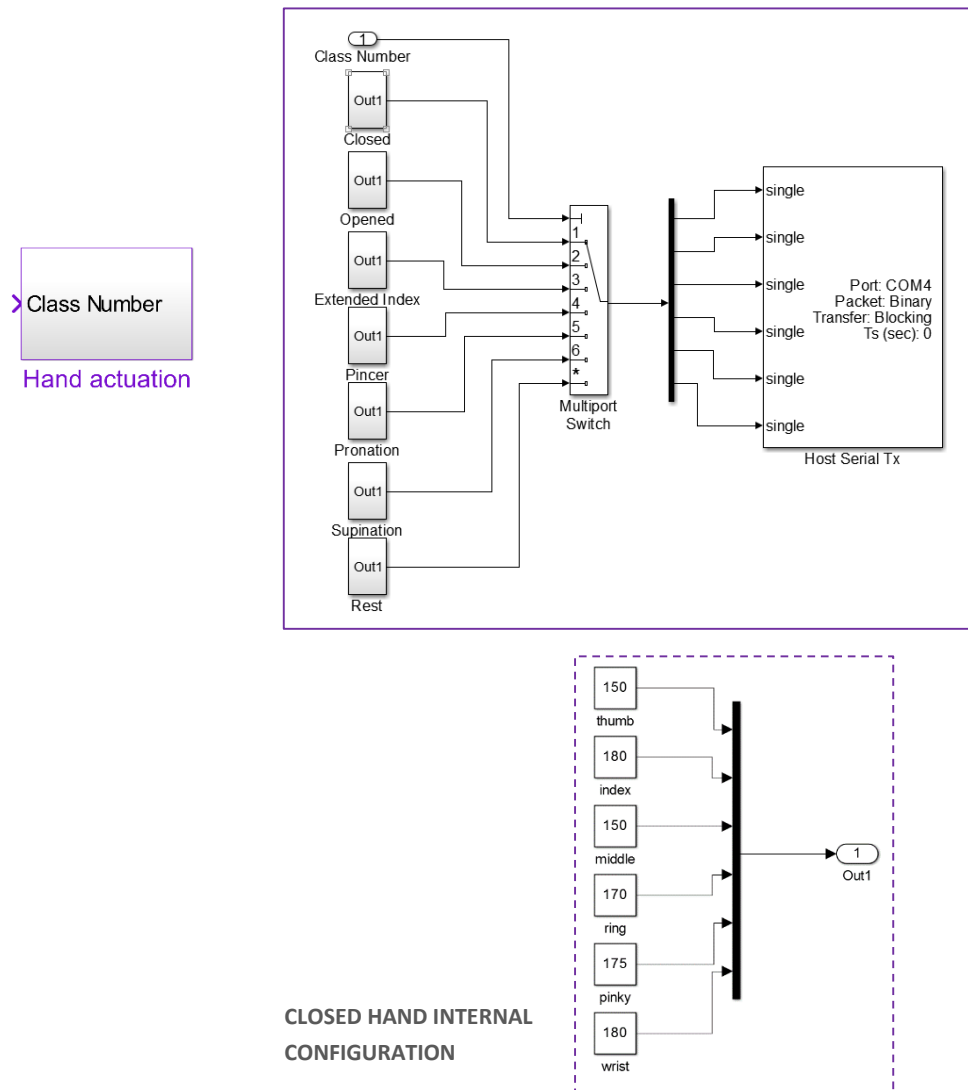


FIGURE 5.12: HAND ACTUATION BLOCK AND INTERNAL DIAGRAMS

This chapter has explained how the pattern recognition steps (signal acquisition, segmentation, feature extraction, dimensionality reduction and classification) have been integrated into an offline MATLAB training program and the NNMCS in Simulink. The remaining steps in real-time control development, involve system simulation and testing.



# CHAPTER 6

## 6. TESTS AND RESULTS

In this chapter several NN and data configurations are tested and discussed. The idea of this evaluation is to provide the optimal parameters to implement the NNMCS. Once the best NN configuration has been identified and implemented, it has been evaluated for real-time application and tested with data sets recorded from different sessions.

### 6.1. OPTIMAL NEURAL NETWORK CONFIGURATION

To determine the optimal NN, a *nprtool* script has been generated so that training algorithm and performance function can be modified. In addition, the code trains 10 NN and average their outputs to increase the generalization of the classifier. In order to evaluate the different NN setups, 5 sets of EMG signals have been recorded in the same session with 5 repetitions of each motion. The tests have focused on: the number of gesture repetitions required to train the NN efficiently, the optimal number of LDA reduced dimensions, the training algorithm and performance function, and the NN architecture or number of hidden neurons.

#### 6.1.1. MOTION REPETITIONS OF THE NEURAL NETWORK TRAINING SET

The main problem of NN classifiers is overfitting, which is characterized by a small error in classification accuracy of the training data set, but a significant error increase when evaluating the NN with new data sets. To solve this, the generalization of the classifier must be improved so that the NN does not memorize the training problem. To achieve this, four training sets with different motion repetitions have been employed in NN training. For the present work, the first issue in NN optimization is determining the ideal number of gesture repetitions required to generalize pattern identification.

Five EMG sets are recorded with five repetitions per movement. These sets have been combined and processed to produce larger data inputs for NN training:

- Training Set 5: 5 repetitions
- Training Set 10: 10 repetitions, combining sets 1 and 2
- Training Set 15: 15 repetitions, combining sets 1,2 and 3
- Training Set 20: 20 repetitions, combining sets 1-4

To test the generated NN independently, the remaining data sets have been used yielding in:

- Testing Set 5: 5 repetitions corresponding to the fifth set
- Testing Set 10: 10 repetitions after fusing sets 4 and 5. Notice that Training Set 20 is not tested with this set, as it already includes set 4.

The next step is determining the dimensions of the LDA dimensionality reduction to process these signals.

### 6.1.2. LDA REDUCED DIMENSIONS

Dimensionality reduction determines the relevant information to represent a feature set, reducing the complexity of the classification problem. Nevertheless, there is a tradeoff between the reduced computational load due to this simplification, and the quality of the mapping. *lda* function of *drtoolbox*, has been employed to assess the feasible dimensions in which the feature matrix can be mapped.

For every NN training set, an LDA mapping is generated for each possible dimension. This mapping is inputted into *out\_of\_sample* function to dimensionally reduce the testing sets. These sets are then fed into the NN to analyze their performance.

### 6.1.3. TRAINING ALGORITHM

In the performed experiments, two training methods are compared: Scaled Conjugate Gradient (*trainscg*) and Bayesian Regularization (*trainbr*) backpropagation.

As previously mentioned, Scaled Conjugate Gradient is the default method, which is based on the gradient to classify the input data. With this method (*trainscg*), data is divided into different subsets: 75% for training, 15% for validation and 15% for testing. This division is especially relevant for early stopping method, which halts the NN training when the validation error increases. The performance is evaluated by means of Cross Entropy (*crossentropy*).

