POLITECNICO DI MILANO
Master's Degree in Computer Science Engineering
Department of Electronics and Computer Science Engineering

INTEGRATING AN EMG SIGNAL CLASSIFIER AND A HAND REHABILITATION DEVICE: EARLY SIGNAL RECOGNITION AND REAL-TIME PERFORMANCES

AI & R Lab
Artificial Intelligence
and Robotics Laboratory

Supervisor: Prof. Giuseppina GINI
Assistant supervisor: Eng. Dario CATTANEO

Master's thesis of:
Luca CAVAZZANA registration number 734498

Academic Year 2010-2011

To Meri. . .

Abstract

The goal of rehabilitative robotics is to take advantage of the developments in modern robotics in order to assist people affected by disabilities using physical training; this way the rehabilitative exercises could be performed autonomously by the patients, without the active involvement of the therapist, making high-intensity rehabilitative therapy an affordable reality for the masses.
Moreover high-precision sensors integrated in rehabilitation devices would allow a quantitative evaluation of the progresses obtained, effectively comparing different training strategies. That would represent a huge scientific achievement in a field where evaluations up to this day are performed only by means of subjective observations.
Important results were obtained in rehabilitative robotics proving his effectiveness; by the way results obtained in the field of the hand rehabilitation are however poorer, especially if compared to the ones achieved with other part of the body. This fact is due to the high complexity of the organ we are dealing with, that makes extremely complex the design of a device suited for its rehabilitation.

In Politecnico di Milano an attempt to design a rehabilitative exoskeleton for the hand using inexpensive materials was made; however the result is a device that was unable to detect the forces generated by the patient making impossible an interactive behavior, which is crucial to obtain maximum benefits from the training.

The main goal of this project is so to provide this device with the capability of an interactive behavior by the mean of the detection of the muscular activity, integrating it with a controller developed during a previous work about prosthesis control, which was able to detect and identify up to seven different movements analyzing the signal recorded using an external electromyograph. To achieve this goal, however, a substantial paradigm shift

was needed: the previous controller wasn't developed for real-time applications, and was basing its classification analyzing signals generated only by a complete execution of each movement. This wasn't suitable for an interactive application, since this way the control variable is generated necessarily after the completion of the very same movement the device is supposed to assist.

The first step was then to achieve a reliable early recognition of the movement: the classifier has been deeply tested to analyze how much classification accuracy varies considering only the initial subsegments the bursts. Experimental data suggest that is possible to obtain classification performances similar to the one obtained with full bursts anticipating the analysis on the initial phase of the movement where signal activity starts to build-up. This is a significant result since it means that we are able to obtain a reliable recognition even before the movement is complete.
The second step was then to re-implement the whole system minimizing the time required for each step of the signal analysis. This was obtained through a complete rewriting of the original code, modifying his architecture to make it suitable for an interactive application and optimizing the algorithm considering the underlying architecture in order to maximize the overall performances of the analysis, maintaining at the same time the original recognition rate.

Experimental results confirmed that the new system is able to recognize the performed movement with just a limited delay: the first control variable is generated only after 200ms, mainly due to the time needed to gather a minimum number of samples needed for a reliable classification, then producing further variables every 25ms, a delay given exclusively by the analysis time. The response time is below the commonly accepted delay of 300ms for interactive applications, while at the same time the classification robustness of the original algorithm has been maintained, since the logical subsiding architecture was not changed in its core.

These results demonstrate that it is indeed possible to effectively use an EMG classifier to obtain a responsive and reliable controller through early analysis of the signal and that is possible to integrate it with a low-cost rehabilitative device, giving to the device itself the capability to assist the patient only after having detected his effort thus promoting his engagement, a fundamental feature to obtain an effective rehabilitation.
To obtain a complete functional rehabilitative framework, however, more

work is needed, mainly concerning the design and the realization of a functional device able to help the patient with the execution of movements that are performed during the most common daily activities.

Sommario

Scopo principale della robotica riabilitativa è quello di sfruttare i recenti sviluppi nel campo della robotica moderna con il fine di offrire assistenza a quei pazienti affetti da disabilità fisiche che inficiano la corretta esecuzione da parte loro di esercizi fisioterapici; tramite questa assistenza l'esercizio riabilitativo può essere eseguito dal paziente in modo autonomo e senza intervento da parte del fisioterapista, rendendo la terapia riabilitativa intensiva accessibile a tutti, evitando gli alti costi imposti dalle tecniche al momento disponibili. Un altro importante aspetto riguarda invece la possibilità di ottenere una valutazione quantitativa dei progressi ottenuti dal paziente utilizzando i sensori con cui solitamente sono equipaggiati i dispositivi. Ciò permette un oggettivo confronto fra differenti strategie, rappresentando un importante passo in avanti dal punto di vista scientifico in un campo nel quale l'efficacia delle terapie è valutato solitamente tramite osservazioni prevalentemente soggettive.
Nonostante siano stati raggiunti risultati importanti, che dimostrano l'efficacia della riabilitazione robotizzata, la riabilitazione rivolta alla mano rimane ancora un argomento relativamente inesplorato: questo deriva dalla complessità dell'organo in sé, elemento che rende estremamente difficile la progettazione di un dispositivo in grado di sostenere il paziente nell'esecuzione della vasta gamma di movimenti che la mano è in grado di eseguire.

Presso il Politecnico di Milano si è cercato di sviluppare un prototipo di esoscheletro atto alla riabilitazione della mano utilizzando materiali di basso costo, il dispositivo realizzato è tuttavia incapace di rilevare forze dal paziente usando i pochi sensori di cui è equipaggiato; questo rende impossibile l'ottenimento di un comportamento interattivo da parte del dispositivo, fattore invece indispensabile per stimolare l'impegno da parte dell'utente, ottenendo così migliori risultati durante la terapia.

L'obbiettivo principale di questo progetto è quello di fornire al suddetto

dispositivo una maggiore capacità interattiva, sfruttando il segnale elettromiografico come controllo ed integrando il dispositivo di riabilitazione con un classificatore software sviluppato durante un precedente studio riguardante gli arti prostetici. Tale classificatore è in grado di riconoscere fino a sette differenti movimenti della mano analizzando il segnale grezzo acquisito da un elettromiografo, qualità che lo rendeva integrabile nel sistema in via di sviluppo. Per ottenere questo obbiettivo occorreva però un sostanziale cambio di approccio: il precedente controllore non era stato sviluppato per applicazioni interattive, e basava la propria robustezza nel riconoscimento sull'analisi di segnali completi, acquisiti cioè fino alla completa esecuzione del movimento. Questo particolare non è adatto alla realizzazione di un'applicazione interattiva, considerando che la variabile di controllo verrebbe necessariamente generata solo dopo l'effettivo completamento del movimento che il dispositivo dovrebbe invece assistere.

Il primo obbiettivo è stato dunque quello di ottenere un riconoscimento affidabile già nella fase iniziale dell'esecuzione del movimento: il classificatore è stato dunque testato per valutare la capacità di riconoscimento utilizzando solamente i segmenti iniziali dei burst. I risultati sperimentali suggeriscono che già nella fase iniziale del movimento, fase in cui l'attività del segnale comincia ad incrementare, è possibile ottenere una percentuale di riconoscimento prossima a quella ottenuta usando burst completi. Il secondo passo è stato quello di re-implementare il sistema rendendo la sua architettura adatta ad applicazioni interattive, apportando diverse ottimizzazioni anche più a basso livello in modo tale da minimizzare i tempi d'esecuzione dell'analisi, tutto senza però modificare l'algoritmo precedentemente proposto, allo scopo di mantenerne intatta la capacità di riconoscimento.

I risultati sperimentali raccolti confermano come il sistema sviluppato sia in grado di riconoscere il movimento realizzato con un ritardo inferiore al limite comunemente accettato di 300ms: la prima variabile di controllo è disponibile infatti dopo soli 200ms dall'inizio del movimento, intervallo temporale principalmente dovuto al tempo necessario per raccogliere un numero di campioni sufficienti per ottenere un'analisi significativa. Ulteriori variabili di controllo vengono poi generate ogni 25ms fino al completamento del movimento, rate dettato dunque esclusivamente dalle tempistiche d'analisi del software stesso. La capacità di riconoscimento è rimasta invece sostanzialmente invariata come atteso, dal momento che l'algoritmo generale non è stato modificato, ma solo ristrutturato.

I risultati ottenuti sono dunque confortanti, e mostrano come sia possibile usare un classificatore di segnali elettromiografici per un controllo affidabile e dal ritardo limitato tramite anticipazione dell'analisi del segnale elettromiografico, e come sia possibile integrarlo con un dispositivo riabilitativo a basso costo in modo da guidare il movimento del paziente solo dopo aver rilevato il suo sforzo, promuovendone un impegno attivo, utile ad una riabilitazione efficace.
Allo scopo però di ottenere un sistema funzionale e completo rivolto alla riabilitazione della mano sono tuttavia necessari ulteriori sviluppi, principalmente riguardanti la progettazione meccanica di un dispositivo più raffinato, realmente in grado di assistere il paziente nell'esecuzione almeno di quei movimenti che sono alla base delle più comuni attività quotidiane.

Acknowledgments

I wish to express my sincere gratitude to my advisor, Prof. Giuseppina Gini, and my assistant advisor, M.Sc. Eng. Dario Cattaneo, for their guidance and help, whose support had been crucial for the development of this work. Thanks also to M.Sc. Eng. Giuseppe Lisi, whose work on EMG classification is the backbone of the current project, to Ph.D. Paolo Belluco, without his work on the Eracle system this series of projects would not exist, and to B.Sc. Eng. Lucia Prisciantelli and B.Sc. Eng. Giuseppe Ventimiglia for their work with Polimanus.

Thanks to my family, for supporting me along all these years.
Thanks to the friends found within this faculty and to the ones found on the track, for the experiences shared together.
And thanks to my girlfriend, Meri, for her invaluable support in these months.

Ringraziamenti

Desidero esprimere la mia gratitudine alla mia relatrice, la professoressa Giuseppina Gini e al mio correlatore, l'ingegnere Dario Cattaneo, per disponibilità e gentilezza dimostrata nel corso della realizzazione di questo progetto di tesi.
Un ringraziamento va posto anche al Dott. Ing. Giuseppe Lisi, il cui studio sulla classificazione dei segnali EMG è stato il punto di partenza del presente progetto, al Ph.D. Paolo Belluco, senza il sistema Eracle da lui progettato questa serie di progetti non potrebbe essere, e al Dott. Ing. Lucia Prisciantelli e al Dott. Ing. Giuseppe Ventimiglia per il loro lavoro con Polimanus.

Grazie alla mia famiglia, per avermi supportato in tutti questi anni.
Grazie agli amici trovati fra i banchi di questa facoltà e sulle piste di atletica per le esperienze condivise.
E grazie alla mia fidanzata, Meri, per l'insostituibile supporto in questi mesi.

Contents

Chapter 1

Introduction

1.1 Overview

Each year 800'000 people in the U.S. experience stroke attacks, with 50% of the survivors left with some kind of hemiparesis and 26% unable to independently perform daily living activities [1]. In Italy 200'000 cases are recorded every year, leading to a total of 913'000 survivors left with inabilities [2]. These statistics make stroke the leading cause of physical impairment all over the world, but other pathologies such as multiple sclerosis, cerebral palsy, spinal cord injury, Parkinson disease or simple bones trauma and ligament degradation are cause of poor quality of life.
Tests shows how significant improvement in movement ability can be achieved through proper physical training, but this leads to a strong economic pressure on the health care system ($34.3 billion in 2008 for U.S., with an estimated lifetime cost of $140'048 per patient [1]) which, combined with a low ratio between patients and therapists, results in a qualitatively poor assistance.

The idea behind rehabilitative robotics is to take advantage of modern robots to assist patients through physical training: from the point of view of the therapist, exercises are highly repetitive, time consuming and physically demanding, the kind of task robots are suited for; so in these last twenty years a great effort was put in developing devices able to offer a (semi-)autonomous training and in studying better methods to help patients to regain their motor functionalities (*robot-aided rehabilitation*), or providing active assistance with the most common daily activities (*assistive exoskeletons*). This way the patient could autonomously perform the training, while the therapist would assume the role of supervisor, monitoring multiple patients at

the same time; moreover with mass production the costs for rehabilitation would dramatically drop.
Currently, robotic therapy programs are offered by several health care centers all over the world, and some devices are commercially available. The ultimate goal is to make affordable domestic intensive therapy a reality in the near future.

Tests conducted within the Veteran Affairs (VA) hospitals [3] show how patients who received high intensity therapy by the MIT robotic device for thirty-six weeks achieved a significant gain in motor functionality, similar to the control group which received a training of similar intensity from a human therapist. Both groups benefited of a significantly greater gain if compared to a second control group who received the traditional care. Over all the study the robotic therapy cost an average of $17'831 per patient, while usual care costs $19'098 per patient and $19'746 and intensive non-robotic therapy.
This suggest that at the current state robotic therapy does not offer better improvements per se, but with eventual drops in costs due to mass production promises to make high-intensity rehabilitation available on a scale that the traditional therapy can't be afforded by the health-care system, both in terms of costs and therapist-per-patient.

Another advantage offered by robotic therapy is the ability to compare different training approaches: in fact for several disorders is not yet clear which treatments are objectively more effective [4], leaving the choice to subjective considerations of the therapist who will tailor the exercise case by case (making this ad-hoc rehabilitation even more costly). This lack of a common protocol and a quantitative evaluation of the patients make hard to scientifically monitor the progresses of the subject and compare different strategies on a cost/benefit basis [3] [5]. With robotic rehabilitation we would have a common framework for different patients, promoting consistency and reproducibility of the training; united to the ability of quantitatively measure patients' progresses using robot's embedded sensors would allow us to objectively compare results obtained with different strategies.

1.2 Research focus

A key element in rehabilitation is interactivity: the patient must not be simply dragged by the robot letting him do the work, or there won't be

an optimal improvement from the neuromotor point of view [6]. So, to avoid slacking, the rehabilitative system must incentive user's effort providing *assistance-as-needed*, meaning only after detecting a concrete effort and providing no more than the force the patient needs to complete the task.
This approach also makes the training exercise naturally scale with the condition of the patient, as the robot will gradually step aside as motion capabilities are regained [7].
Another advantage in assist-on-need is that the patient is able to perform the exercise at his own pace allowing him to better focus on a correct execution, while a 'blind' device would impose an already programmed pace regardless the status of the patient, possibly leading to demotivation and ultimately slacking.

The problem arises when the patient is unable to perform even the slightest movement, making force-triggered assist-on-need useless. For this reason EMG-triggered therapy was introduced: even if muscular tone makes the patient unable to perform any movement his intentions can still be detected if he is able to generate a minimum amount of nervous activity.
Even with users with better muscular conditions EMG analysis can be exploited to further anticipate their movements, or check if they are performing correctly the movement monitoring which muscles are being activated.

Although this field is rapidly developing hand rehabilitation is relatively less dealt with if compared with the arm or the leg. This is because we are working on a very elaborate organ, composed by twenty-two degrees of freedom and actuated by nineteen muscles, resulting in a structure which is very hard to replicate. Moreover muscles are tightly packed in the forearm, making signal analysis pretty complex to perform using non-invasive techniques like surface EMG because of the high interference between muscles (*cross-talk*). Such complexity makes the realization of a rehabilitative device endowed with a comparable dexterity extremely challenging, since it requires an extremely sophisticated mechanical design and a very advanced control strategy.

1.3 Project objectives

The main goal of this project is to develop and test a hand rehabilitation system able to detect the patient's volition by analysing EMG activity, and to assist him through the entire training, in order to regain the capability to

perform at least the most common movements recurring in daily activities. The rehabilitative device employed was developed during previous projects within Politecnico di Milano [8], but its lack of compliance makes impossible the implementation of an *assist-as-needed* strategy using the equipped sensors. So we decided to opt for EMG-based control, exploiting the know-how developed in previous works concerning EMG prosthesis control [9].
Since the classification system already developed was unable to meet the time performance requirements for a smooth interaction with the patient (even in literature, while reports are never short of statistics on recognition rates, the topic of real-time suitability is seldom covered), a key element for the realization of this EMG-triggered rehabilitative device was to redesign the software for signal analysis in order to make it suitable for real-time applications while maintaining its recognition rate.

This control paradigm could be applied beyond hand rehabilitation, like assistive exoskeletons, tele-operation or more generally to haptic interaction.

1.4 Structure of the work

Chapter 2 presents an overview of the hand from an anatomical point of view, exploring detail how muscles work and how they are stimulated. An explanation on how the EMG signal is generated and detected is then offered. Finally, the last section describes the most common hand movements in daily tasks.
Chapter 3 gives an overview of the works done in the field of technology aided rehabilitation reported in literature. The first section explains the most important requirements for an effective rehabilitation and the proposed high-level strategies to achieve it. Then a summary of the algorithms proposed for EMG analysis is presented, and finally an overview of the aspects that should guide the design of a rehabilitative or assistive device is exposed, reviewing the most recent prototypes.
Chapter 4 discusses the design of the implemented system, highlighting how recognition delay was reduced through a more thoughtful implementation and how it interacts with the devices developed within Eracle and Polimanus projects.
Chapter 5 evaluates the performances of the system, both in terms of reaction delay and classification accuracy, discussing also the issues met.
Chapter 6 summarizes some final considerations, reviewing the results achieved and the issues met. It also highlights the issues still open and proposes pos-

sible future developments.

Appendix A contains the diagram of the implemented classes and the logic diagram of the flow of execution.
Appendix B contains the project documentation of the realized controller.
Appendix C presents more in-depth explanations of some of the mathematical concepts used within this work.

Chapter 2

Physiology and anatomy of the hand

This chapter presents an overview of the biologic system underlying the functioning of the human hand and electromyographic signal's acquisition, both needed to understand the present work. The first section explores muscle anatomy, while the second one describes them from a physiological point of view, needed to understand the genesis and acquisition of the EMG signal described in section three. Finally the classification of the most common hand movements performed during daily activities is presented.

The sections about anatomy and physiology are taken from Human Biology by Mader [10], Human Biology - Concepts and Current Issues by Johnson [11] and Human Anatomy & Physiology by Marieb and Hoehn [12].

2.1 Anatomy

Muscles are organs that convert chemical energy in mechanical movement; when stimulated they contract moving the bones through the *tendons*. The force they exert is unidirectional, so to perform complex movements more than one muscle is required, that are called *synergic*. Muscles transmitting opposite movements on the same joint are called *antagonists*. In skeletal muscles one extremity (*origin*) is connected to a fixed bone (with respect to the muscle), while the other one (*insertion*) is connected to the actuated bone.
If we consider the hand most of the muscles are located in the forearm, moving fingers by means of a system of tendons running through sheaths

and leverages. This way is possible to transmit a large force to the fingers relocating the bulky muscles in another area (so called *extrinsic muscles*) thus without burdening the dexterity and weight of the hand.

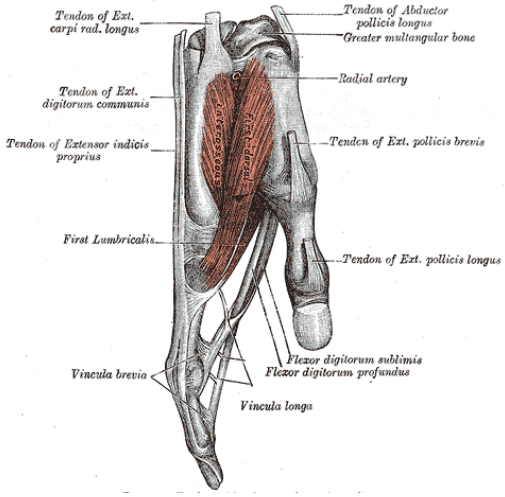


Figure 2.1: Sagittal plane of the hand. The pulley system can be observed [13]

The extrinsic muscles located in the forearm are organized in two compartments, anterior and posterior. The anterior compartment contains muscles responsible for finger and wrist flexion:

- *Flexor digitorum profundus*: responsible for fingers and wrist flexion. It fans out in four tendons, reaching the four fingers.
- *Flexor pollicis longus*: responsible for thumb opposition and wrist flexion.
- *Pronator quadratus*: responsible for hand pronation.
- *Pronator teres*: responsible for hand pronation.
- *Palmaris longus*: responsible for wrist and fingers flexion.
- *Flexor carpi radialis*: responsible for wrist abduction and flexion.
- *Flexor carpi ulnaris*: responsible for wrist adduction and flexion.
- *Flexor digitorum superficialis* (*flexor digitorum sublimis*): responsible for wrist and finger flexion.

The extensors muscles are located in the posterior compartment:

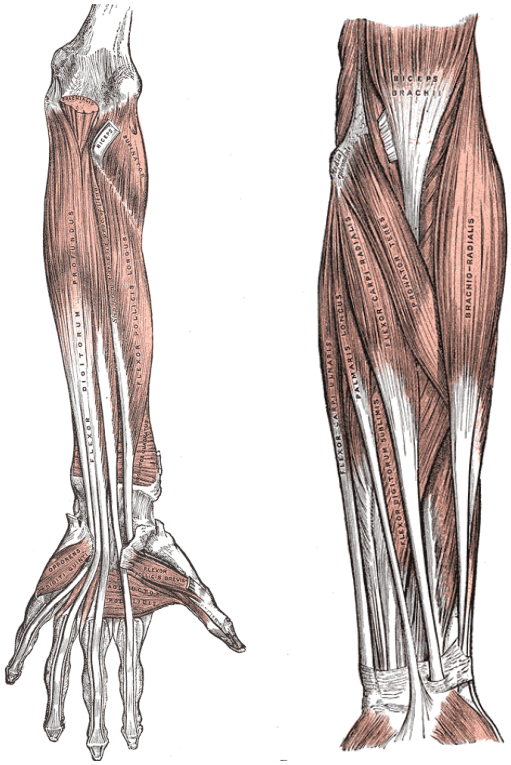


Figure 2.2: Extrinsic flexion muscles, deep and superficial view [13]

- *Extensor carpi radialis longus*: responsible for wrist extension and abduction.
- *Extensor carpi radialis brevis*: responsible for wrist extension and abduction.
- *Extensor carpi ulnaris*: responsible for wrist extension and adduction.
- *Extensor digitorum communis*: responsible for wrist extension, main muscle for finger extension.
- *Extensor indicis profundus*: responsible for wrist and index extension.
- *Extensor digiti minimi*: responsible for wrist and little finger extension.
- *Abductor pollicis longus*: responsible for thumb abduction and extension.
- *Extensor pollicis brevis*: responsible for thumb extension.

- *Extensor pollicis longus*: responsible for thumb extension.
- *Brachioradialis*: situated in the lateral compartment, is responsible for elbow flexion and both hand pronation and supination.

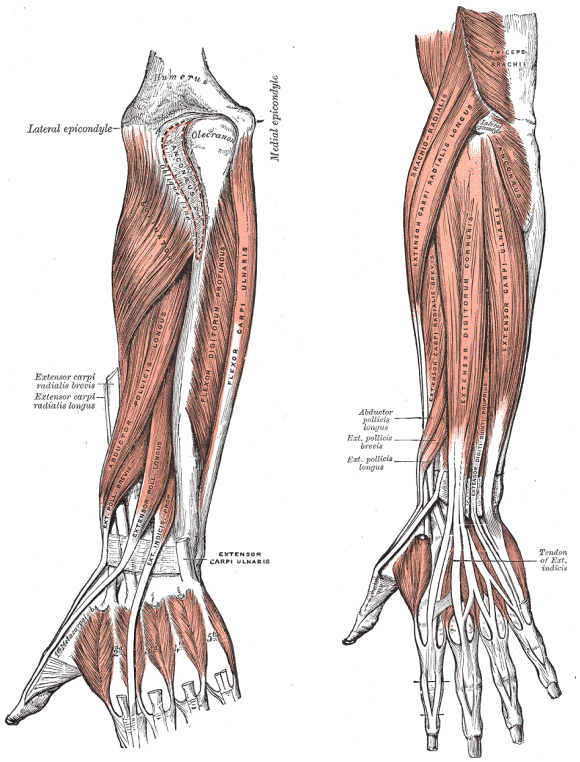


Figure 2.3: Extrinsic extension muscles, deep and superficial view [13]

In the hand there are several *intrinsic muscles* (being in proximity of the very same joints they act on). They are divided in:

- *Thenar eminence*: composed by the *abductor pollicis brevis*, *flexor pollicis brevis* and *opponent pollicis*. They originate from the *flexor retinaculum and* insert at the proximal phalanx of the thumb.
- *Hypotenar eminence*: composed by the *opponens digiti minimi*, *flexor digiti minimi* and *abductor digiti minimi*, controls the motion of the little finger.
- *Lumbrical*: contributes to the flexion of the metacarpophalangeal joints and to the extension of the interphalangeal joints.

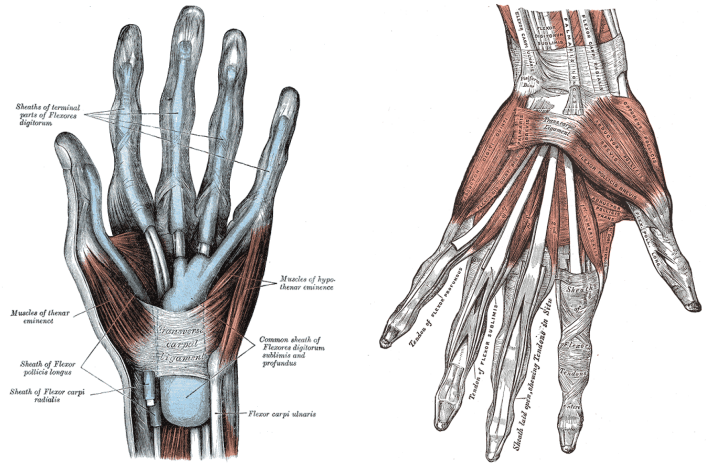


Figure 2.4: Intrinsic muscles of the hand [13]

- *Interossei*: group composed by three volar and four dorsal muscles; contribute to fingers adduction and abduction.

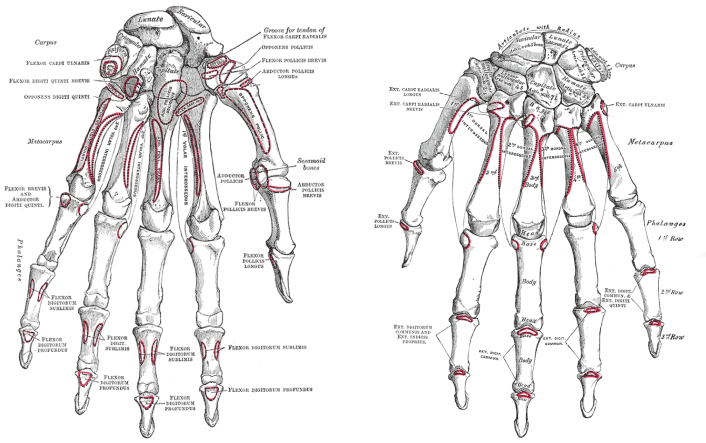


Figure 2.5: Tendons' insertion, volar and dorsal view [13]

The muscles we are mostly interested in this work are the *flexor digitorum profundus* and *flexor digitorum superficialis* which, thanks to their size, exert most of the force needed for wrist and metacarpophalangeal and interphalangeal flexion. They both originate in the proximal part of the forearm (as the name suggests, the profundus muscles is located deeper) and fan out in four tendons that run through the carpal tunnel and attach to the

phalanxes (in the palmar base of the distal phalanxes for the first muscle, in the palmar side of the intermediate phalanxes).

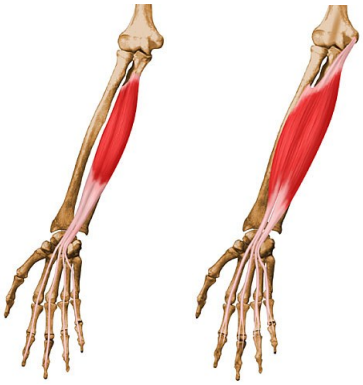


Figure 2.6: Flexor digitorum profundus (left) and superficialis (right), volar view

Another relevant muscle is the *extensor digitorum communis*, which is located in the superficial part of the dorsal side of the forearm, originating near the elbow and fanning out in four tendons which insert in the middle and distal phalanxes, contributing to wrist and finger extension. The *extensor indicis profundus* also contributes to index and flexion, but because of his length and depth his activity is too hard to detect for the scope of the present work.

Figure 2.7: Extensor digitorum communis, dorsal view

For thumb abduction and flexion the muscles we are most interested in are the *flexor pollicis longus* and *abductor pollicis longus and brevis*. The first one is deep in the dorsal side of the forearm, originating in the proximal part of the forearm and inserting in the base of the base of the metacarpal

bone of the thumb. The latter originates around 3/4 of the volar forearm and inserts in the base of the distal phalanx of the thumb.

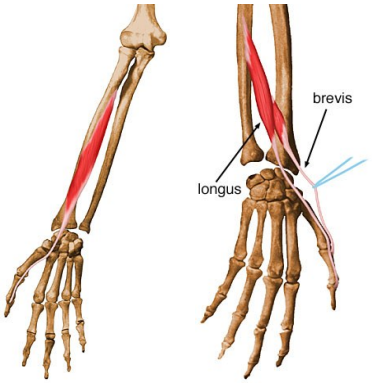


Figure 2.8: Flexor pollicis longus (left) and abductor pollicis longus and brevis (right), volar and dorsal view

2.2 Physiology

2.2.1 Neurons

Neurons are electrically excitable cells which transmit and process information through chemical signaling. A neuron is composed by a body, called *soma*, which projects the *axon*, an extension that travels and branches through other cells (traveling even up to 1m in humans) and propagates the electrical signal along his membrane. It is connected to the *dendrites*, structures which arise from the soma and connect to other axons to gather incoming signals; the connection between the axon and the dendrite is called *synapse*.

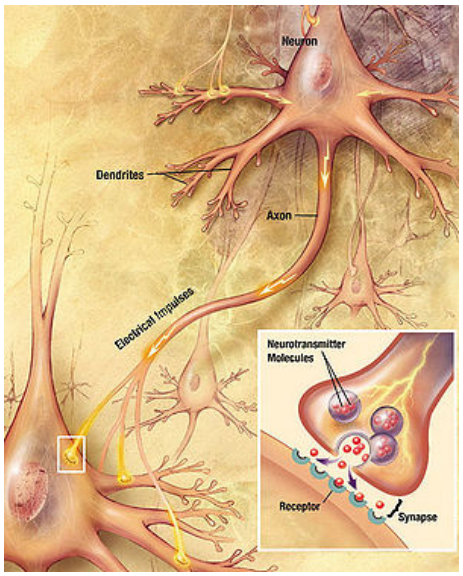


Figure 2.9: Neuron schema

When the axon is not conducting an impulse a potential difference of -70mV exists across the membrane (*resting potential*). This potential is maintained by the sodium-potassium pump, a protein which transport Na^+ out and K^+ inside the axon. When the neuron is stimulated an action potential propagates through the axon, opening sodium gates (allowing Na^+ in, shifting the potential difference to +40mV) and then opening potassium gates (allowing K^+ out, repolarizing the membrane to -70mV). This action potential propagates along the length of the axon like a domino effect at the speed of 700 km/h. Short after the concentrations of Na^+ and K^+ are restored by

the sodium-potassium pump activity.
Axon terminations store vesicles filled with neurotransmitters (*acetylcholine*). When the impulse traveling down the axon branches reaches the axon bulb calcium gates opens, allowing Ca^{2+} in. This causes the vesicles to merge with the presynaptic membrane (*exocytosis*), diffusing the stored neurotransmitters within the *synaptic cleft*. Then the acetylcholine binds with specific receptor proteins on the postsynaptic membrane (on the receptor's dendrites or, in our case, muscle's *motor end plate*), stimulating his sodium-potassium gates thus modifying his membrane potential and subsequently stimulating/inhibiting an action potential in the receptor cell. Short after to avoid continuous stimulation the *acetylcholinesterase* breaks down the neurotransmitter, which will be reabsorbed within the axon bulb.