On the other hand, Bayesian Regularization (*trainbr*) is an automatized regularization method that minimizes the linear combination of squared errors and weights to create a NN with high generalization. In this case, the performance function is the Mean Squared Error (*mse*). Bayesian regularization takes place within the Levenberg-Marquardt algorithm, which estimates the quadratic approximation of the problem. This process is iteratively refined until convergence, so it is not advisable to divide the data and end the training process with early stopping. During testing, the default parameters of *trainbr* have been used, being most relevant ones:

- Epoch: 1000
  - Performance goal (perf): 0
  - Minimum performance gradient (min\_grad): 1e-7
  - Maximum mu (mu\_max): 1e10
- Mu is the Marquardt adjustment parameter, which guides the optimization process. It changes during each iteration decreasing if the sum of squared errors decreases, and increasing otherwise.

For the present configuration, the training process is halted when one of the following conditions is satisfied:

- Maximum number of epoch is reached.
- Performance is minimized to the goal (perf).
- Performance gradient falls below min\_grad.
- Mu exceeds mu\_max.

Therefore, the duration of the offline NN training process is expected to be longer for *trainbr* than *trainscg*.

#### 6.1.4. NUMBER OF HIDDEN NEURONS

Once the optimal number of movement repetitions required for training and reduced dimensions have been assessed, several NN architectures are tested. The evaluated NNs are two-layered feedforward networks with varying number of hidden neurons (3, 5, 7, 10 and 15). For the first part of the testing process the default configuration of 10 hidden neurons has been applied. This evaluation aims to simplify the NN the maximum possible without compromising its classification accuracy and generalization. The 15 hidden neuron setup has been also included to increase the flexibility of the network and check if there are complex data relationships and how they affect classification accuracy and generalization. Higher values of hidden neurons have not been employed due to the increase in complexity and the tendency to overfit the data.

#### 6.2. DIFFERENT SESSION TESTING

This final testing step aims to quantify the generalization and robustness of the implemented NN for testing sets recorded during different sessions. These signals have been processed following the same protocol of the previously evaluated sets.

Two sets of 5 motion repetitions are acquired on two different days to obtain Testing Set 5 Day 1.a and 1.b, and Testing Set 5 Day 2.a and 2.b. Both signals are dimensionally reduced to the optimal number of dimensions using the mapping of the best training set. Finally they are inputted into the generated NN to analyze the classification accuracy.

#### 6.3. EVALUATION FOR REAL-TIME APPLICATION

Throughout the development of the NNMCS, meeting real-time constrains have been the bare essential. After determining the best NN setup for EMG pattern recognition, the implemented NN has been evaluated for its suitability for real-time control systems.

In this test, not only the time spent by the NN in the decision-making process is measured, but also the time required to segment the acquired signal, extract the features and reduced them to the optimal dimensions.

This evaluation is performed with MATLAB *profile* function, which calculates the execution time of a MATLAB code and enables tracking the time spent in each child function. In this case, each processing step has been implemented in the following functions: *segmentation*, *feature\_extraction*, *feature\_reduction* and *NN\_classification*. The program is fed with just one 150-sample segment for each channel of a recorded signal, so the Simulink segmentation process is simulated with a delay of 150 ms. The final result is then, compared to the real-time constrain of 300 ms.

## 6.4. RESULTS

### 6.4.1. RESULTS OF OPTIMAL NEURAL NETWORK CONFIGURATION

Tables 6.1 to 6.4 represent the classification accuracy and generalization results obtained for the different training sets reducing their dimensions to 2-6 LDA dimensions. The reduced features have been analyzed for both *trainscg* and *trainbr* algorithms. Training sets provide a measure of classification accuracy of the trained NN. Generalization is analyzed from the classification accuracy of the testing data sets. The higher these percentages are, the better the NN generalizes and fits new data.