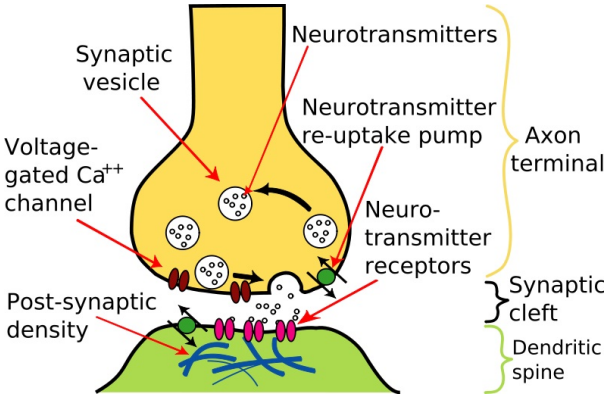


Figure 2.10: Synapse schema

2.2.2 Motor Units

Skeletal muscles are composed by several elementary units called *motor units* (MU). Each MU is composed by a α-*motor neuron* and by all the muscle fibres his axon innervates. The body of the neuron is located in the central nervous system, and when activated propagates the action potential along the axon running through the whole body. As the impulse reaches the axon's terminals directly stimulates the innervated fibres causing the contraction. α-motor neurons involved in the contraction of the same muscle are grouped in *motor neuron pools*, and their number depends on the level of finesse needed to control the force the muscle applies.
When the neuron is stimulated the signal propagates along the axon to the axon bulbs, releasing the acetylcholine stored in the synaptic vesicles. The

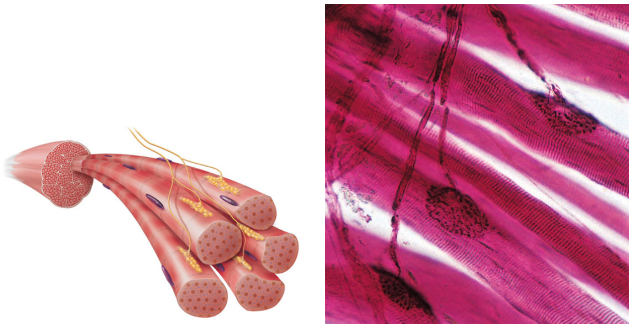


Figure 2.11: Muscle innervation

neurotransmitter diffuses in the neuromuscular junction and binds to the receptors sites on the *motor end plate*, opening the sodium-potassium channels on the membrane. As Na^+ flows in and K^+ flows out the plate depolarizes (*end-plate potential*, EEP), starting a domino effect which propagates the electrical impulse into the inner part of the muscle fibre through the *T-tubules*. The neurotransmitter is then split and reabsorbed by the axon, while the propagation of the impulse runs through the T-tubules, triggering the release of calcium ions from the sarcoplasmic reticulum into the muscle cell cytoplasm. This causes the filaments within sarcomere to slide paste each other, resulting in myofibril and finally muscle fibre contraction.
As nerve activity ends calcium is pumped back into the sarcoplasmic reticulum.

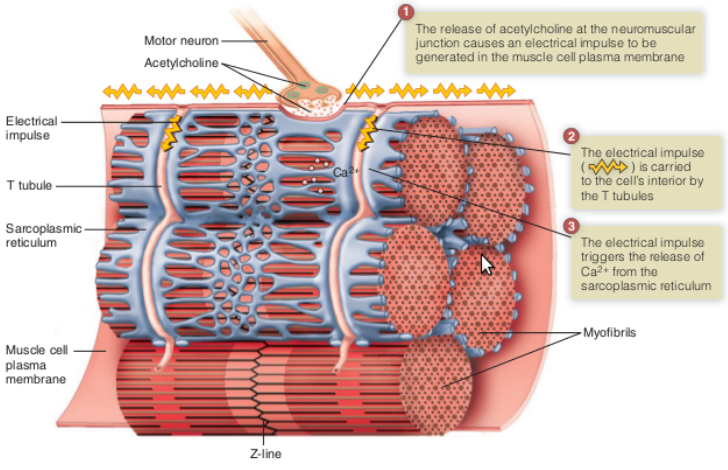


Figure 2.12: Muscle activation

2.2.3 Muscle contraction

At first the stimulus may be too weak to cause the contraction of the fibre, but as soon it reaches the threshold value the fibre contracts and then relaxes. This contraction, called *muscle twitch*, lasts only a fraction of seconds and is maximal (all-or-none). That means that a stronger stimulus won't cause a single stronger contraction, but will cause the fibre to contract more frequently: additional stimuli will arrive before relaxation is complete, bringing more calcium to what is left from previous contractions and leading to a greater force (*summation*). When the fibre reaches the maximum possible contraction (*tetanus*) the muscle eventually depletes his energy reserves (*fatigue*), relaxing even if still stimulated.
The global strength depends on the total number of activated fibres (*recruitment*) in any instant.

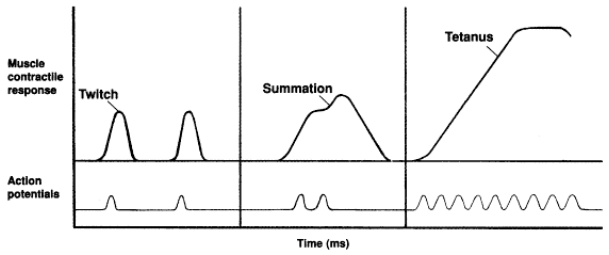


Figure 2.13: Twitch, summation and tetanus patterns

2.3 The electromyogram

2.3.1 sEMG

Changes in muscle membrane potential during contractions vary from -90mV to 50µV or 30mV (depending on the muscle), causing fluctuations in the electro-magnetic field around the muscle. These changes are measured by the *electromyograph*, which generates a time-mapping of these potentials called *electromyogram*. These fluctuations are detected placing electrodes over the skin surface or by means of more invasive techniques like intramuscular needle (more precise but less hygienic and comfortable).
The action potential generated by a single motor unit is called *motor unit action potential* (MUAP), and is composed by the summation of the same impulse travelling through different axon branches to the innervated fibres

and recorded multiple times with different attenuations and delays (depending on the axon's ramification length and distance of the fibre from the electrode).

Figure 2.14: Surface and needle electrodes

A sequence of MUAPs by the same motor unit generates a stochastic signal called *MUAPTrain*.
The measured signal is the space-time summation of all the muscle fibre action potentials nearby occurring at random intervals (plus some random noise). The contribution of each signal will fall as the source-electrode distance increases (inversely proportional), so closer MUs will dominate the EMG. Other factors affecting the strength of the signal are the diameter of the stimulated fibre (which holds a proportional relation) and the intrinsic filtering characteristics of the electrode.

Many sources in literature establish that the amplitude of the EMG signal has a stochastic nature with Gaussian distribution. It ranges from 0 to 10mV (peak-to-peak) or 0 to 1.5mV (RMS). The electrodes are placed across the motor end plate in order to detect the electrical potential. The signal is then usually amplified, filtered to remove frequencies out of the range we are interested in and then digitally converted [14].

Signal frequency falls within 10-200Hz: 10-30Hz is the burst frequency of the single motor unit, over 30Hz is the frequency of the superposition of multiple MUs.

Several factors affect the measured signal:

- Noise caused by the movement of the electrode. This usually causes a signal below 10Hz, which is out of the EMG range and can be filtered out.

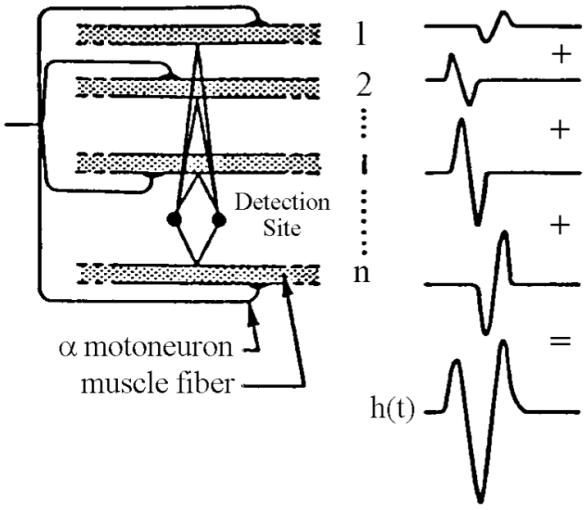


Figure 2.15: Summation of action potentials

- When several muscles are packed in a small area (like in the forearm) reading the signal of a specific muscle becomes hard because of the activity of nearby muscles. This phenomenon is called *cross-talk*. This interference cannot be filtered out, although is possible to improve the signal using mathematical techniques like *independent component analysis*.

- Muscle fatigue causes signals with lower frequency and higher amplitude [15].

- External interference, caused by surroundings devices and, mainly, the power supply of the electromyograph itself. Falling within the range of the EMG (50-60Hz), this noise cannot be filtered since falls within the natural frequencies of the signal.

2.4 Hand movements

The upper limb is divided in three regions: the *arm*, from the shoulder to the elbow, the *forearm*, from the elbow to the wrist and the *hand*, composed by *wrist*, *palm*, four *fingers* and the *thumb*. The hand is one of the most complex organs we are provided, counting twenty-two degrees of freedom actuated by nineteen muscles and able to perform very dexterous movements. As we can observe in the *cortical homunculus* (figure 2.16), which is a pictorial representation of the functionality division of the cerebral cortex, the hand

covers most of both the motor and somatosensory cortex. This gives an idea of the high complexity of the control needed to perform movements with such a fine organ.

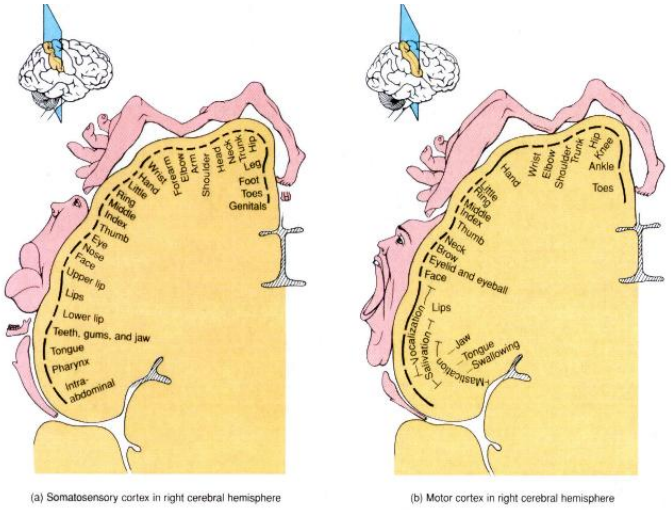


Figure 2.16: Cortical homunculus

It is important to define the taxonomy of the most important gestures accordingly to their function. The most widely referred to is the one proposed in 1989 by Cutkosky after observing the most common grasp types recurring in manufacturing tasks [16]: he highlighted sixteen different postures divided in power and precision grasp from left to right, and by shape and function down the tree.
This taxonomy provides an object-centric classification of the hand, and does not completely describe the full range of possible gestures. More recently a new taxonomy was proposed considering a more complete motion-centric prospective [17] (see 2.18).

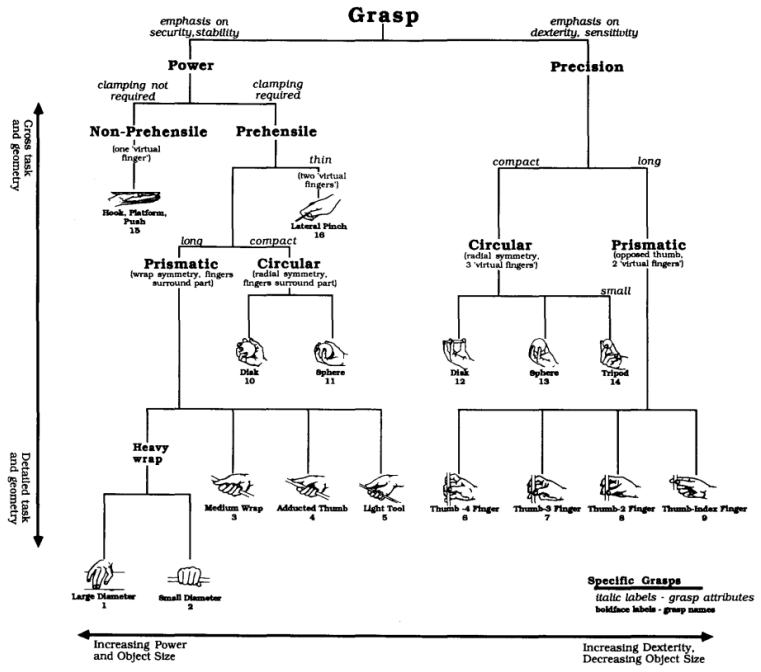


Figure 2.17: Cutkosky's taxonomy [16]

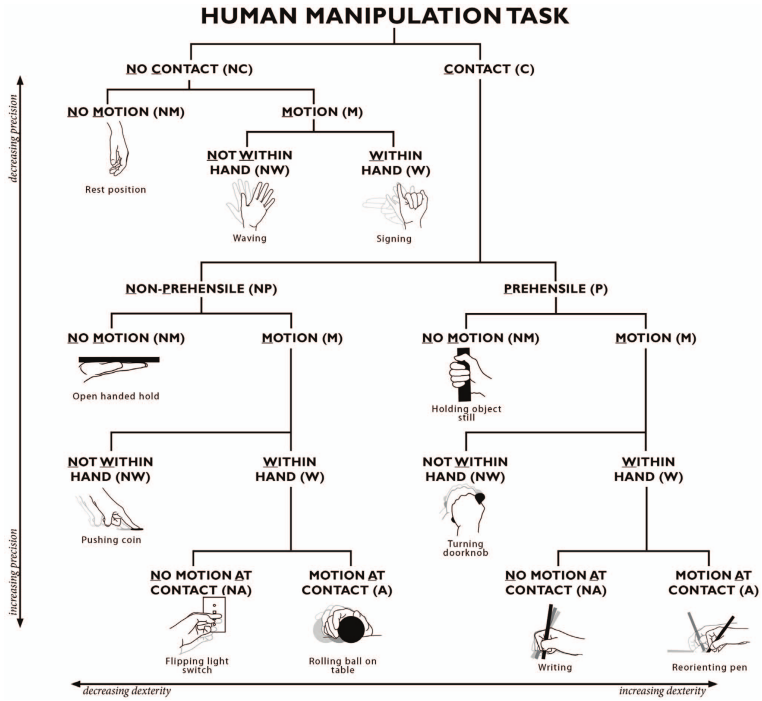


Figure 2.18: Functional taxonomy [17]

These movements can be obtained combining the following basic movements.

Hand pronation and supination The neutral position is with the elbow flexed ad 90°, with the axis of the palm perpendicular to the floor. From this position, *pronation* is the rotation of the forearm that will make the palm face down (axis perpendicular to the floor). *Supination* is the rotation in the opposite direction, making the palm face up.

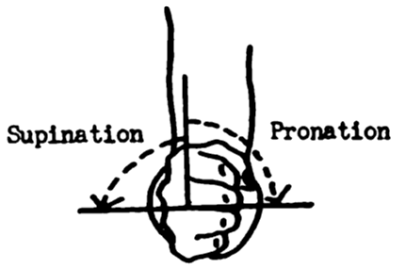


Figure 2.19: Hand pronation and supination

Wrist extension and flexion Starting from the neutral position (elbow flexed at 90°), *wrist extension* (or *dorsiflexion*) is the upward rotation of the wrist, while *wrist flexion* (or *palmar flexion*) is the downward rotation.

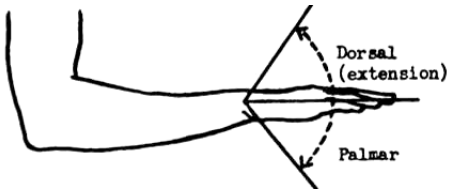


Figure 2.20: Wrist extension and flexion

Finger hyperextension and flexion The neutral position is the open hand with all fingers extended lying on the same plane parallel to the floor. *Finger flexion* consists in the bending of the fingertips toward the palm, while *finger hyperextension* is a unusual movement where finger raises above the palmar plane.
The flexion of all fingers will result in what is informally called *hand closing*, while the motion from the close state to the neutral position is called *hand opening*.

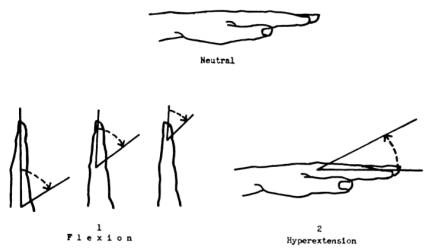


Figure 2.21: Finger hyperextension and flexion

Thumb abduction and opposition The neutral position is with the thumb extended alongside the hand, parallel to the other fingers. *Abduction* is the movement that pulls the thumb away from the palmar midline, while the *opposition* is the flexion on the palm.

Figure 2.22: Thumb adbuction and opposition

Chapter 3

State of the art

This chapter gives an overview of the most recent achievement in the field of rehabilitative robotics and exoskeletons, drawing also from fields sharing some subjects such as prosthesis control.

The first section describes the most promising high-level strategies for rehabilitative exercises, explaining the principles they are based on. In particular the importance to promote user participation to obtain real improvements is highlighted, and several diametrical strategies to achieve this goal are considered.

The second section explores several control algorithms, classified into continuous control, used to control a limited number of actuators analysing the muscular activity, and classifiers, able to identify more complex gestures using more sophisticated features. A description of the techniques proposed in literature is provided, finally reviewing the performances of the discussed systems.

The third part lists the most important features that should be taken into account during the design of a rehabilitative or assistive device. Finally a review of the most recent prototypes is offered.

3.1 Training

Is well known that is possible to improve motor conditions by repetitive training: performing motions helps to improve joint plasticity, continuous contraction and extension (even if passive) help to regain muscle tone. If the patient is affected by a neurological disorder guiding him through the

exercise helps him 'by example' to associate alternative pathways to help the damaged ones in performing the movement, unmasking redundant motor network and re-organizing around the lesion site [18].