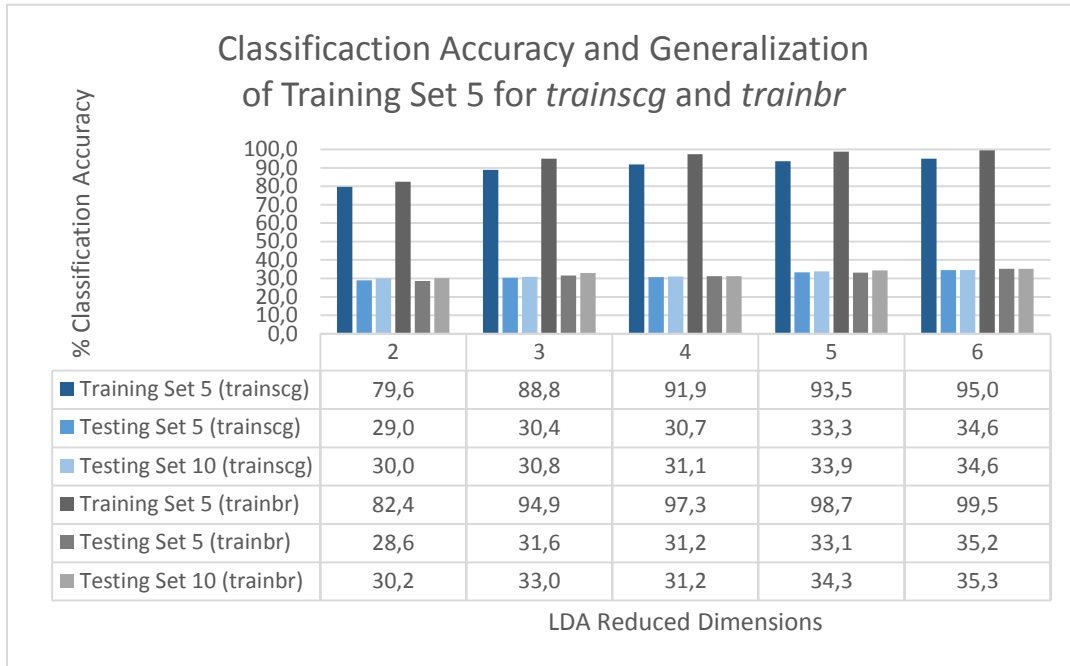


TABLE 6.1: CLASSIFICATION ACCURACY AND GENERALIZATION OF TRAINING SET 5 FOR *TRAINSCG* AND *TRAINBR*

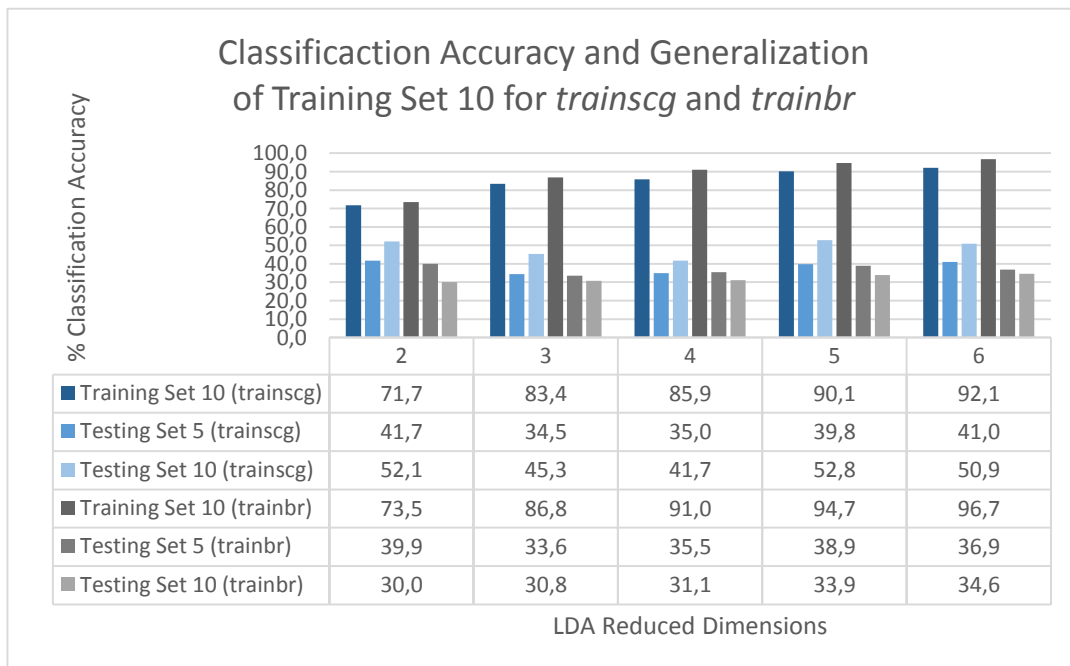
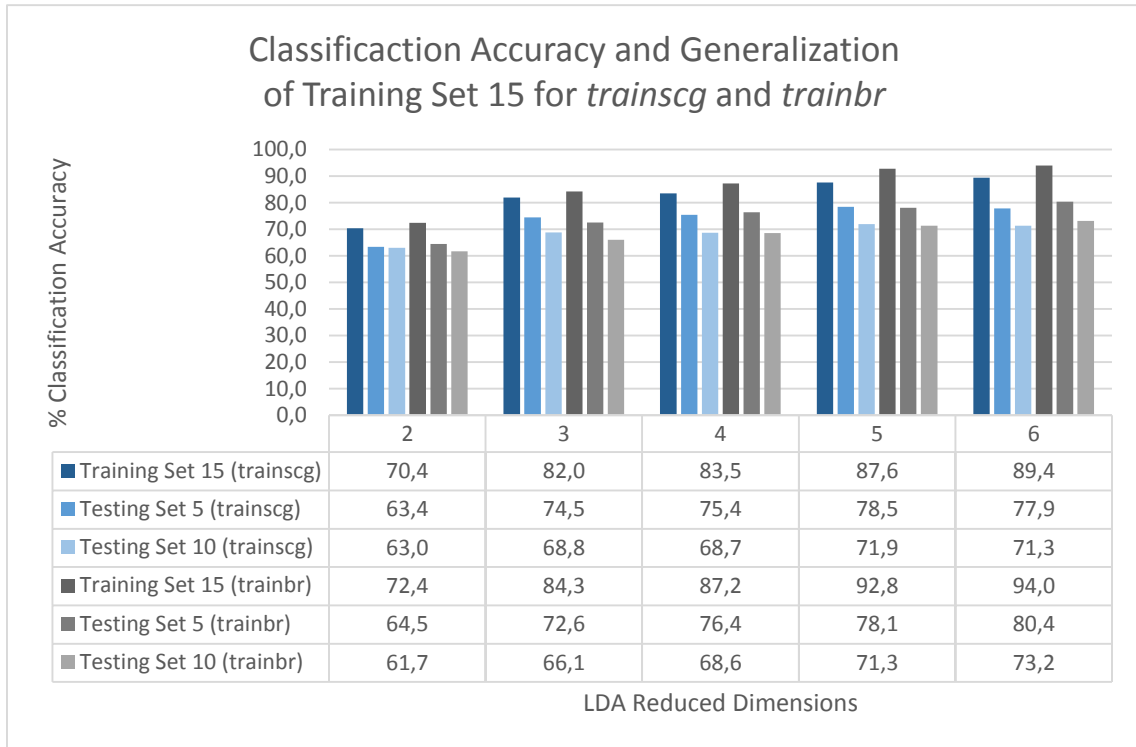
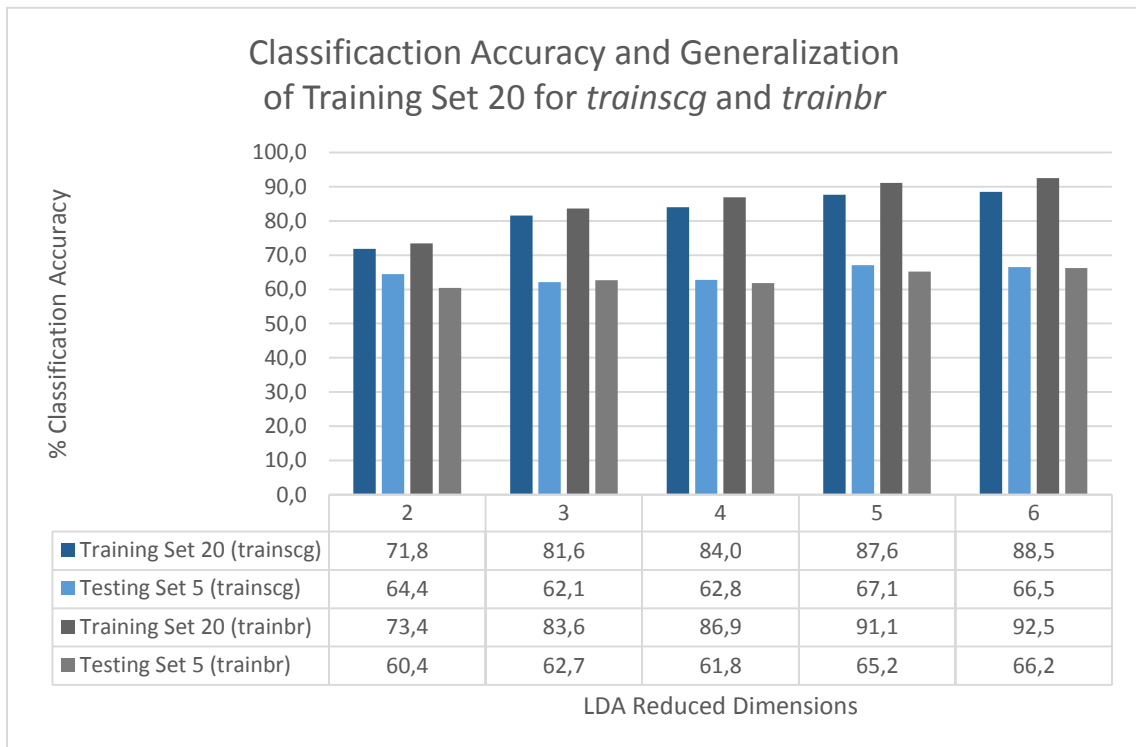