However recent tests show how simply forcing the patient through a set of movement results in suboptimal improvements from the neuro-muscular point of view: better results are achieved when there is an active effort from the user [7]. In fact human control model is able to adapt to new dynamic environments optimizing his own effort, meaning that soon it will naturally learn to exploit the therapist/robot letting him generate most of the force, while the patient will simply be dragged through the exercise with the minimum participation, leading to poor improvements (aka: *guidance hypothesis*). So a key element in rehabilitation is the so called *assistance as needed* (or *faded guidance*), meaning the robot must trigger help only after detecting a concrete effort to complete the given task, thus forcing user engagement and promoting motor learning while retaining the voluntary control of the limb; moreover it should provide only the minimum force the patient needs to complete the task, promoting the maximum strain from the user [7, 19, 20].
Another advantage in user-triggered assistance is that the pace of the exercise is determined by the patient itself: a 'blind' device can only execute an exercise with a pre-defined speed, forcing the patient to the same pace regardless his condition (fatigue, for example), putting him in a state of discomfort since he would feel he's not in control of the situation, leading to frustration and potentially giving up to robot guidance. With assistance-as-needed is the patient to determine the pace of the exercise, so the training will naturally adapt to his conditions.

Assistive control algorithms are based on highly backdrivable devices, able to quickly react to the force applied by the patient. Issues arise if the condition of the patient does not allow him to execute even the slightest movement, making the force-control paradigm useless. More recently new approaches based on detecting user's effort through surface EMG analysis were explored: if the patient is still able to generate some nervous activity his intentions could be detected, and some kinematic parameters could even be estimated [19](see subsection 3.2).

An alternative paradigm, termed *impedance based*, consists in let the patient independently move through the goal, and guide him applying mechanical impedances when he deviates from the desired trajectory [4, 21]. This kind

of strategy however can only be applied in a more advanced stage of the therapy, since it requires the patient to be able to perform autonomously a minimum amount of movement.

Other opposite approaches, termed *challenge-based strategies*, are meant to achieve rehabilitation through making the task even more difficult: some examples are providing an increasing resistance to the movement of the limb [7], applying disturbances along directions diagonal to the desired trajectory or amplify the size of movement error, thus encouraging a higher effort [4]. These strategies are meant to prevent patients from slacking and forcing an even greater involvement making the rehabilitative task harder.

Another commonly implemented strategy involves adding a sensory feedback with images and sounds, stimulating the user with a simulated reality which provides him more high-level and entertaining goals beside the mere completion of the movement, thus engaging his interest and motivation. These virtual tasks usually consist in simple games or simulation of common daily activities [4, 22, 23].

Several studies showed how these different strategies for robot-aided therapy can lead to a significant improvement in motor skills [3, 24], however is yet unclear which ones have the potential to produce greater benefits mostly because tests are designed on ad hoc basis and performed over different framework. A more rigorous study should be started, involving randomized trials on a common framework with the goal to effectively compare different strategies and identify the ones best suited for every type of injuries and stage of recover, but that would be a very expensive and time-consuming work [4, 21].

3.2 EMG analysis

Historically rehabilitation engineering has been the first field of application of myoelectric control, initially used for arm prostheses by Wiener's Cybernetics in the late 1940s and more recently applied for tele-operation and virtual reality. The field that so far promoted this research the most is active prostheses, with the goal to use the remaining part of the muscles to control an active artificial limb to perform the most common daily activities [19]. However there are some differences between rehabilitation and prosthesis control: in the first one we must restore the very same pre-accident pattern,

so we are interested in a control as natural as possible. In the latter instead the level of amputation could force to map the control over other areas, so what is really important is the repeatability of the signal [25].
Controlling the device using the EMG signal allows us to implement the aforementioned assistance as needed when user's conditions are not suitable for a force control. Plus it offers the most natural control for assistive exoskeletons or prostheses (mapping each actuator on the very same muscle it is meant to assist).

Different approaches to signal analysis were proposed, each one with different features and applications. The simplest technique is based on signal thresholding [19, 26]: the controller checks when the signal (or his linear envelope, to cut most of the noise) exceeds a fixed value. An extension of this technique is to use a hysteresis instead of the single-level threshold, making it more robust against oscillations due to noise [27]. This control paradigm is very easy to implement, but has limited capabilities since the output is only a binary value. However it can be used to trigger help when a minimum muscle activity is detected.
A more sophisticated approach is to output a continuous value roughly proportional to muscle activity (computed as the linear envelope of the rectified signal, to dim noise contribution). Being the signal a continuous variable we have a finer control if compared with simple thresholding [27].
These algorithms are already commercially employed for prostheses control (Otto Bock, Touch Bionics), but they permits to control only few hand postures with a coarse control if compared to the level of dexterity required by most daily activities. This is due to the limited number of independent signals we can register from the forearm: in this area of the body there is a high number of muscles packed in a relatively small volume, making impossible to register the activity of a single muscle by means of non-invasive techniques like sEMG; what we have instead is the summation of the activity of all the nearby muscles (*cross-talk*), making hard to precisely identify the performed grasp type. This requires a classification architecture based on the extraction of more complex features.

Machine learning gives us powerful tools for automatic classification that can be used to train a system able to catch signal patterns and predict the movement of the patient [28].
As preliminary step *independent component analysis* (ICA) over the acquired channels can be applied: it is a mathematical technique to separate two or more signal sources from a set of mixtures using multivariate sta-

tistical data analysis. The idea is to use ICA algorithms for unsupervised cross-talk removal before feature extraction in order to improve recognition rate [29, 30]. As reported in [31] ICA can improve recognition rate up to 97% from the 60% obtained on the raw signal using simple features.
The first stage of the analysis consists in extracting the most significant features from the burst. The most common time-domain features are the mean absolute value (MAV) [32], integral absolute value (IAV), standard deviation (SD) [33], signal power (SP), logRMS [25, 33, 34] and waveform length. Frequency-domain features are zero crossing, power spectrum, spectrum centroid [33] and frequency ratio [34].

More recently time/frequency-domain features were introduced: being the EMG a non-stationary signal, features based on the classical Fourier transform can't be effectively applied because of the shifting frequencies over time. *Short time Fourier Transform* (STFT) consists in applying the FT on a moving window, allowing the time-localizing of frequency features. The downside of STFT is the resolution of the analysis is fixed by the size of the window, allowing a limited flexibility. An alternative technique widely employed is the novel *wavelet analysis* [32, 9, 35]: developed in the early 1980s, is a time-frequency representation of the signal obtained by means of the convolution with a finite-energy function (*wavelet*) obtained scaling and stretching by several coefficients the same a single basic wavelet (*mother wavelet*) [36]. The main advantage if compared with STFT is we have a variable time-frequency aspect ratio, with high frequency localisation at lower frequencies (long time windows) and high time localisation for higher frequencies (short time windows).

Various implementation can be used, like *continuous wavelet transform* (CWT), the computationally faster *discrete wavelet transform* (DWT) or the *discrete wavelet pack transform* (DWPT).
We can choose the mother wavelet from a large number of families, depending on the application (most notably considering the nature of the signal and computational constrains). The Haar wavelet for example has the advantage of being simple and fast to compute. Wavelets from the Morlet or Daubechies families instead are more complex to compute but can pick more details from the signal [37, 38]. Since choosing a wavelet matching closely the nature of the signal is of utmost importance, a trade-off between resolution and speed must be considered.
The output matrix of wavelet analysis contains a large number of elements ($N \times C$ for CWT for example, where N is the length of the signal and C

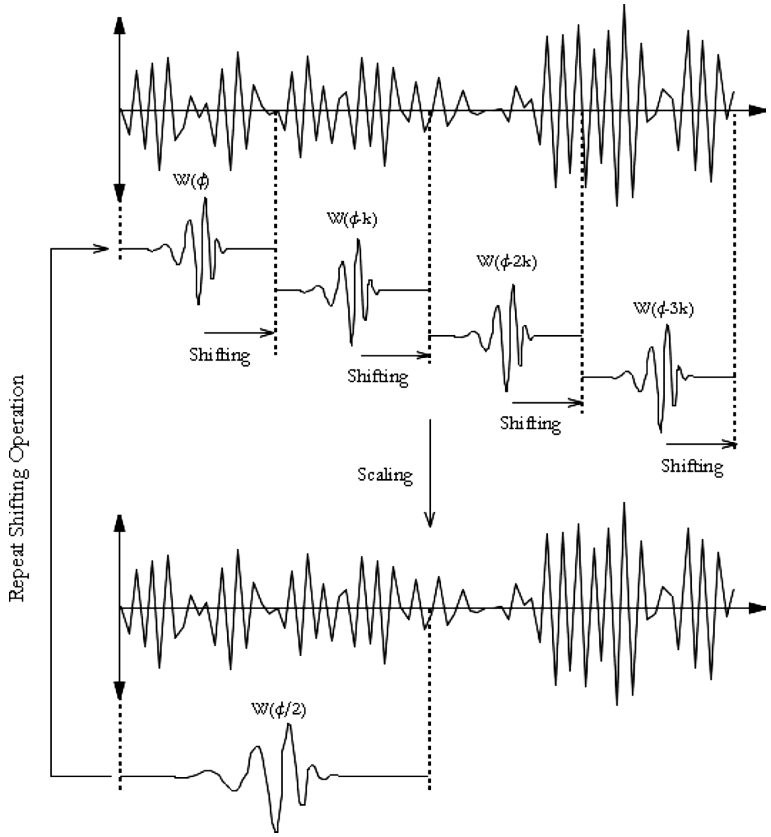


Figure 3.1: Wavelet transform

is the number of wavelet scales), so a technique to reduce the number of elements while preserving as much information as possible must be applied. Dimensionality reduction can be performed applying *principal component analysis*, a techniques that projects the signal in a new space of orthogonal vectors (principal component) thus compressing it in a vector of coefficients associated with the PCs [35]. Another technique is to extract the *singular values vector* obtained performing the *singular values decomposition* of the matrix [32]. *Self-organizing maps* (SOM) uses a lattice of artificial neurons to reduces features' dimensionality and increases their separability mapping the output on a two-dimensional map which preserve inputs' neighbouring [38].

The third and final part consists in inputting the selected features in a classifier to recognize the performed movement. This is achieved using artificial *neural network* (non-linear function approximator) [31, 32, 39], *support vec-*

tor machines (classification through maximization of boundaries between separate classes) [9, 35] or *fuzzy logic* (inference using approximated values, similar to human reasoning) [28].

Discrete signal approach gives us information about the final posture of the device, but does not allow per se a continuous control of the movement. Thus hybrid strategies could be implemented, using classification for gesture identification and the aforementioned proportional control to determine force or speed of execution.

Classification performance depends on several system parameters. The most important factor is the set of gesture under analysis: a simple set of few very different gestures (involving only antagonistic muscles for example) will obtain better recognition performances than a more exhaustive set. The reason is as the number of gestures increases it becomes more difficult to separate their features, especially if some have similar patterns: for example there will be a great difference among the signal generated by hand opening and closing since their movements differ significantly. Contrary, hand opening and wrist extension will present a more similar pattern, making them harder to distinguish.
Another parameter that affects recognition performances is the number of acquired signals: it is intuitive how few channels provide only a limited number of features to work on, while a greater number of input signals allow a better coverage of forearm activity.

Hudgins in 1993 was able classify four discretional[1] movements for prostheses control with a 90% rate analysing time-domain features with ANNs over a single channel [39]. More recent works achieved a 99% rate introducing WT over four channels on a six-class problem [40, 41]. More trials were performed to highlight the relation between number of channel and gestures: with ten gestures a 94% rate was achieved over sixteen channels, dropping to 93% and 87% respectively for eight and four channels [42].
In [35] a 97.5% rate was obtained over six gestures with RMS + DWT + PCA + SVM on two channels only. Previous works within Politecnico di Milano which inspired this project achieved a 96% rate over five gestures with two electrodes [32], and later the same rate was obtained with seven gestures over three channels using CWT+SVD+NN [9].

[1] With 'common' gestures we refer to the most recurrent able-bodied movement in daily activities. With 'discretional' we mean an arbitrarily chosen set of contractions specifically selected to increase class separability.

As highlighted in [15] other physiological factors can affect classification performances, such as the thickness of the skin under the electrode (which attenuates the signal), sweating, muscle fatigue, muscular tone and patient's motivation.

3.3 Hardware

Hand rehabilitation isn't very developed if compared with studies on the arm or the leg: this is because the hand is a very complex organ, with a lot of degrees of freedom independently actuated in a relatively small space, and able to perform a wide set of movements. That makes the design of a complete rehabilitative device or exoskeleton extremely complex from the mechanical point of view.
The first thing we must consider in the design of a mechanical rehabilitative or assistive device is the set of target movements: the device must be allow the selected movements (usually the most recurrent in daily activities) taking in consideration the number and type of the variables that the control strategy can actually provide (potentially leading to an under-actuation of the device). Since the design become more complex as the dexterity level increase, functionality requirements must be taken into consideration carefully.

Then the most important requisite all medical devices must satisfy is safety: this is a key feature for robot that must interact with humans, especially if we are dealing with patients whose disabilities could require extra-constrains because of their peculiar conditions. The robot must move in a way to help the patient executing the correct movement without forcing unnatural or harmful positions, so safety constrains must be implemented on both control and mechanical level [3, 43]:

- Joint's range of motion should not exceed the ones of the hand.
- The controller should limit the controlled variable to safe positions and speeds.
- Should be available a kill-switch which depowers the device or brings it to a safe position.

Another key feature is ergonomics: this is a very important aspect since the patient could be forced to interact with the device for a long period of time, so an uncomfortable design could disincentivize his use. For assistive device

there is the extra requirement of being lightweight as it must be carried over and not be an obstacle in patient's workspace (for example his structure must not occlude the palm or it would make impossible gripping objects); actuators should be located outside the device and transmit forces through artificial tendons, since a self-contained design would result in a bulkier structure whose inertia would lead to clumsy movements [44, 45]. Moreover the device should present a fast setup process and must be adjustable in order to fit different hand sizes.
A high backdrivability is very important for a natural interaction with the device: this allows the ability to quickly respond to external forces, essential for a smooth interaction with the patient, and to provide an apparent null mass, critical to not encumber the already weakened limb with the inertia of the device. This is obtained integrating the device sensors with enough resolution to detect user generated forces.
A feature rehabilitative robot must provide is antigravity support: the device must be designed to support the arm, since user's condition could make his weight an unsustainable burden. Thanks to antigravity support the robot will make the arm virtually weightless, allowing the patient to focus on the correct execution of the movement [4].

The device developed at Carnegie Mellon University [44] is a lightweight prototype to assist the index with pinch movement. It was developed to test different control strategies using impaired patients.

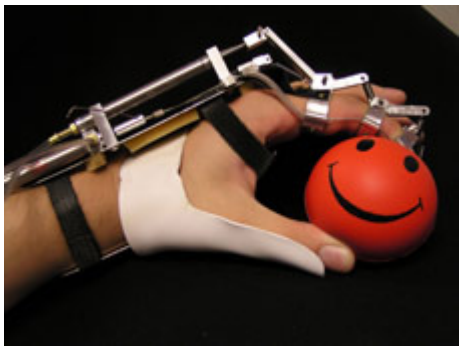


Figure 3.2: The exoskeleton developed by Carnegie Mellon University [44]

The Rutgers Master II-ND [46] consists in a set of low-friction pneumatic actuators located in the palm connects to the fingertips, applying up to 16N of force. Feedback is provided by a series of infrared and Hall-effect encoders

(contactless sensors were specifically chosen to minimize disturbance force due to friction).

Figure 3.3: Rutgers Master II-ND

The overall system is very light and comfortable, preserving user's freedom of motion. Although originally designed for haptic applications, it has been successfully employed for VR rehabilitation [47]. Since his structure occludes the palm, it can't be used to assist grasping movements.

The device developed at the Hong Kong Polytechnic University [24] is a wearable device that can be used bot for training and daily assistance. It was designed to potentially actuate all fingers independently, but the current control system based on two-channel thresholding allows only performing grasping and hand-opening movements. Test performed on stroke subject showed how after a twenty sessions of twenty minutes exercise clinical assessments highlighted significant motor improvements.