TABLE 6.2: CLASSIFICATION ACCURACY AND GENERALIZATION OF TRAINING SET 10 FOR *TRAINSCG* AND *TRAINBR*



**TABLE 6.3: CLASSIFICATION ACCURACY AND GENERALIZATION OF TRAINING SET 15 FOR *TRAINSCG* AND *TRAINBR***



**TABLE 6.4: CLASSIFICATION ACCURACY AND GENERALIZATION OF TRAINING SET 20 FOR *TRAINSCG* AND *TRAINBR***

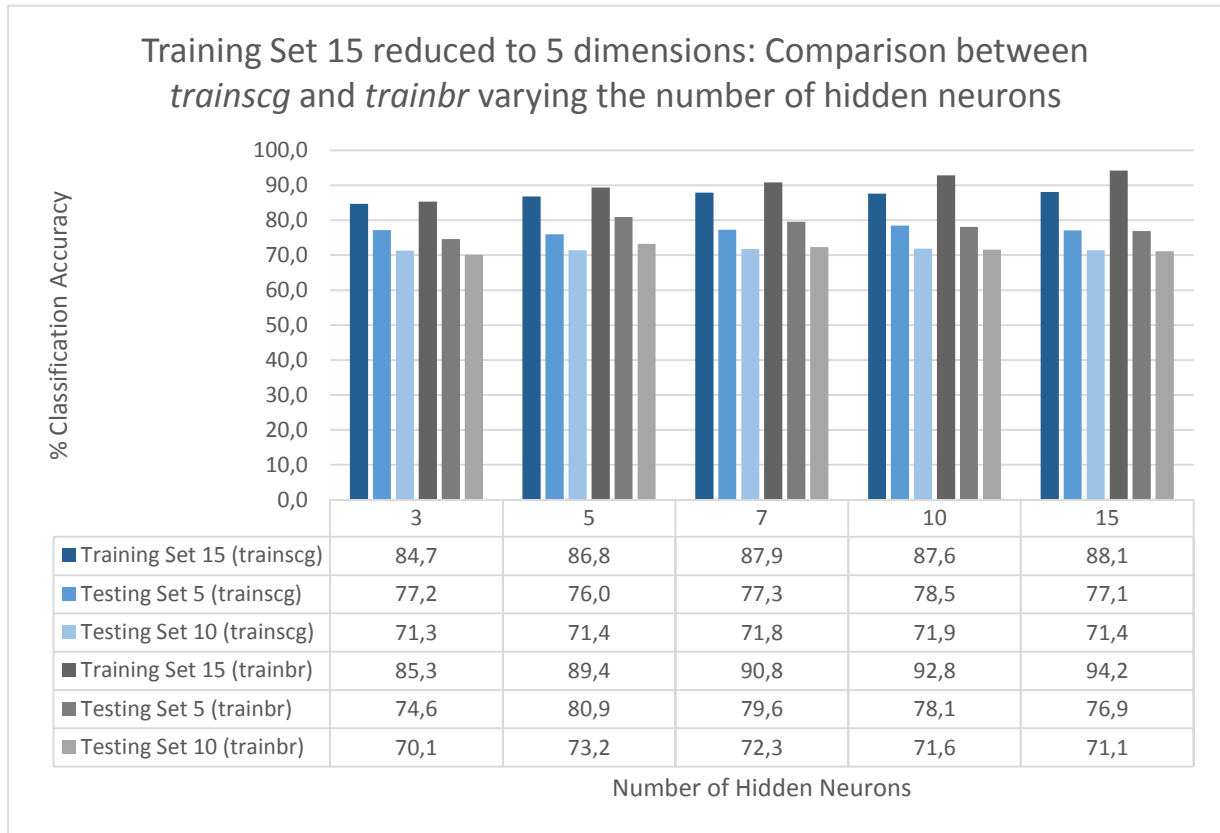
Results confirm that for the same training set, the higher LDA dimensions, the better mapping and the higher classification accuracy; reaching a maximum of 99.5 % training the NN with Set 5 reduced to 6 dimensions and *trainbr* algorithm.

The highest classification percentages are achieved for the smallest training sets (5 and 10). However, these sets do not recognize motion patterns properly when being tested with new data, ranging from 30-52.8% of classification accuracy, showing overfitting and poor generalization.

In general, both training algorithms generalize similarly for the same testing sets when applied to the same trained NN; nevertheless, *trainbr* exhibits higher classification accuracy of the training sets.

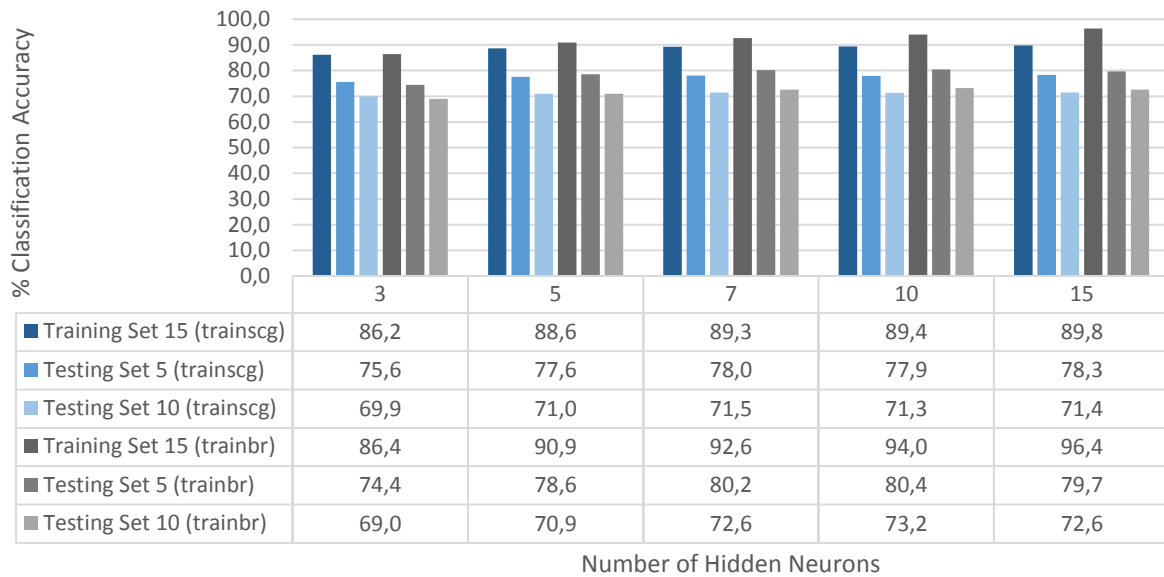
The highest generalization is accomplished with Training Set 15. Training algorithm *trainscg* show better performance for 5 LDA dimensions, whereas *trainbr* behaves better with 6 dimensions.

In order to determine the best NN architecture, both training algorithms have been analyzed for Training Set 15 reduced to 5 and 6 dimensions, varying the number of hidden neurons. The obtained results, depicted in Tables 6.5 and 6.6, do not follow a clear pattern when changing the number of hidden neurons. Whereas *trainscg* generalizes better with 10 or 15 hidden neurons for reduced dimensions 5 and 6, respectively; the overall best generalization is achieved with *trainbr* and 5 LDA dimensions. This configuration employs one neuron for each input dimension, similarly to the matching number of hidden neurons and number of outputs of the output layer.