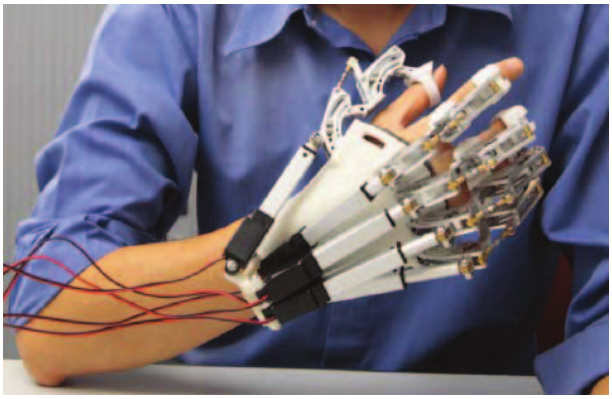


Figure 3.4: The exoskeleton developed atHong Kong Polytechnic University [24]

The prototyped developed by Zheng Li in his master's thesis for North Carolina State University [43] is able to independently actuate each finger using custom-made McKibben actuators. Artificial muscles emulate the biological ones, originating in the proximal forearm and inserting into the phalanxes through a system of Velcro rings replicating hand's sheaths system.

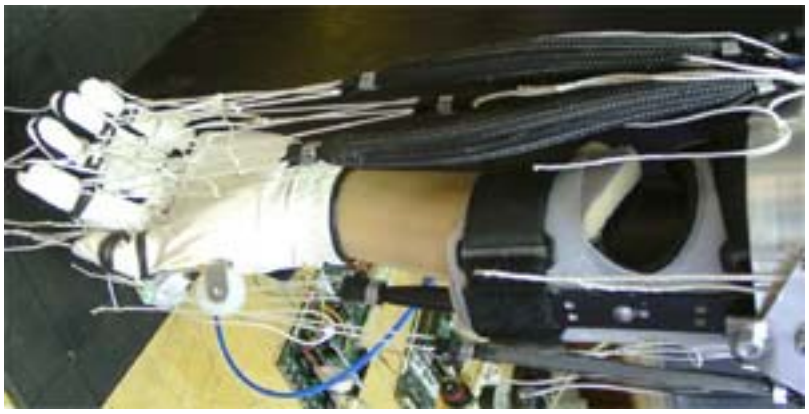


Figure 3.5: The prototype developed by Zheng Li [43]

The overall system is lightweight, low-cost and easily adapts to different hand sizes. McKibben actuators have a high force-weight ratio, and are easy to control thanks to the relation between force and pressure in the pipe. Their natural compliance makes them safer for medical purposes, but the high response time makes them unsuitable for applications which require a fast interaction.

Developed by UIC, J-Glove is a jointless exoskeleton able to assist the patient in grasp/release exercises. It consists in a glove which uses the very same hand as structure, while the force (up to 130N) is transmitted through a system of Bowden cables, allowing to relocate actuators off hand thus making it more comfortable [48].
This device can be controlled through a button, voice activation or surface EMG. In particular EMG control is simply based on thresholding the activity of the extensor digitorum communis and the flexor digitorum superficialis [49].

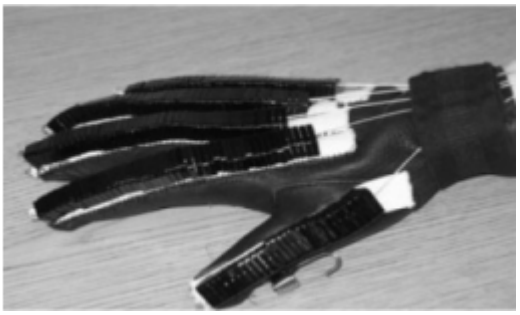


Figure 3.6: J-Glove [48]

His design is very promising, since makes the device slim, comfortable and easy to manufacture. It already inspired new prototypes [50].

Chapter 4

Project

This project is based on the exoskeleton developed within the Artificial Intelligence and Robotics Laboratory (AIRLab) of Politecnico di Milano. It is a low-cost rehabilitative exoskeleton able to help patients performing grasp, pinch and fingers extension [8]; it is an open-loop device, since the high impedance of the actuators makes impossible to feedback the user-generated forces using the two equipped potentiometers. The idea then is to control it using a gesture classifier previously developed for EMG prosthesis control [15]. It is able to identify seven different gestures (grasp, hand extension, wrist flexion and extension, thumb abduction and opposition, index hyperextension) acquiring EMG activity from three channels placed on the forearm.

According to empirical analysis the maximum tolerable delay between the user command and the action of the prosthesis/exoskeleton is around 300ms [51], so the main goal of this project is to make Polimanus usable for an interactive rehabilitation, achieved modifying the architecture of the classifier reducing the delay of the analysis in order to meet this time-performance requirement.
The first issue was to determine the number of samples needed for a successful classification: the previous version of the software acquires big batches of data (way longer than the duration of an average movement) which are later analysed to find complete movements to identify. With this approach an online classifier is unfeasible, since it requires the movement to complete before being able to recognize it (so the control variable for the mechanical device will necessarily be available too late). The first goal was then to determine if is possible to identify a movement before its completion, and if so study how classification rate vary with the length of the analysed seg-

ment. The second step was then to rethink the architecture in a way to allow continuous control, eventually modifying the algorithm to reduce the computational delay without affecting the recognition rate and implement it efficiently on the underlying system (Matlab in our case).

To simplify the architecture three modules where created:

- The first one provides an interface with the Eracle board and outputs the parsed signal.
- The second module encapsulates and analyses the acquired signal. It is designed to work with both continuous online and batch classification.
- The last module provides an interface to control Polimanus

Our system is based on the interaction of these modules.

Finally we have the delay brought by serial communication with the artificial limb and the reactivity of is mechanical components, even if this aspect is out of the scope of this thesis.

The overall system consists in a USB electromyograph connected to a PC (P4-2.4 GHz, 4GB RAM) where runs the classifier written in Matlab 7.11, which controls the serial-connected rehabilitative exoskeleton.

4.1 Recognition anticipation

The previously developed classifier took into consideration only complete bursts, which means we had to wait for the movement to complete before his identification. This is unacceptable for an interactive device, since the control signal would be necessarily late, so we investigated how recognition rate vary using samples of an incomplete movement.

A test was set to observe the performances of the net over features extracted from the initial segments of several recorded movements: five neural network were trained using the bursts gathered during an acquisition performed following the protocol previously suggested [15]; the set was composed by thirty bursts for seven gestures, with an average length of about 200 samples each (as will be explained more in detail in section 4.3.1, 100 samples are the

preamble kept to catch possible early dynamics whose amplitude is too low to allow detection, while the others are the actually detected activity).
Finally the classification performances were computed over the testing subset, feeding the networks with features extracted from the initial segments of the bursts with increasing lengths. Figure 4.1 displays the recognition rate over the percentage of the analyzed burst. As can be observed the classifier

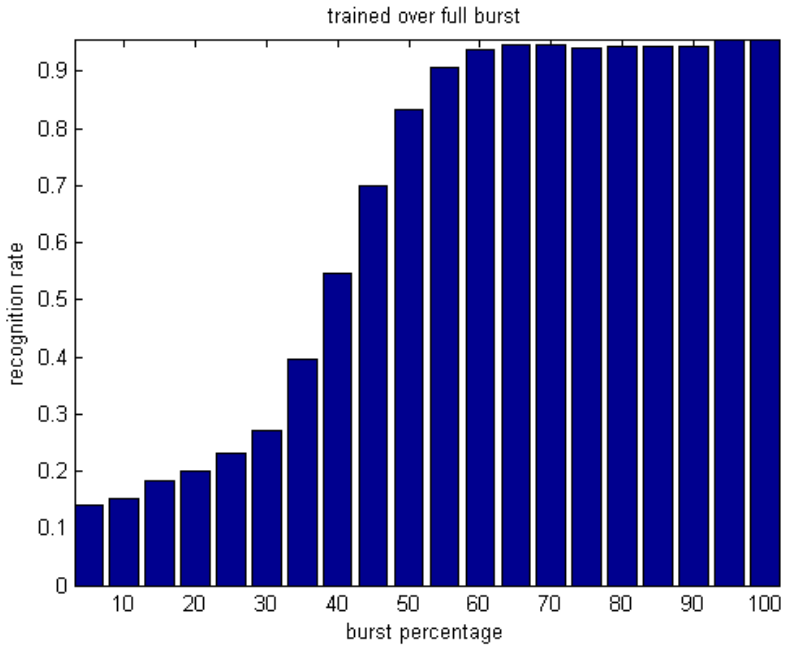


Figure 4.1: Recognition rate over burst

is unable to identify the signal in the preamble; after 50% (on the initial segment of the wavefront) the recognition rate dramatically increases reaching performances close the ones obtained analyzing a complete burst. It means the features characterizing a gesture start to show in the early movement.
The test was repeated using networks trained using only half bursts. Recognition rate as foreseeable reached his maximum with half burst and slightly decreased for longer segments (see 4.2). The decreasing trend near full burst is due to the fact that the network reaches the peak of his performances over the very same burst length used for his training; as the percentage of the analysed segment moves from 50% the extracted features slightly change, thus scoring under 90% at the end of the graph.
Further tests were made training nets over segments even shorter, but no significant result was achieved.

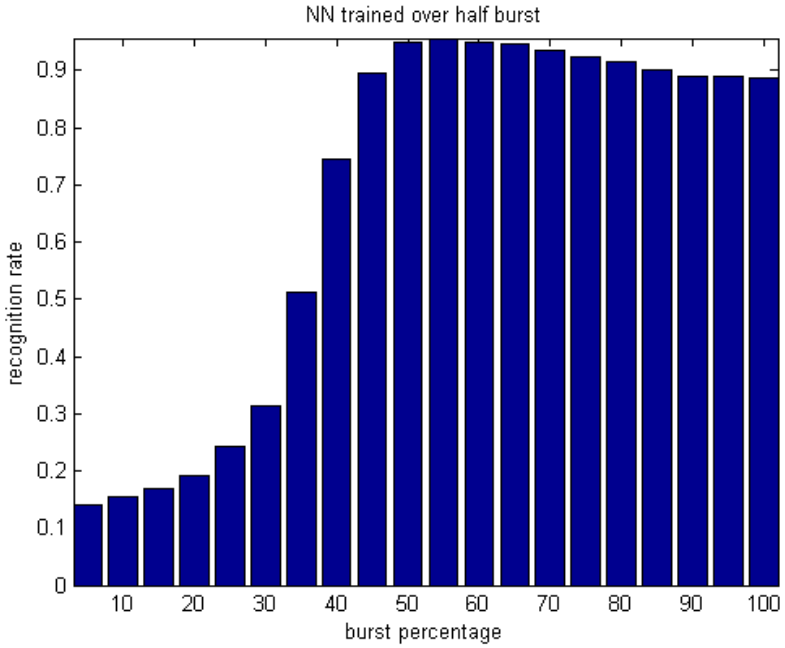


Figure 4.2: Recognition rate over burst percentage

These tests shows how we can already obtain in an early stage a recognition rate close to the one obtained analyzing full bursts, using features extracted from the preamble and few samples right after the detection of the movement.

4.2 Signal acquisition

4.2.1 Eracle

The EMG board is a wearable device developed within the Department of Mechanics of Politecnico di Milano [30], designed with the goal to interact in virtual reality environments; it provides three bipolar channels plus a common reference, which is used to remove the common noise between two paired electrodes. Each electrode records the voltage with respect to the common reference; each channel measures then the difference between two coupled electrodes. The three resulting signal are then amplified using an INA (whose gain can be set between 20 and 200dB), sent through a high-

pass filter with cut-off frequency at 1.5Hz and a Sallen-Key anti-aliasing filter (double pole at 150Hz). Finally a PIC16F688 microcontroller samples the signal at 237Hz with his 10bit ADC and transfers the output to the computer through a FT232RL interface (RS232 to USB converter) for further processing.

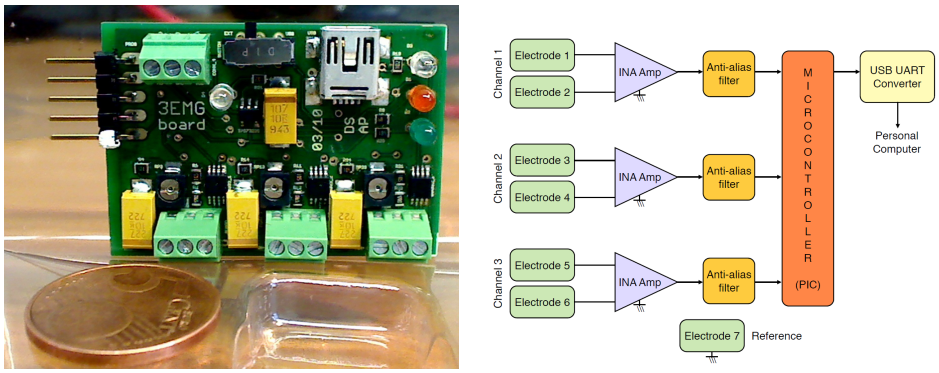


Figure 4.3: Eracle board. Photo from [30]

The output of the board is an ASCII string where each rows contains the data for each channel. The following is an example of the stream:

```
I:a b c
D:447 502 988
D:323 471 1011
D:220 666 567
D:438 519 530
D:327 834 816
...
I:a b c
```

where `\rD:` separates consecutive samples and `\rI:a b c` is repeated every hundred samples for synchronizing purposes.
The device is extremely small and lightweight (29×45×9mm and weights 35g) and can fit in a pocket. His production cost is relatively low, making it suitable for consumer electronics. Many sources in literature models the signal as a stochastic Gaussian process between 0 and 500Hz, with the most relevant frequencies between 50 and 150Hz ; having the board a sample rate of 237Hz accordingly to the Nyquist sampling theorem we can catch frequencies up to 118.5Hz.

4.2.2 Gestures selection

The very first step was to select the movements we are interested in, since it determines how electrodes will be placed.
In precedent works were selected the most common gestures performed during daily living activity: hand opening, hand closing, wrist flexion and extension, thumb abduction and opposition, index extension [15]. This set was maintained with the goal to compare the new system with the old one. A second set of gesture was then selected based on the movements the exoskeleton is able to perform: hand opening, hand closing and precision grasp (alias: pinch) [8].
As seen from chapter 2.4 the muscles we are most interested in are the *flexor digitorum profundus* and *superficialis*, since they are responsible of most of the force generated for fingers and wrist flexion, the *extensor digitorum communis*, responsible for fingers and wrist extension, and the *flexor pollicis longus* and the *abductor pollicis brevis* for thumb movement.

4.2.3 Electrodes

The choice for electrodes was intentionally limited to the ones commercially available for electro-stimulation. The reason is they are easily available at orthopaedic stores and their signal keeps an acceptable signal-to-noise ratio (SNR) even if used multiple times (while medical-level electrodes are usually mono-use). As the surface increases the electrode becomes more prone cross-talk, while smaller surface we can read a cleaner signal and allows us to place the electrodes nearer, decreasing the noise even more. We ultimately opted for 32mm circular electrodes (the smallest model we could find) with 0.60€price per piece.
The first channel was placed in the volar side of the forearm along the *flexor digitorum profundus* and *superficialis*, with the aim to highlight the muscle activity that leads to hand closing and wrist flexion. In a similar way the second channel was placed over the *extensor digitorum communis*, to acquire signal that characterizes finger and wrist extension.

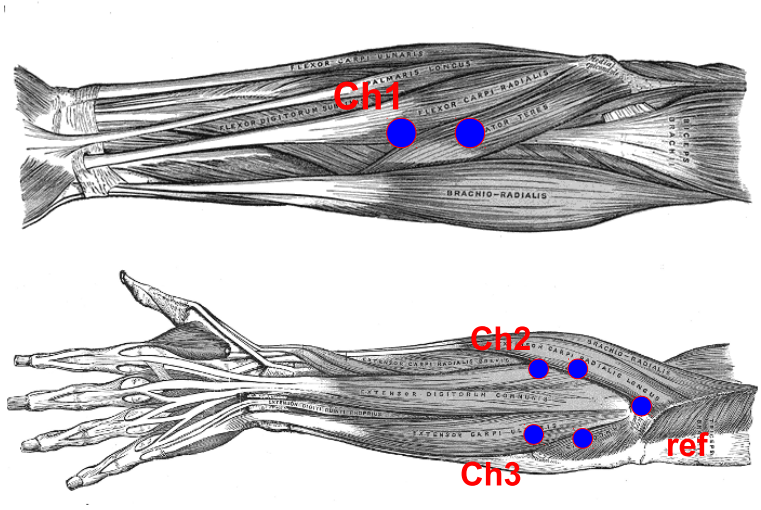


Figure 4.4: Electrodes placement, volar and dorsal view

In an initial attempt the third channel was placed over the *flexor pollicis longus* and the *abductor pollicis brevis* in order to better read the features that lead to thumb movement, but these muscles resulted to be too small and deep to generate a measurable signal. We ultimately opted to place the last channel in the internal dorsal part of the forearm, forming a triangle configuration with the other two channels in order to acquire signals from the entire organ, relying on the classifier's generalization capability.

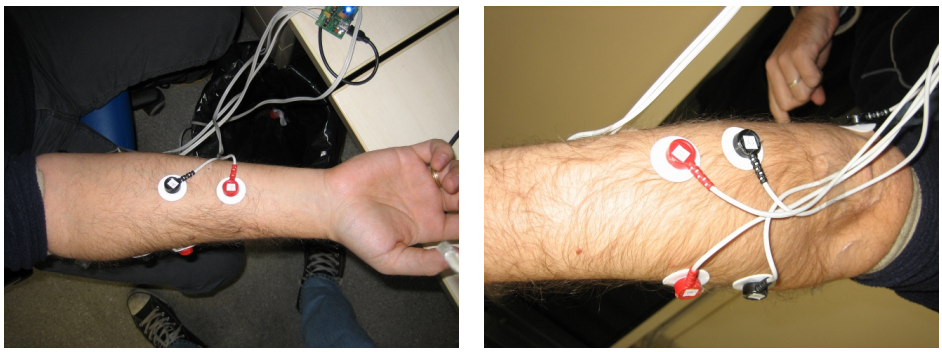


Figure 4.5: Electrdes placement, volar and dorsal view

The function of Eracle's common reference is to detect the noise coming from the body and subtract it from the other channels. It was then placed

on the elbow since this under area is far from any involved muscle, so the electrical activity recorded should be due to noise only.

Finally wires were fixed to the skin in proximity of the clip using adhesive tape, preventing them from causing artifacts in the signal moving the electrodes or, worse, tearing them out.

4.2.4 Parsing

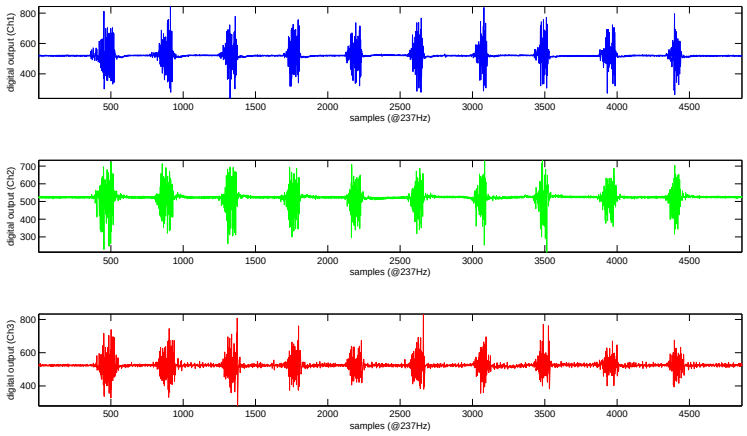


Figure 4.6: Signal parsed from the board

For an easy interaction with Eracle an appropriate interface was written: it consists in a Matlab class which handles serial communication, logging and output parsing. The signal is acquired from the board using Matlab's *Serial Port Interface* and converted from ASCII string to a $N \times 3$ matrix, where N is the number of acquired samples and the values are unsigned integer representing the 10bit encoding of the signal. The input buffer is set to contain up to 1s of samples, so the computation time between two consecutive pools to the board must be under this interval to guarantee signal continuity.
These are the step performed by the parser every time it is called between two consecutive readings:

- Concatenate the incomplete sample from the previous acquisition and the new string.

- Search for `D:`s in the string.
- For each `D:` (except the last) convert into integers the three following numbers.
- Save the substring starting from the last `D`, since it will contain an incomplete sample. It will be forwarded to the next acquisition.

#read.	prev. chunk	serial read	parsed signal

n	`D:519 4`	`96 326`
		`D:515 513 576`
		$\cdots$
		`D:499 515 647`
		`D:504 49`
			519 496 326
			515 513 576
			$\cdots$
			499 515 647

$n+1$	`D:504 49`	`6 324`
		`D:524 504 408`
		$\cdots$
		`D:512 518 612`
		`D:49`
			504 496 324
			524 504 408
			$\cdots$
			512 518 612

$n+2$	`D:49`	`2 524 601`
		`D:496 509 473`
		$\cdots$
		`D:524 497 526`
		`D:504 520`
			492 524 601
			496 509 473
			$\cdots$
			524 497 526

Table 4.1: Example of parsed samples. The incomplete sample from the serial (in red) is prefix to the successive read to guarantee signal continuity

4.3 Classification

The acquired EMG is analysed by the classification module. It consists in a class that encapsulates the parsed signal and provides the methods for extracting the features which will be used by the neural network for signal recognition.

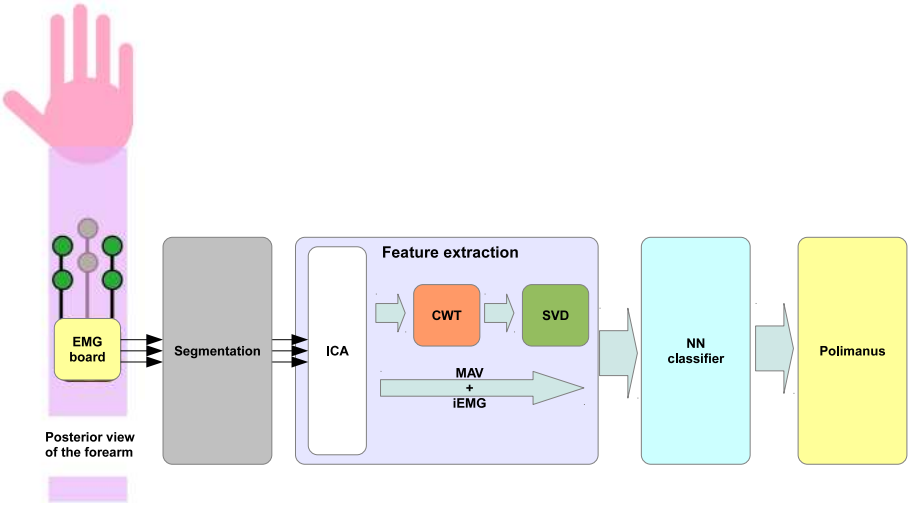


Figure 4.7: Schema of the recognition system

The steps performed for burst detection and identification are listed in the following subsections.

4.3.1 Segmentation

Before identifying a movement we must detect it and define his boundaries. As we can see from figure 4.8 the forearm is idle all we read is random noise, but when a movement is performed the features of the signal change, most notably the amplitude. The first step for burst detection is to subtract signal mean and then rectify it.

$$\text{rect}_{c,i} = \left|\text{emg}_{c,i} - \overline{\text{emg}}_c\right| \tag{4.1}$$

where $\text{emg}_{c,i}$ is the i-th sample on the c-th channel.
This way we can perform a rough detection simply thresholding features such as the moving average or the integral signal. The downside of these methods is they are sensitive to high-frequency noise, so the next step is to filter the signal applying a second-order low-pass Butterworth filter (cut-off frequency at 2Hz) in order to smooth the signal.

This way we can determine the beginning of a movement (*burst's head*) using lightweight techniques such as fixed thresholding (to ignore the residual noise energy) and checking his derivative (to cut-off small oscillations due

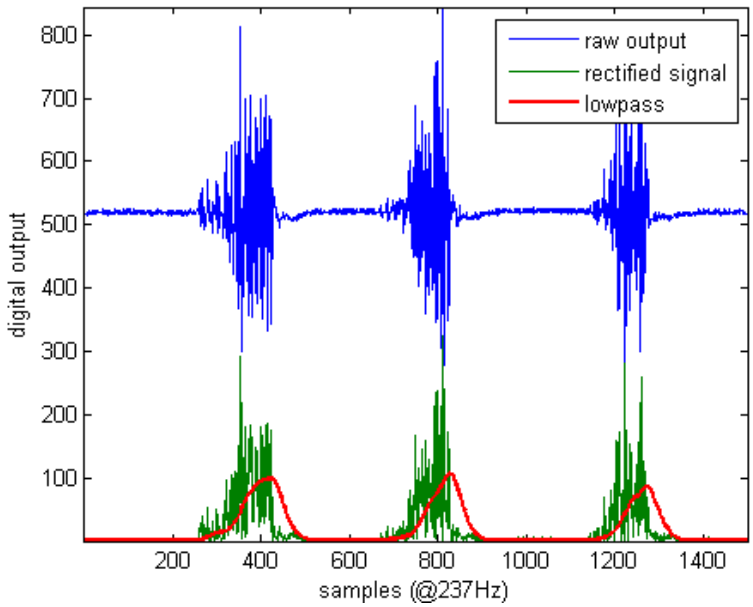


Figure 4.8: Segmentation over three hand closings on channel 1

to artifacts). Instead we can consider a movement ended (*burst's tail*) when the signal or his derivative drops under a fixed value. Fixed values were found empirically and may vary depending on the gain set on acquisition board.
A burst is considered open while activity in any of the channels is detected, and closed when activity does not satisfy the parameters anymore in any of the channels. The burst will be then further analysed starting fifty samples before the first one detected: this preamble is added to catch possible early low-amplitude dynamics that would otherwise be lost.

4.3.2 Independent component analysis

Independent component analysis is a technique that allows extracting N independent source signals from at least N samples containing linear mixtures using multivariate statistical analysis. Given enough channels we could theoretically eliminate cross-talk separating the activity of every single muscle. We used then the FastiICA package using Hyvarinen's fixed-point algorithm[1] with the goal to separate the activity of the most significant muscles, thus leading to a better separability of the extracted features. The number of

[1]FastICA packages: `http://www.cis.hut.fi/projects/ica/fastica/`

muscles however is much larger than the number of channels, so one of the prerequisites of ICA is missing. We obtained results similar to [29], where channels were not independent leading to a mixing matrix A far from being dominant, which means we can't associate a predominant signal to every channel.
Since we could not successfully separate the sources we tried then to use ICA for denoising: this was achieved rebuilding the acquired signal with the most significant contributions, applying an adaptive thresholding to the matrix of weights.

$$a_{i,j} = \begin{cases} a_{i,j} & \text{if } |a_{i,j}| \geq \alpha \max_j \left(|a_{i,j}|\right) \ , \ \alpha \in [0,1] \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

This way the reconstructed signal will contain the most relevant contributions, discarding the other sources as less significant.

ICA is performed when a burst is detected. Since the FastICA is a fixed-point algorithm the weighting matrix is stored and used as initial value during further ICA analyses, thus allowing the algorithm to reach the solution in fewer steps (an average of ten iterations against the around twenty needed starting from random values). The matrix is then reset after the closing of the burst.

4.3.3 Features extraction

Once the boundaries of the burst have been determined we must extract some numerical parameters for gesture classification, creating a *feature vector* to feed the neural network.

Temporal features

In this work were maintained the features used in the previous work. First are considered the mean absolute value (MAV) and his integral (iEMG): they were chosen since they are closely related to signal energy and different gestures generate very different activities on different channels caused by different patterns in recruitments of MU, mostly in terms of number of units and localization with respect to the electrodes. For example hand closing will generate a strong signal on the channels placed on the volar side

of the forearm, while fingers hyper-extension will generate more energy in the dorsal channels. Only using these features we can obtain a classification with an 80% success rate.

$$\mathrm{iEMG}_c = \sum_{i=1}^{N} \mathrm{emg}_{c,i} \tag{4.3}$$

$$\mathrm{MAV}_c = \frac{\mathrm{iEMG}_c}{N} \tag{4.4}$$

Wavelet transform

Gestures differ also in frequency patterns but, as discussed into chapter 3.2, a pure frequency representation would lose too much information because of the non-stationary nature of the EMG; a time-domain analysis was then adopted using the Continuous Wavelet Transform, which is able to retain the temporal localization of the shifting frequency features. This is obtained by the convolution of the signal with several version of the same signal (mother wavelet) scaled and stretched by different sizes: the more similar the frequency of the stretched wavelet to the analysed signal segment, the higher the result of the computed coefficient.

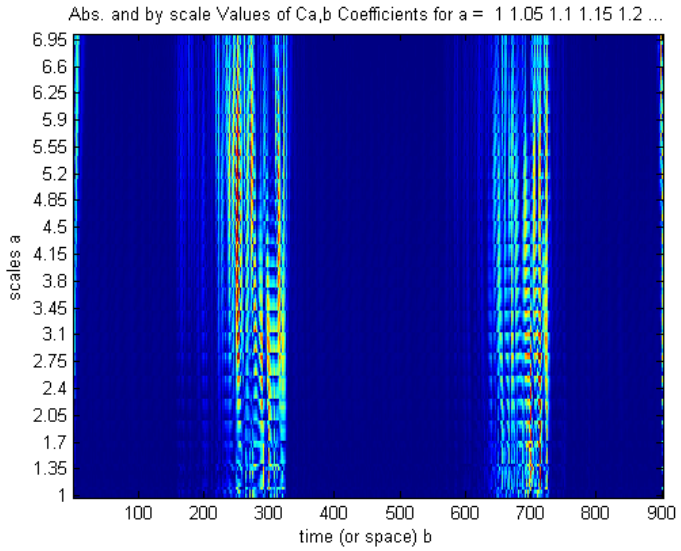


Figure 4.9: Example of CWT over two bursts

An important factor in wavelet analysis is the shape of the mother wavelet: the more the shape matches the nature of the signal, the better the analysis

performances. For some wavelet families must be also taken in consideration the number of *vanishing moments* (or *level*): this parameter is related to the order of polynomials the wavelet is able to represent; with this value increases his regularity, but also his computational requirements.
We decided to choose the Daubechies wavelet family (while previously Morlet was used), since it has a wide application in biosignal analysis. To determine the number of vanishing moments we evaluated empirically how performances vary with this parameter, considering the mean reconstruction error applying wavelet compression (see C) on a set of bursts. The computation time increased almost linearly with the moment, but at level four (`db4`) the error stopped to decrease significantly, so this wavelet was kept.

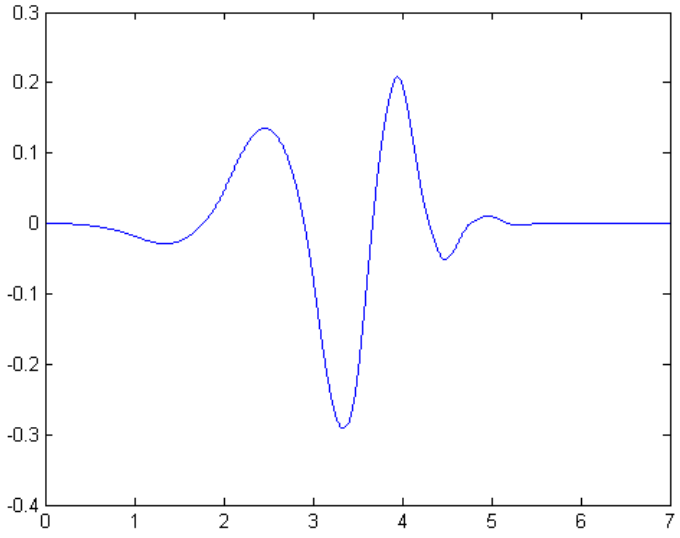


Figure 4.10: 'db4' wavelet

The last parameters we need for the analysis are the scales of the wavelet: these values determine which frequencies the analysis will be centred on, and depend on the base frequency of the chosen mother wavelet and the sample rate of the electromyograph. These values were then manually selected using the tool offered by Matlab Wavelet Toolbox: we picked up only five coefficients to limit computation time, chosen in a way to uniformly scan the range of frequencies we are interested in. The resulting scales were `[1.5:6.5]`.

Dimensionality reduction

The CWT for an N-samples signal generates a $M \times N$ matrix, where M is the number of scaling coefficients. For an average burst of 200 elements this means 1000 features, so a dimensionality reduction is needed. As introduced in[32], we exploited SVD decomposition, an algorithm widely used in many fields for tasks such as signal processing, data compression, noise reduction or PCA. It factors any $M \times N$ matrix in three different matrices.

$$M = U\Sigma V^T \tag{4.5}$$

where U is a $M \times M$ orthogonal matrix, Σ a diagonal rectangular matrix with nonnegative values and V^T is the conjugate transpose of an orthogonal $N \times N$ matrix V.
We decided to use as feature the Σ matrix, whose diagonal contains in descending order the singular values of the transform: the resulting vector still contains information related to the original coefficient matrix, but compressed in a much smaller $M \times 1$ vector (five values in our case).
Therefore the neural network classifier will receive as input the *feature vector* composed by seven elements: MAV, iEMG and the five singular values.