**TABLE 6.5: TRAINSCG AND TRAINBR COMPARISON FOR TRAINING SET 15 REDUCED TO 5 DIMENSIONS VARYING THE NUMBER OF HIDDEN NEURONS**

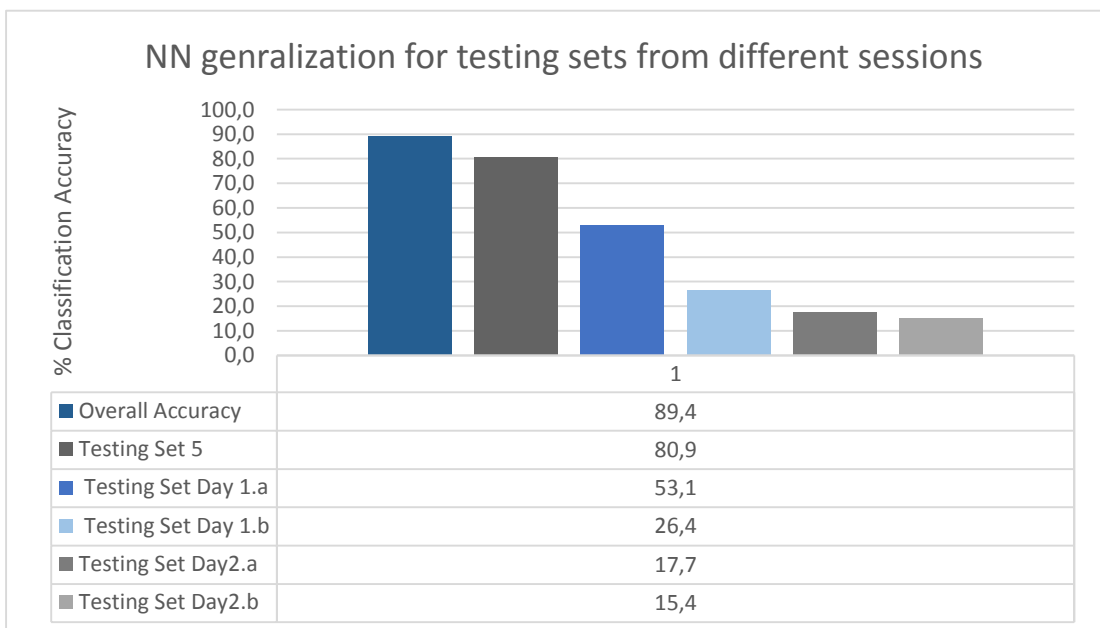
### Training Set 15 reduced to 6 dimensions: Comparison between *trainscg* and *trainbr* varying the number of hidden neurons



**TABLE 6.6: TRAINSCG AND TRAINBR COMPARISON FOR TRAINING SET 15 REDUCED TO 6 DIMENSIONS VARYING THE NUMBER OF HIDDEN NEURONS**

#### 6.4.2. RESULTS OF DIFFERENT SESSION TESTING

Table 6.7 compares the generalization of the optimized NN configuration, 5 hidden neurons using Training Set 15 reduced to 5 dimensions and *trainbr* algorithm, for two testing sets recorded on different days. The obtained results show a significant decrease in classification accuracy for different session testing sets, ranging from 15.4-53.1%. This poor generalization can be caused by physiological changes or slight differences in electrode placement.



**TABLE 6.7: NN GENERALIZATION EVALUATION FOR TESTING SETS OF DIFFERENT SESSIONS**

### 6.4.3. RESULTS OF REAL-TIME APPLICATION EVALUATION

The *profile* time evaluation depicted in Table 6.8, shows that the system does not meet real-time requirements as its total elapsed time is greater than 300 ms, actually being 527 ms. Although the computational load of feature extraction and dimensionality reduction has been efficiently fixed to 73 ms, the time required to classify the gestures using the NN consumes almost the real-time limit.






Function Name	Function Type	Calls	Total Time	% Time	Time Plot
<a href="#">NN_classification</a>	function	1	0.282 s	53.5%	
<a href="#">segmentation</a>	function	1	0.156 s	29.7%	
<a href="#">feat_extraction</a>	function	1	0.055 s	10.4%	
<a href="#">feat_reduction</a>	function	1	0.018 s	3.4%	
Self time (built-ins, overhead, etc.)			0.016 s	3.0%	
Totals			0.527 s	100%	

TABLE 6.8: EXECUTION TIME OF EACH PROCESSING STEP OF THE NNMCS FOR ONE ITERATION

The overall evaluation of the performed tests is explained in the next chapter, along with the conclusions reached after the development of the control system.



# CHAPTER 7

## 7. CONCLUSIONS AND FUTURE WORK

This section analyzes the performed steps into the implementation of a NNMCS, and discuss its suitability for hand motion pattern recognition and robotic hand actuation in real-time. Several guidelines, extracted from these conclusions, are also proposed, aiming to help and orient further research in this topic.

### 7.1. CONCLUSIONS

Regarding the initially proposed objectives, most of them have been accomplished, as the implemented NNMCS is able to acquire, process and classify EMG signals to reproduce hand motions with enough accuracy; integrating the EMG acquisition system, NN pattern recognition based control system and robotic hand actuation into just one Simulink model. However the designed system fails to satisfy real-time constraints, being the most important issue for future work improvements.

The employed acquisition methodology, which records the stationary components of forearm EMG signals using four channels, and divides them into disjoint segments of 150 ms, not only enables feature extraction to characterize each motion but also transitions between gestures.

Furthermore, the time-domain selected features have proven low computational load, as well as the LDA dimensionality reduction algorithm. The elapsed time has been minimized during these pre-processing steps to prepare the EMG data for the NN.

The rationale under the classification of seven hand and wrist movements with a MATLAB/Simulink NN has proven to be partially successful. Although it does enable the incorporation of all the technology involved in myoelectric pattern recognition and robotic hand actuation; it has not been able to adjust to real-time limitations and solve completely the generalization problem.

The optimization of the NN inputs and its internal architecture has been experimentally adjusted, depending on the number of motion repetitions employed in NN training, number of reduced dimensions, training and performance algorithms and number of neurons in the hidden layer. The present study has focused on these parameters to evaluate the behavior of the NN for pattern recognition; however, multiple setups can also be modified with a deep knowledge in MATLAB Neural Network Toolbox to examine the classification accuracy more profoundly.

The foremost NN configuration employs one neuron per input/ output, yielding in a two-layered feedforward network with 5 hidden neurons that match the 5 LDA reduced dimensions, and 7 output neurons for the 7 possible movements. The implemented NN has been trained with Bayesian Regularization (associated to Mean Squared Error performance function) for 15 gesture repetitions, yielding in high classification accuracy (89.4%) and relative good performance for testing sets recorded on the same session (80.9%).

Nevertheless, testing signals acquired on different sessions have shown low and random classification performance. This misclassification increase is thought to be caused by the stochastic nature of the EMG signals, varying easily from one session to another; not only due to physiological changes but also if electrode placement is not exactly the same.

The analysis of the performed tests reveal a tradeoff between generalization improvement and the required time to classify the NN input signal. These are the two key factors in pattern recognition, representing the sensitivity and speed of the control system. Results show that NN decision making process alone, barely fits real-time restrictions. Although all the previous processing steps have been implemented effectively with low computational load; the execution time of the system surpasses by 200 ms the real-time constraints.