4.3.4 Neural network classification

The classifier is a model able to identify patterns found in samples of input-output couples (in our case the output is a label associated to a class) and use it to determine the output of new data (*supervised learning*).
Among all the techniques available we chose to use artificial neural networks because of their ease in the setup process. An ANN is a bio-inspired mathematical model represented by an interconnected group of *neurons*. Each neuron takes as input the weighted sum of the other neurons, and outputs a value accordingly to a given *activation function* which will be forwarded as input to other neurons. The *universal approximation theorem* states that, given enough neurons, multilayer feed-forward networks can approximate any continuous function with an arbitrarily small error (proved for several types of activation functions).
Learning is obtained finding the optimal set of weights so that the network is able to minimize the error between the *target output* and the current output; this is obtained through different methods, such as gradient descent.Usually a subset of the samples (*validation set*) is kept out the training procedure

with the goal to use it to test the generalization capability of the trained network. In our case we need to train a pattern recognition NN: it takes as input the previously discussed feature vector and has as target the label of the related gesture. The output of the network however is a real value, so a method to encode the classes is needed: the net is designed to output a number of values equal to the number of classes, and the training target is a vector of zero values with the exception of the i-th element (associated to the i-th gesture) which is set to one. This way the NN learns to assign a value close to one to the element associated to the identified gestures, while the remaining elements of the vector will have near-zero values.
The network has a feed-forward topology with one hidden layer composed by thirty-five neurons (the number of neurons was limited to avoid overfitting) with a tan-sigmoid transfer function (a faster approximation of the hyperbolic tangent). Training set was partitioned using 75% of the data for the actual training, 15% of the samples for validation and the remaining 10% for external testing.As learning algorithm *Levenberg-Marquardt* (LM) was selected, since it is the fastest back-propagation algorithm [52], with early stopping (to avoid overfitting).

4.3.5 Training protocol

To make tests consistent a common protocol for gesture acquisition is required, so we inspired from the one proposed in [15]; to facilitate this process a GUI was created to guide the user through the acquisition and automate the training task (see appendix B).
Each gesture is acquired with three repetitions, each containing ten movements. After each repetition the user must relax the arm for thirty seconds in order to avoid fatigue, which is cause of signal degradation [15], eventually slightly moving the electrodes to achieve greater cross-session robustness. When a set of repetition is complete the user must relax for one minute, and then proceed in the same way with the next gesture.
When the acquisition phase is complete the interface stores the raw data of each repetition in a text file properly tagged. Each file has then to be parsed and manually inspected to detect and remove unwanted artifacts that could affect the training. Once the signals are clear the automatic training function is called: it automatically detects the bursts and extracts the related features and trains the ANN as specified in subsection 4.3.4 using the built-in Matlab functions.
The resulting network and the related training records are then stored in a

file.

4.4 Polimanus

Polimanus3 [8] is the last of a series of hand rehabilitative prototypes developed within AIRLab. It consists in a polystyrene bracer ending with a glove. A series of Bowden cables connects the fingertips to two Hitec servos HS-805BB (position and speed controllable, 180°rotation, capable to produce a force of 19.8 kg/cm) transmitting up to 105N of force, helping the patient to perform hand opening, closing and pinch movement. The actuators are controlled by an 8MHz Microchip PIC18F452 microcontroller, which reads also from the two potentiometers placed to measure the position of the gears. Communication is provided by a FTDI232RL which allows connecting Polimanus to the host computer via USB.

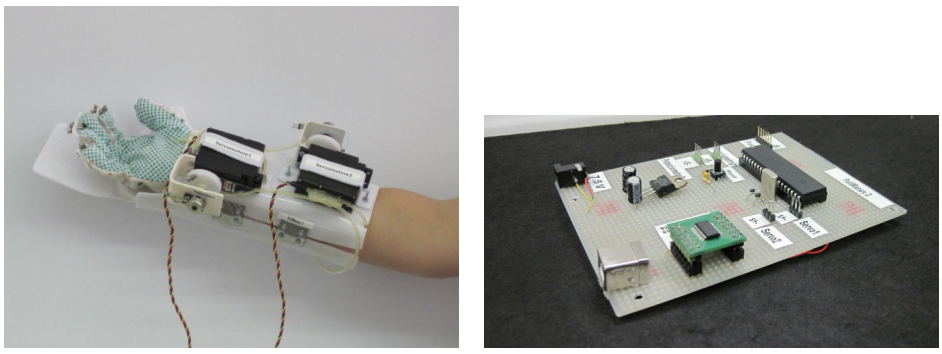


Figure 4.11: Polimanus (left) and the control board (right)

The communication protocol is based on the exchange of unsigned 8bit integers:

```
PC [x> PIC (byte sent from PC to PIC)
PC <x] PIC (byte sent form PIC to PC
-----------------------------------------------------
PC [253> PIC (command new position)
PC <253] PIC (ACK, PIC ready to receive 16 bytes)
PC [pos0> PIC (new position servo 0)
PC [pos1> PIC (new position servo 1)
```

```
PC [pos2> PIC (new position servo 2, unused)
...
PC [pos7> PIC (new position servo 7, unused)
PC [vel0> PIC (speed servo 0)
PC [vel1> PIC (speed servo 1)
...
PC [vel7> PIC (speed servo 7, unused)
PC <ACK] PIC (checksum received bytes)
-----------------------------------------------------
PC [251> PIC (command read voltage on analog port)
PC [port> PIC (analog port number)
PC <ADRESH] PIC (most significant byte)
PC <ADRESL] PIC (less significant byte)
-----------------------------------------------------
+ PC [250> PIC (command read voltage analog port 0 and 1)
PC <ADRESH] PIC (most significant byte port 0)
PC <ADRESL] PIC (less significant byte port 0)
PC <ADRESH] PIC (most significant byte port 1)
PC <ADRESL] PIC (less significant byte port 1)
```

The device presents several flaws in his design, mostly due to the low-cost hardware employed: the overall structure results uncomfortable and the housing of the artificial tendons are anchored in a position which allows only the half-closing of the hand. The high torque of the actuators makes the device not compliant, and the type of sensor equipped does not allow detecting user-generated forces.

To control this device an appropriate interface was created during this work: it consists in a Matlab class which masks serial communication and provides high-level methods to control the movements of the exoskeleton (see appendix B.5).

4.5 Online classification

Offline classification, used during the training phase, consists in acquiring long batches of samples which will be stored and then analyzed. On the contrary, in online classification the analysis is performed as soon as enough data are acquired in order to provide an interactive behavior. Since only the samples over a limited time-window are needed to achieve a continuous

control, a policy to flush the sample buffer is needed.

During online classification the EMG board is periodically polled for new samples, the new chunk is appended to the data already stored in the buffer and the updated signal is then processed using the algorithms previously described. If any burst is detected during the segmentation further analysis are executed: if the detected burst is at least 120 samples long then, accordingly to the tests performed in section 4.1, is considered long enough to produce a reliable output so features are extracted, classified and the resulting command is forwarded to Polimanus; otherwise if the burst has less than 120 samples only ICA is performed in order to compute the mixing matrix A, which will be used by the ICA in the next iteration as starting point, with the goal to converge to the solution with fewer steps (an average of ten steps starting from a good matrix compared to the twenty needed to converge from a random initial matrix).

Before proceeding with the acquisition of the next signal segment we need to understand what we need to keep of the old signal.

- If no burst is detected only the last 100 samples are kept, since are needed to fill the preamble of any potential new burst detected.
- If an incomplete movement was detected the signal is flushed only up to the head of the burst.
- If an already completed movement was detected the buffer is flushed up to the tail of the burst, in order to avoid it is analyzed again.

The signal buffer is implemented in Matlab using a dynamic array, able to increase to accommodate the activity of the currently detected movement in all his length and shrinking again as it ends. For the typical range of the buffer (100-300 samples) Matlab is able to manage resizing without a significant overhead.

4.6 Matlab implementation

The following subsections summarize how the algorithms presented in the previous sections were implemented and organized in Matlab. A more detailed description is provided in the appendix B.

Profiling highlighted how the functions for continuous wavelet analysis and neural network simulation, `cwt` and `network.sim`, were the bottleneck of the online classifier. These functions were then optimized accordingly to our needs, leading to a significant speedup (see tests in section 5.2.3 and 5.2.4).

4.6.1 Core modules

The following classes implement the functionalities of signal acquisition, classification and control of the external device. They were implemented in a way to improve the modularity and reusability of the system, making the development of the more high-level functions for training and online classification easier. Modularity allows interfacing the controller with possible new devices (such as an alternative EMG board or another prototype of exoskeleton) almost effortless.

Signal acquisition It is performed by the `emgboard` class which handles serial communication with Eracle and provides the parser which takes as input the ASCII stream and outputs a matrix containing the signal.
In the initial phase of the project an EMG board emulator, `dummyboard`, was created extending `emgboard`: it provides a graphical interface which allows the user to select a gesture, whose activity is simulated outputting at the same rate of the original board the samples of a pre-recorded burst loaded from file. The inactive state instead is simulated generating random low-amplitude noise. This emulator allowed an interactive testing of the online classifier in his early stage, thus limiting the waste of electrodes.

Features extraction It is performed by the `emgsignal` class, which encapsulates the signal and implements the previously discussed algorithms for segmentation to detect the boundaries of the bursts and for features extraction. It also provides the methods which automatically manages the length of the analysis buffer during online classification.

Classification It is performed by the `emgnet` class, which extends the built-in `network` and overrides the `sim` function: software profiling highlighted a series of computationally demanding instructions within the initialization phase whose execution time, for the size of the network we are dealing with, was an order of magnitude greater than simulation itself. This segment of code was then modified caching the results of redundant instructions and properly structuring the input parameters to avoid costly

conversions; the final result is a function with almost no overhead whose execution time is determined only by the activation functions (see section for the achieved speedup).

Polimanus The class `polimanus` handles serial communication with the rehabilitative exoskeleton, thus giving to the programmer a high-level interface to control the movement of the device. It provides methods which automatically perform the predefined movements, hand opening, closing and pinch, plus the possibility to manually define the position and speed of the actuators to execute custom movements.

4.6.2 Training

System training is performed in two stages, implemented by two different functions.
The first stages is the acquisition of the training samples: the function `farmData` guides the user through the acquisition protocol (proposed in 4.3.5) while acquires the signal using through `emgboard`; the samples are then automatically stored and tagged, so the user can manually remove the artifacts.
The second phase is the training itself: `trainNN` loads, parses and analyses (through `emgsig`) the stored batch of signals. The features extracted from the found bursts are then used to train a neural network for patter recognition calling the built-in toolbox. The resulting nets are then stored to be later used by the online classifier.

4.6.3 Online classification

Online classification is performed by the `onlineRecognition` function: in the initialization phase loads the stored network and connects to the external devices through the apposite interfaces.
Classification is then performed continuously polling the EMG board and analyzing the acquired signal through `emgsig`. If any burst is found further analysis are performed, extracting and classifying the features, using the output label to determine the control variable to forward to Polimanus.
The signal buffer is finally flushed accordingly to the policy presented in section 4.5.

Chapter 5

Results

This chapter describes how the designed system is able to classify the performed gestures within the specified time constrains. Profiling was initially executed offline on a batch of data with the goal to estimate the delay introduced by signal analysis only. Further tests were then performed to determine the minimum signal length needed to obtain a reliable classification.
Finally online classification was tested with the goal to determine the performances of the overall system considering the classification rate over three gestures (the ones performed by Polimanus) and the user perceived delay.

Tests were performed over a P4-2.4GHz PC with 4GB of ram and running Matlab 7.11 for Windows. Execution times were profiled using the built-in performance tools.

5.1 Recognition anticipation

This test was conceived to determine if is possible to have a reliable classification using only an incomplete burst and eventually which are the limits, thus significantly reducing the response time of the exoskeleton.
The same batch of movements used for the previous profiling (using thirty repetitions over seven gestures) was used to train five neural networks with a recognition rate between 95% and 97.5%. Classification was then performed using the testing set containing the 10% of the bursts. Bursts had an average length of 197 samples (28 std), where, as already pointed out, the effectively detected burst start from the hundredth sample, and the previous ones are kept to catch possible early low-amplitude dynamics that would otherwise be lost.

Features where extracted from the initial segments with increasing lengths and classified using the five nets. Recognition rate over burst percentage are displayed in figure 5.1.

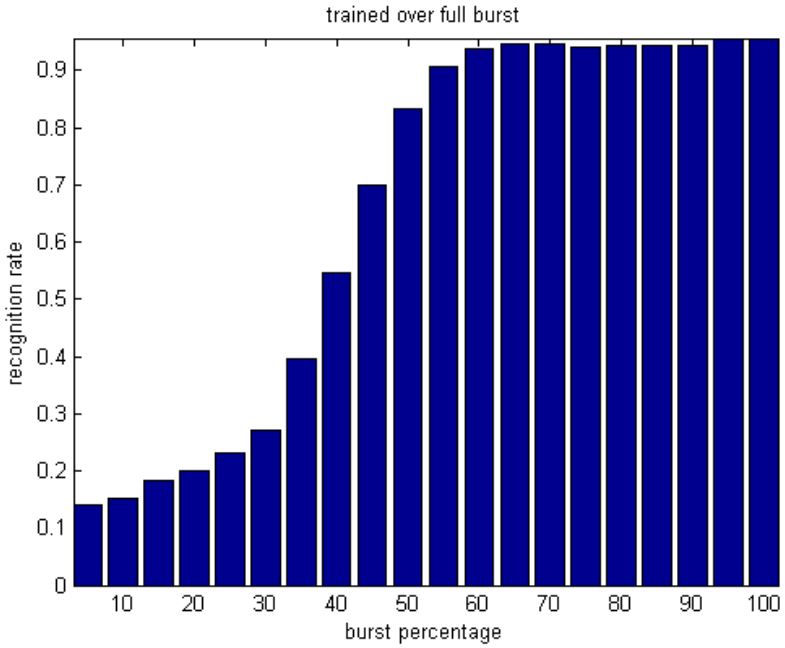


Figure 5.1: Recognition rate over burst

We note that the system is unable to successfully identify the gesture in the first half of the burst, which is right at the end of the preamble, where the activity of the signal starts to build up. At 60% of the burst we obtain a recognition rate near 90%, further increasing until reaching the nominal rate at full burst. This test suggested that is possible to achieve a reliable classification using only the 20-30 samples recorded right after the segmentation algorithm detects the burst.

The test was then repeated over a set of networks trained following the same procedure, but using only the first half of the bursts: the figure 5.2 displays how the recognition rate reaches his peak around 50% of the length, then slowly decreasing for longer segments. This was expected, since the network performs best around the very same burst length used for his training, while obtaining lower scores as the features slowly change as the signal length increase.

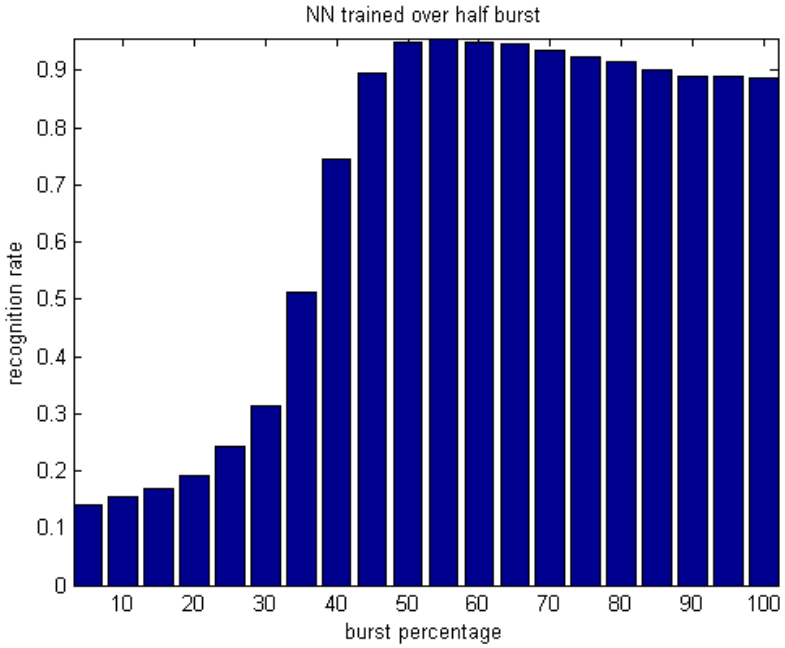


Figure 5.2: Recognition rate over burst percentage

Repeating the test training the burst over segment even shorter made no sense, since that would mean using features that would show before the system is able to detect the burst.

These tests showed how in a early stage we can already have a recognition rate close to the one obtained analyzing full bursts, around 100ms after the effective detection of the burst. This allows us to forward a control variable to the exoskeleton before the movement is complete.

In a real interactive application however the delay introduced by the EMG board, the serial communication overhead and the mechanical latency of the exoskeleton should also be taken into account, but these aspects cannot be precisely measured using the same tools used within this test. They will be considered in section 5.3, providing a experimental measure observing the system behaviour.