Therefore, the principal conclusion elicited from the present work is that MATLAB/ Simulink Neural Networks can indeed recognize hand motion patterns based on EMG signals, and be integrated into a control system model for the actuation of a printable robotic hand. Still, training process is a user specific task that requires manual refinement until the optimal setup is found; and execution time must be reduced to overcome the present lag.

## 7.2. FUTURE WORK

The formerly derived conclusions suggest several work lines to improve the present study, and bring closer the implementation of this technology in the real world, outside the laboratory.

The first and foremost proposal is to research towards the fulfillment of real-time requirements. As previously mentioned, MATLAB Neural Network Toolbox provide an extensive set of configuration parameters that can be evaluated to reduce the execution time; like the computational features associated to training functions, or applying instead a transfer function to map NN input data into their corresponding output.

However, changing these parameters may affect the classification accuracy and generalization of the NN classifier. Hence, further analysis should be carried out to clearly assess the crucial parameters, from both input data and NN, involved in NN performance and how they affect the elapsed time. Special considerations should be taken into account regarding dimensionality reduction algorithms and their generated mappings as they are the base of the training process. Once all these specifications have been optimized, the next advisable step is their directly implementation into a microcontroller to reduce the execution time.

If none of the mentioned modifications of the NN yield in an improvement of the control system, there are other classifiers that can be also examined and are already integrated in MATLAB environment like Support Vector Machines (SVM).

The present work requires EMG signal acquisition and NN training for each session, which implies the placement of nine surface electrodes in the precise locations. This arrangement is not only tedious for the user but can also produce significant changes in the recorded signals, reducing greatly the generalization ability of the classifier. Therefore, developing a unified recording system that integrates the nine electrodes and perfectly fits the subject,

so that the electrodes are placed correctly and easily regardless the session, could be quite advantageous to the system.

To conclude, the present bachelor thesis can be interpreted as the first step towards the development of a robust, real-time, neural network-based control system in MATLAB/Simulink. The present work has faced both promising results and severe problems; however the overall analysis of this chapter aims to provide a solid base to elucidate the advantages and drawbacks of this technology, to guide future work in this field and improve the model.



## REFERENCES

- [1] Konrad, Peter. *The ABC of EMG: A Practical Introduction to Kinesiological Electromyography*. Version 1.4 ed. Version 1.4 ed. N.p.: Noraxon INC. USA., Mar. 2006.
- [2] Rechy-Ramirez, Ericka Janet, and Huosheng Hu. *Stages for Developing Control Systems Using EMG and EEG Signals: A Survey*. United Kingdom: School of Computer Science and Electronic Engineering University of Essex, 29 June 2011.
- [3] Martini, Frederic H., Michael J. Timmons, and Robert B. Tallitsch. "The Muscular System: Skeletal Muscle Tissue and Muscle Organization." *Human anatomy*. United States: Benjamin-Cummings Publishing Company, Subs of Addison Wesley Longman, 3 Jan. 2014. 248–249. Print.
- [4] Luca, Carlo J. De. "Physiology and Mathematics of Myoelectric Signals." *Biomedical Engineering, IEEE Transactions on BME-26.6* (June 1979): 313–325. Web. 31 Dec. 2015.
- [5] Luca, Carlo J De. *Surface Electromyography: Detection and Recording*. N.p.: DelSys Incorporated, 2002.
- [6] Tortora, Gerard J, and Bryan H Derrickson. *Principles of Anatomy & Physiology*. 13th ed. United Kingdom: Wiley, John & Sons, 5 Jan. 2011.
- [7] OpenStax College. "The Muscular System." *Anatomy & physiology*. Houston, United States: Rice University, 25 Apr. 2013. 446–449. Print.
- [8] Day, Scott. *Important Factors in Surface EMG Measurement*. Calgary, Canada: Bortec Biomedical Ltd, 27 July 2004. Web. 27 Jan. 2016.
- [9] Englehart, K., and B. Hudgins. "A Robust, Real-Time Control Scheme for Multifunction Myoelectric Control." *Biomedical Engineering, IEEE Transactions on* 50.7 (July 2003): 848–854. Web. 14 Jan. 2016.
- [10] Oskoei, M.A., and Huosheng Hu. "Support Vector Machine-Based Classification Scheme for Myoelectric Control Applied to Upper Limb." *IEEE Transactions on Biomedical Engineering* 55.8 (Aug. 2008): 1956–1965. Web.
- [11] Christodoulou, C.I., and C.S. Pattichis. "Unsupervised Pattern Recognition for the Classification of EMG Signals." *IEEE Transactions on Biomedical Engineering* 46.2 (1999): 169–178. Web.
- [12] Gut, R., and G.S. Moschytz. "High-Precision EMG Signal Decomposition Using Communication Techniques." *IEEE Transactions on Signal Processing* 48.9 (2000): 2487–2494. Web.
- [13] Kaur, G., A.S. Arora, and V.K. Jain. "Comparison of the techniques used for segmentation of EMG signals." *Proceedings of the 11th WSEAS international conference on Mathematical and computational methods inscience and engineering*. N.p.: World Scientific and Engineering Academy and Society (WSEAS), 2009. 124–129. Web.
- [14] Oskoei, Mohammadreza Asghari, and Huosheng Hu. "Myoelectric Control systems—A Survey." *Elsevier* 2.4 (Oct. 2007): 275–294. Web. 14 Jan. 2016.
- [15] Hudgins, B., P. Parker, and R.N. Scott. "A New Strategy for Multifunction Myoelectric Control." *IEEE Transactions on Biomedical Engineering* 40.1 (1993): 82–94. Web.

- [16] Englehart, K., B. Hudgin, and P.A. Parker. "A Wavelet-Based Continuous Classification Scheme for Multifunction Myoelectric Control." *IEEE Transactions on Biomedical Engineering* 48.3 (Mar. 2001): 302–311. Web.
- [17] Zecca, M., et al. "Control of Multifunctional Prosthetic Hands by Processing the Electromyographic Signal." *Critical ReviewsTM in Biomedical Engineering* 30.4-6 (2002): 459–485. Web. 31 Dec. 2015.
- [18] Karlsson, S., Jun Yu, and M. Akay. "Time-Frequency Analysis of Myoelectric Signals during Dynamic Contractions: A Comparative Study." *IEEE Transactions on Biomedical Engineering* 47.2 (2000): 228–238. Web.
- [19] Rechy-Ramirez, Ericka Janet, and Huosheng Hu. "Bio-Signal Based Control in Assistive Robots: A Survey." *Digital Communications and Networks* 1.2 (Apr. 2015): 85–101. Web.
- [20] Englehart, K, et al. "Classification of the Myoelectric Signal Using Time-Frequency Based Representations." *Medical Engineering & Physics* 21.s 6–7 (1 July 1999): 431–438. Web. 18 Jan. 2016.
- [21] Phinyomark, A., C. Limsakul, and P. Phukpattaranont. "A Novel Feature Extraction for Robust EMG Pattern Recognition." *Journal of Computing* 1.1 (Dec. 2009): n.pag. Web. 28 Jan. 2016.
- [22] Huang, H. P., and C. Y. Chen. "Development of a Myoelectric Discrimination System for a Multi-Degree Prosthetic Hand." *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on* 3. (1999): 2392 – 2397. Web. 20 Jan. 2016.
- [23] Wang, Xuechuan. "Feature extraction and Dimensionality reduction in pattern recognition and their application in speech recognition." 2002. Web. 18 Jan. 2016.
- [24] Oskoei, M. A., and H. Hu. "GA-based feature subset selection for Myoelectric classification." *International Conference on Robotics and Biomimetics*. Kunming, China: IEEE, Dec. 2006. 1465–1470. Web.
- [25] Oskoei, M. A., and H. Hu. "GA-based feature subset selection for Myoelectric classification." *International Conference on Robotics and Biomimetics*. Kunming, China: IEEE, Dec. 2006. 1465–1470. Web.
- [26] Fodor, I K. *A Survey of Dimension Reduction Techniques*. Springfield, United States: Lawrence Livermore National Laboratory, 9 May 2002. Web. 18 Jan. 2016.
- [27] Liu, Jie. "Feature Dimensionality Reduction for Myoelectric Pattern Recognition: A Comparison Study of Feature Selection and Feature Projection Methods." *Medical Engineering & Physics* 36.12 (1 Jan. 2016): 1716–1720. Web. 18 Jan. 2016.
- [28] Mitchell, T. M. "Bayesian Learning." *Machine learning*. New York: McGraw Hill Higher Education, 1 Apr. 1997. 156–158. Print.
- [29] Bu, N., M. Okamoto, and T. Tsuji. "A Hybrid Motion Classification Approach for EMG-Based Human–Robot Interfaces Using Bayesian and Neural Networks." *IEEE Transactions on Robotics* 25.3 (June 2009): 502–511. Web.
- [30] Haykin, S. "Introduction." *Neural networks and learning machines: A comprehensive foundation*. Ed. M. J. Horton. Harlow: Prentice Hall, 18 Nov. 2008. 1–46. Print.
- [31] Ahsan, Md. R., M. I. Ibrahimy, and O. O. Khalifa. "Neural Network Classifier for Hand Motion Detection from EMG Signal." *5th Kuala Lumpur International Conference on Biomedical Engineering 2011*. Ed. Noor Azuan Abu Osman, Wan Abu Bakar Wan Abas, Ahmad Khairi Abdul Wahab, and Hua-Nong Ting. Kuala Lumpur, Malaysia: Springer Science + Business Media, June 2011. 536–541. Web.

- [32] Gavin, Henri P. *The Levenberg-Marquardt Method for Nonlinear Least Squares Curve-Fitting Problems*. Durham, United States: Duke University, Department of Civil and Environmental Engineering, 29 Sept. 2015. Web. 26 Jan. 2016.
- [33] Subasi, A., M. Yilmaz, and H. R. Ozcalik. "Classification of EMG Signals Using Wavelet Neural Network." *Journal of Neuroscience Methods* 156.1-2 (Sept. 2006): 360–367. Web.
- [34] Chu, Jun-Uk, Inhyuk Moon, and Mu-Seong Mun. "A Real-Time EMG Pattern Recognition System Based on Linear-Nonlinear Feature Projection for a Multifunction Myoelectric Hand." *IEEE Transactions on Biomedical Engineering* 53.11 (Nov. 2006): 2232–2239. Web.
- [35] Sivanandam, S. N., S. Sumathi, and S. N. Deepa. *Introduction to Fuzzy Logic Using MATLAB*. Berlin: Springer-Verlag Berlin and Heidelberg GmbH & Co. K, 3 Nov. 2006. Print.
- [36] Ajiboye, A.B., and R.F.H. Weir. "A Heuristic Fuzzy Logic Approach to EMG Pattern Recognition for Multifunctional Prosthesis Control." *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 13.3 (Sept. 2005): 280–291. Web.
- [37] Xiong, Tao, and V. Cherkassky. "A Combined SVM and LDA Approach for Classification." *Proceedings. 2005 IEEE International Joint Conference on Neural Networks* 3. (2005): 1455 – 1459. Web.
- [38] Balakrishnama, S, and A Ganapathiraju. *Linear Discriminant Analysis - A Brief Tutorial*. Mississippi, United States: Institute for Signal and Information Processing, n.d. Web. 19 Jan. 2016.
- [39] Phinyomark, A., et al. "EMG Feature Evaluation for Improving Myoelectric Pattern Recognition Robustness." *Expert Systems with Applications* 40.12 (1 Jan. 4832): 4832–4840. Web. 15 Jan. 2016.
- [40] Chan, A.D.C., and K.B. Englehart. "Continuous Myoelectric Control for Powered Prostheses Using Hidden Markov Models." *IEEE Transactions on Biomedical Engineering* 52.1 (Jan. 2005): 121–124. Web.
- [41] Cunningham, P., and S. J. Delany. *K-Nearest Neighbour Classifiers*. N.p.: ResearchGate, 27 Mar. 2007. Web. 20 Jan. 2016.
- [42] Purushothaman, Geethanjali, and Kalyan Kumar Ray. "Motion Control of Drives for Prosthetic Hand Using Continuous Myoelectric Signals." *Journal of The Institution of Engineers (India): Series B* (24 Dec. 2014): 1–6. Web.
- [43] Kaufmann, Paul, et al. "Classification of Electromyographic Signals: Comparing Evolvable Hardware to Conventional Classifiers." *IEEE Transactions on Evolutionary Computation* 17.1 (Feb. 2013): 46–63. Web.
- [44] Radmand, Ashkan, et al. "Investigation of optimum pattern recognition methods for robust myoelectric control during dynamic limb movement." *36th Canadian Medical and Biological Engineering Conference*. N.p.: ResearchGate, May 2013. Web.
- [45] Jain, A.K., P.W. Duin, and Jianchang Mao. "Statistical Pattern Recognition: A Review." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.1 (Jan. 2000): 4–37. Web.
- [46] Fariman, H. J., et al. "Hand Movements Classification for Myoelectric Control System Using Adaptive Resonance Theory." *Australasian Physical & Engineering Sciences in Medicine* (18 Nov. 2015): n.pag. Web.

- [47] Duin, Robert P.W., and David M.J. Tax. "Experiments with Classifier Combining Rules." *Multiple Classifier Systems*. Ed. G. Goos, J. Hartmanis, and J. van Leeuwen. Cagliari, Italy: Springer Science + Business Media, June 2000. 16–29. Web.
- [48] Karlik, B., M.O. Tokhi, and M. Alci. "A Fuzzy Clustering Neural Network Architecture for Multifunction Upper-Limb Prosthesis." *IEEE Transactions on Biomedical Engineering* 50.11 (Nov. 2003): 1255–1261. Web.
- [49] Breiman, Leo. "Arcing Classifier." *The Annals of Statistics* 26.3 (June 1998): 801–849. Web.
- [50] García Martín Engeños, Ángel. "Sistema de Adquisición Multicanal para Señales Mioeléctricas." July 2015. Print.
- [51] Villoslada, Álvaro. "Design and Implementation of a Myoelectric Control System for a Printable Robotic Hand." July 2012. Print.
- [52] Karki, James. *Fully-Differential Amplifiers*. N.p.: Texas Instruments, Jan. 2002. Web. 31 Jan. 2016.
- [53] Winter, Bruce B., and John G. Webster. "Driven-Right-Leg Circuit Design." *IEEE Transactions on Biomedical Engineering* BME-30.1 (Jan. 1983): 62–66. Web.
- [54] STMicroelectronics. *UM1472 User Manual Discovery Kit for STM32F407/417 Lines*. <http://www.st.com/web/en/home.html>: STMicroelectronics, Jan. 2014. Web. 1 Feb. 2016.
- [55] Flores Caballero, Antonio. "Sistema avanzado de prototipado rápido para control en exoesqueletos y dispositivos mecatrónicos." Dec. 2014. Web. 1 Feb. 2016.
- [56] The MathWorks, Inc. "Simulink Documentation." *Mathworks*. n.d. Web: <http://es.mathworks.com/help/simulink/> 1 Feb. 2016.
- [57] Aimagin. "Waijung Blockset." *STM32F4 Target*. 21 Apr. 2015. Web: <http://waijung.aimagin.com/> 1 Feb. 2016.
- [58] Marín Conde, Esperanza. "Construction of a Robotic Hand for Myoelectric Control System Research Applied to Low-Cost Prostheses." Print.
- [59] Maaten, Laurens Van Der. "Matlab Toolbox for Dimensionality reduction." *Laurens Van Der Maaten*. n.d. Web. 9 Feb. 2016. <https://lvdmaaten.github.io/drtoolbox/>
- [60] Hudson Beale, Mark, Martin T. Hagan, and Howard B. Demuth. *Neural Network Toolbox™ User's Guide*. N.p.: The MathWorks, Inc., Sept. 2015. Web. 10 Feb. 2016.



# ANNEX

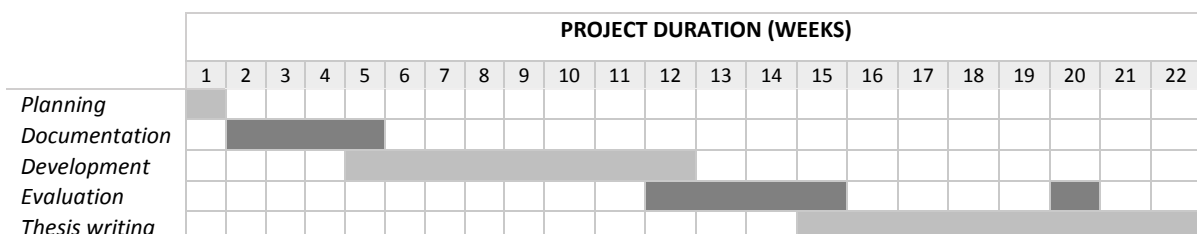
## A.1. WORK BREAKDOWN STRUCTURE

This section explains the different stages involved in the present thesis development and planning. Table A.1 summarizes the hours devoted to each phase. The project schedule is depicted in the Gantt chart of Table A.2

1. PROJECT PLANNING AND ORIENTATION (5 H)  
Series of meetings to determine topic of the project, the objectives and the organization.
2. DOCUMENTATION (50 H)  
This stage comprises the familiarization with the research on the topic, study of signal processing and learning how to use the different employed toolboxes.
3. MODEL DEVELOPMENT (75 H)  
This period includes the calibration of the EMG acquisition system and control model implementation, comprising: signal acquisition, signal processing, NN classification and integration in Simulink.
4. MODEL EVALUATION (40 H)  
Time devoted to test the implemented model and optimize the configuration.
5. WRITING THE THESIS (150 H)  
Bibliography documentation and writing of the bachelor dissertation.

<b>Project stages</b>	<b>Devoted hours</b>
<i>Planning and orientation</i>	5
<i>Documentation</i>	50
<i>Model development</i>	75
<i>Model evaluation</i>	40
<i>Thesis writing</i>	150
<b>TOTAL</b>	<b>320</b>

**TABLE A.1: BREAKDOWN OF WORK STRUCTURE AND DURATION**



**TABLE A.2: GANTT CHART**

## A.2. LIST OF MATERIALS

The materials employed during the present study include both hardware and software.

### 1. HARDWARE

- PC.
- STM32F4 Discovery microcontroller.
- 2 USB cables to communicate and power the STM32F4 Discovery.
- EMG acquisition board.
- Skintact FSTC1 Ag-AgCl ECG surface electrodes.

### 2. SOFTWARE

- Windows® 8.1 OS
- Mathworks® MATLAB/ Simulink
- Mathworks® Neural Network Toolbox
- Advanced Rapid Control Prototyping Toolbox for Simulink
- Waijung Toolbox for Simulink

## A.3. PROJECT BUDGET

### A.3.1. MATERIALS

<b>Description</b>	<b>Units</b>	<b>Unitary Price</b>	<b>Subtotal</b>	<b>Total</b>
<i>Skintact FSTC1 Ag-AgCl ECG surface electrodes</i>	60	0.12 €	7.30 €	7.30 €
<i>STM32F4 Discovery</i>	1	14.28 €	14.28 €	14.28 €
<i>USB cables</i>	2	3.95 €	7.90 €	7.90 €
<i>EMG acquisition board</i>	1	114.65 €	114.65 €	114.65 €
<b>TOTAL</b>				<b>144.13 €</b>

TABLE A.3: COST OF THE MATERIAL

### A.3.2. INFORMATICS

<b>Description</b>	<b>Unitary Price</b>	<b>Useful Life</b>	<b>Use</b>	<b>Cost</b>
<i>PC</i>	800 €	60 months	6 months	80 €
<i>MATLAB/ Simulink License (student)</i>	69.00 €	48 months	6 months	8.63 €
<i>Neural Network Toolbox (student add-on)</i>	7 €	48 months	6 months	0.87 €
<b>TOTAL<sup>1</sup></b>				<b>89.5 €</b>

TABLE A.4: COST OF THE COMPUTING TOOLS

<sup>1</sup> ARCP Toolbox and Waijung Toolbox are under Creative Commons Licenses.

A.3.3. LABOR COST

<b>Description</b>	<b>Labor hours</b>	<b>Labor cost/ hour</b>	<b>Cost</b>
<i>Tutor, Engineer</i>	50	45 €	2 250 €
<i>Director, PhD Engineer</i>	3	50 €	150 €
<i>Engineering Student</i>	320	20 €	6 400 €
<b>TOTAL</b>			8 800 €

TABLE A.5: LABOR COST

A.3.4. TOTAL BUDGET

<b>Description</b>	<b>Cost</b>
<i>Materials</i>	144.3 €
<i>Informatics</i>	89.5 €
<i>Labor cost</i>	8 800 €
<b>SUBTOTAL</b>	9 033.8 €
<b>Taxes (21% IVA)</b>	1 897.1 €
<b>TOTAL</b>	10 930.9 €

TABLE A.6: UNIFIED TOTAL BUDGET