5.2 Delay analysis

The goal of this set of tests is to determine the computation time required by the several steps of the proposed algorithm during online classification, with the goal to estimate elapsed time between gesture execution and the command of the control of the exoskeleton. These preliminary tests were executed using samples extracted from a batch of signals acquired during the previous work [15], tagged as Subject A. It consists in sixty repetitions for seven gestures (hand opening and closing, wrist extension and flexion, thump abduction and opposition, index extension), with an average length for the detected bursts of 200 samples (20 std).

5.2.1 Segmentation

As stated in section 4.5 the signal buffer length vary from 100 to up to 300 samples over three channels. To determine the delay introduced by the segmentation algorithm we performed the analysis over 100 segments randomly extracted from the acquisition batch, each one 300-sample long (to consider the upper bound).

Function	*Calls*	*Total time (s)*
`emgsig.findBursts`	100	0.894

Table 5.1: Profiling of emgsig.findBursts

We can observe that the average execution time is under 100ms. Among all segments 57 bursts were detected.

The same test was repeated over the complete batches of acquisition to see how computation time vary over long signals (such as during the training phase, where big batches are processed); they consist in 42 repetitions containing 10 movements each, with an average length of 41642 (492.2) samples.

Function	*Calls*	*Total time (s)*
`emgsig.findBursts`	42	3.241

Table 5.2: Profiling of emgsig.findBursts

5.2.2 Independent component analysis

To compute the performances of ICA the present test was performed profiling the execution of this function over the burst extracted from the acquisition set. First for each burst were analysed only the first 120 samples saving the computed mixing matrix, then ICA was performed over full burst starting from both the computed A matrix and a random value. The average length of a full burst is 197 (28 std) samples.

Function	*Calls*	*Total time (s)*
`fastica` (first 120 samples)	420	6.326
`fastica` (full burst, precomputed A)	420	2.138
`fastica` (full burst, random A)	420	4.351

Table 5.3: Profiling of ica

The results over full bursts result suggest that what really improves the convergence is the greater amount of information given by a longer signal rather than pre-computing the unmixing matrix, which leads nonetheless to interesting improvements. The high execution time of ICA on short signals brings anyway into question the real utility of having an initial guess of A: if the signal is too short for a meaningful classification the system would then proceed polling the EMG board for new samples, but that would mean an average of 10ms (for `emgsig.findBurst`, as showed the previous test) are past from the last access to the board, thus obtaining only two new samples; thus the new signal won't be very different from the previous one, so computing an initial guess for the unmixing matrix helps to slightly increase time between acquisitions and have more new samples (a mild delay balanced by a speed-up in further iterations).

5.2.3 Feature extraction

The set of data obtained from the last test was used to measure the computation time of the steps performed for feature extraction. `emgsig.extractFeatures` performs the analysis over three different channels (so the subfunctions are called three times as much). `cwt` was also included in the test with the goal to determine the speed-up of the customized `myCwt`.
We can see from table 5.4 how the customized wavelet analysis provides a significant speed-up if compared to the built-in function (six time faster); it increase considerably the performances of the system, since it is one of the heaviest bottlenecks. `myCwt` takes the 88.5% of all `extractFeatures` time,

Function	*Calls*	*Total time (s)*
`cwt`	1260	12.745
`myCwt`	1260	2.075
`svd`	1260	0.198
`iEMG + MAV`	1260	0.027
`extractFeatures` (`myCwt`, `svd`, iEMG, MAV)	420	2.449

Table 5.4: Profiling of emgsig.extractFeatures

`svd` takes the 8.4% and the remaining features, iEMG + MAV, take the 1.1%. Over a full burst we have an average computation time of 5.8ms.

5.2.4 Classification

This test was performed to compare the execution time of the custom `emgnet.sim` with the built-in `network.sim`. The features computed from the last test were used as inputs.

Function	*Calls*	*Total time (s)*
`network.sim`	420	28.6962
`emgnet.sim`	420	3.3631

Table 5.5: Profiling of emgnet.sim

We obtained a speed-up of 8.5, with an average execution time of 8ms. This is significant since the neural network is another bottleneck in the online classifier.

5.2.5 Complete analysis

The following test was performed executing the complete analysis using 100 segment of signals containing a burst.
Table 5.7 shows how a complete burst classification takes an average of 35ms, which leads to a frequency of 28 analyses per second. Should be noted that in a real online application `emgsig.extractFeatures` and `emgnet.sim` are not executed if no burst is detected, and if the length of the detected burst is under 120 samples only `ica` is performed, thus making the following analysis cycle faster; we can then consider the estimated frequency as a lower bound.

Function	*Calls*	*Total time (s)*
`emgsig.findBursts`	100	1.14
`emgsig.extractFeatures`	100	1.470
`emgnet.sim`	100	0.861
`ica`	100	0.77
`myCwt`	300	0.62
Total	100	3.47

Table 5.6: Profiling of a complete analysis

H. open	*H. close*	*Pinch*	*Total*
20/20	20/20	17/20	57/60

Table 5.7: Profiling of a complete analysis

In a real interactive application however the delay introduced by the EMG board, the serial communication overhead and the mechanical latency of the exoskeleton should taken into account. These aspects will be considered in the tests reported in the next section.

5.3 Online classification

To test online performance on the target device a new acquisition set was needed: following the procedure specified in section 4.3.5 new training samples were acquired. The recorded gestures were limited to the ones the device is able to perform (meaning hand opening, hand closing and pinch), so a total of ninety movements were recorded; the subject is a 26 years old male, with a height of 1.83m and weighting 73kg. With this set a classifier was trained, using cross-validation with 75% of the samples for training, 15% for validation and 10% for external testing; training the net resulted quite easy if compared to the previous set, due to the lower number of gestures and the high separability of their features (being very different movements).

The online classifier was launched and the user was asked to perform a sequence of randomly selected movements while recording the success rate. The classifier scored an overall recognition rate of 95% on the performed movements. Hand openings/closings scored a perfect recognition rate, while the delay recorded between the beginning of the movement and the first

response from the classifier was estimated to be around 200ms (including the delay introduced by the EMG board, serial communication, the time needed to gather enough samples for a reliable classification and finally the analysis itself), while subsequent cycles produced a new output (confirming the first one) every 20-30ms until movement completion.
Some issues were noticed with pinch: this movement generates a mild muscular activity, hard to detect until tetanus is reached (meaning only after the fingertips touch and start pushing against each other, generating a greater muscular activity). It was also the only movement who did not achieve a perfect recognition rate, mistaken for hand closing because of the relatively similar activation during tetanus. Identification could be easily improved with a more thoughtful choice of features, but the output would be inevitably too late if we are unable to detect it in his early stage.

A malfunction, probably located in the FTDI232RL, did not allow to add Polimanus to the online tests. During the early developing stage however the device was extensively tested to make sure his interface module is bug-free and to have an preliminary estimation of the delay introduced by serial communication and the response of his mechanical parts: the interaction with the device was filmed using a 60fps camera, and the recording was then analysed to determine the elapsed time between the instant the user sends the command and the beginning of the movement of the actuators. The delay was evaluated to be around 200ms, which in series with the online classifier would bring the overall response time around 400ms, slightly above the prefixed limit.
This delay however could be reduced through the development of a new device, with a faster communication and reducing the mechanical response time through a more lightweight structure.

Chapter 6

Discussion

6.1 Conclusions

In this work we demonstrated the feasibility of an interactive controller for a rehabilitative device based on the classification of the electromyogram through an early analysis of the burst. While the previous system relied his accuracy on the information extracted over complete gestures, we showed how is possible to obtain similar performances analyzing only a limited number of samples from the beginning of the detected movement; this way we reduced the perceived delay under 200ms, an interval which is under the commonly accepted limit for a smooth interaction.
Further increase in the time-performances were achieved by means of a meticulous optimization, which significantly decreased the analysis time, allowing to continuously generate further control variables every 20-30ms until the end of the movement.

Thanks to its new interactive capabilities the classifier can be integrated into a rehabilitative device, thus offering him the ability to implement an *assist-on-need* strategy based on the residual muscle signal, without the need of a force control that would make his design more complex and affect his production cost.
What we achieved is to significantly improve the quality of the rehabilitative training a cheap device like Polimanus can offer: from simple tool which forces the patient through a set of preprogrammed movements, promoting only joint plasticity, we obtained a device able to detect to user's intentions, quickly reacting to assist whatever movement is trying to perform. This new capability has several implications: forcing an active effort helps to regain muscle tone more quickly, while at the same time allows performing typolo-

gies of exercises aimed to the rehabilitation of the neuromuscular system; it is also suitable for highly-impaired patients, whose inability to perform any movement would not permit them to benefit from robotic rehabilitation through force-controlled devices. The benefits are also from the psychological point of view: putting the patient in a more active role helps him to feel in control of the exercise, significantly boosting his motivation since he would perceive the improvement as a result of his own effort, bringing a greater sense of gratification; moreover with this kind of control is the patient himself to decide the pace of the exercise, accordingly to his conditions.

The new architecture of the system was also conceived to improve his modularity, decoupling the core signal analysis from the specific implementations of the used external devices. This allows to easily adapt to the introduction of new devices, such as a new prototype of exoskeleton, or reusing the system for applications even outside the field of rehabilitation, such as prosthesis control, teleoperation or human computer interaction.

6.2 Further developments

The newly developed greatly enhanced the capabilities of Polimanus and the effectiveness of the therapy it is able to offer. However several important steps are needed to obtain a functional rehabilitative framework.

Computational speed

Matlab was used as environment because of his high performances and the huge availability of functions and toolboxes for every application. However better performances could be achieved porting the code on a microcontroller: a thorough optimization on the underlying architecture would allow faster execution, while the overhead given by serial communication would be significantly reduced; moreover the reduced size would make the controller suitable for wearable application.

Design of the rehabilitative device

The most pressing matter anyway is the need of a new exoskeleton: Polimanus demonstrated that is possible to realize a rehabilitative device using low cost materials, but with serious limitations. The structure is bulky and

uncomfortable, it allows performing only few typologies of movements, some in an incorrect way, the equipped sensors and actuators allow only gross control and some flaws in his design make him jam frequently.
A meticulous anatomical study must be performed with the goal to design a possibly lightweight device able to fit the human hand follow his movements, emulating his range of motion. Actuators must be dimensioned to generate the correct forces without the risk to hurt the patient forcing incorrect positions, and sensor must embedded do have a feedback from user-generated forces. A device with similar features would be far from being cheap at the present time, but as showed in chapter 3.3 there are several designs which can be used as inspiration for interesting trade-off.

Interaction with the medical field

Having a fully functionally rehabilitative or assisting device would allow to proceed with the last stage which is the ultimate step of the project, meaning the field test. This last phase however is not immediate since has serious implications from the legal and ethical point of view: since the system has to be tested on human beings, people hoping in a way to improve the quality of their life, so a thigh cooperation with the health care system is needed, allowing the synergy of medical and engineering competencies and guaranteeing the safety and dignity of the patients.

Rehabilitative robotics promises to bring significant improvements into the quality of life of victims of impairing accidents offering high intensity training and allowing to objectively compare the effectiveness of different therapies. Significant results had been achieved, but a solid framework of both knowledge and devices is yet to be established. More effort must be put into this this field.

Acronyms

DIP	Distal InterPhalangeal joint
DOF	Degree Of Freedom
EMG	ElectroMyoGraphy
FPS	Frames Per Second
HCI	Human Computer Interaction
ICA	Independent Component Analysis
iEMG	Integral EMG
IP	InterPhalangeal joint
LM	Levenberg-Marquardt
MAV	Mean Absolute Value
MCP	MetaCarpal Phalangeal joint
MU	Motor Unit
MUAP	Motor Unit Action Potential
MVC	Maximum Voluntary Contraction
NN	Neural Network
PCA	Principal Component Analysis
PIP	Proximal InterPhalangeal joint
sEGM	Surface ElectroMyography
SNR	Signal to Noise Ratio
SVD	Singular Value Decomposition

SVM Support Vector Machine

Bibliography

[1] American Heart Association. Heart disease and stroke statistics–2011 update. *Circulation*, December 2011.

[2] Società Italiana dell'Ipertensione Arteriosa. Ictus: i numeri in italia. `http://siia.it/i-numeri-in-italia/`. retrieved on Gen, 2012.

[3] Albert C. Lo, Peter D. Guarino, Lorie G. Richards, Jodie K. Haselkorn, George F. Wittenberg, Daniel G. Federman, Robert J. Ringer, Todd H. Wagner, Hermano I. Krebs, Bruce T. Volpe, Christopher T. Bever, Dawn M. Bravata, Pamela W. Duncan, Barbara H. Corn, Alysia D. Maffucci, Stephen E. Nadeau, Susan S. Conroy, Janet M. Powell, Grant D. Huang, and Peter Peduzzi. Robot-assisted therapy for long-term upper-limb impairment after stroke. *New England Journal of Medicine*, 362(19):1772–1783, 2010.

[4] L. Marchal-Crespo and D. J. Reinkensmeyer. Review of control strategies for robotic movement training after neurologic injury. *Journal of NeuroEngineering and Rehabilitation*, 6, 2009.

[5] H.I. Krebs. Rehabilitation robotics: An academic engineer perspective. In *Engineering in Medicine and Biology Society,EMBC, 2011 Annual International Conference of the IEEE*, pages 6709 –6712, 30 2011-sept. 3 2011.

[6] V. Squeri, A. Basteris, and V. Sanguineti. Adaptive regulation of assistance 'as needed' in robot-assisted motor skill learning and neurorehabilitation. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –6, 29 2011-july 1 2011.

[7] E.T. Wolbrecht, V. Chan, D.J. Reinkensmeyer, and J.E. Bobrow. Optimizing compliant, model-based robotic assistance to promote neurorehabilitation. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 16(3):286 –297, june 2008.

[8] Lucia Prisciantelli and Giuseppe Ventimiglia. Polimanus 3: prototipo di esoscheletro per la riabilitazione passiva della mano, 2010.

[9] Paolo Belluco, Dario Cattaneo, Giuseppina Gini, and Giuseppe Lisi. From the classification of emg signals to the development of a new lower arm prosthesis. In *Proceedings of the 18th IFAC World Congress, 2011*, 2011.

[10] Sylvia S. Mader. *Human Biology*. McGraw-Hill, 7th edition, 2001.

[11] Michael D. Johnson. *Human Biology - Concepts and Current Issues*. Pearson, 6th edition, 2001.

[12] E. N. Marieb and K. Hoehn. *Human Anatomy & Physiology*. Benjamin-Cummings Pub Co, 7th edition.

[13] Henry Gray. *Anatomy of the Human Body*. 20th edition, 1918.

[14] Emran M. Tamil, N. S. Bashar, M. Y. I. Idris, and A. M. Tamil. A Review on Feature Extraction & Classification Techniques for Biosignal Processing (Part III: Electromyogram). *4th Kuala Lumpur International Conference on Biomedical Engineering 2008*, pages 117–121, 2008.

[15] Giuseppe Lisi. The study of the electromyographic signal for the control of a prosthetic hand. Master's thesis, Artificial Intelligence and Robotics Laboratory, Department of Electronics and Information, Politecnico di Milano, 2010.

[16] M.R. Cutkosky. On grasp choice, grasp models, and the design of hands for manufacturing tasks. *Robotics and Automation, IEEE Transactions on*, 5(3):269–279, jun 1989.

[17] I.M. Bullock and A.M. Dollar. Classifying human manipulation behavior. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –6, 29 2011-july 1 2011.

[18] Michelle J. Johnson. Recent trends in robot assisted therapy environments to improve real-life functional performance after stroke, 2006.

[19] L. Dipietro, M. Ferraro, J.J. Palazzolo, H.I. Krebs, B.T. Volpe, and N. Hogan. Customized interactive robotic treatment for stroke: Emg-triggered therapy. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 13(3):325 –334, sept. 2005.

[20] Riccardo Secoli, Marie-Helene Milot, Giulio Rosati, and David J Reinkensmeyer. Effect of visual distraction and auditory feedback on patient effort during robot-assisted movement training after stroke. *Journal of NeuroEngineering and Rehabilitation*, 8(1):21, 2011.

[21] Leonard E. Kahn, Peter S. Lum, W. Zev Rymer, and David J. Reinkensmeyer. Robot-assisted movement training for the stroke-impaired arm: Does it matter what the robot does? *Journal of rehabilitation research and development*, 43(5):619–630, 2006.

[22] Sha Ma, M. Varley, Lik-Kwan Shark, and J. Richards. Emg biofeedback based vr system for hand rotation and grasping rehabilitation. In *Information Visualisation (IV), 2010 14th International Conference*, pages 479 –484, july 2010.

[23] D. Jack, R. Boian, A.S. Merians, M. Tremaine, G.C. Burdea, S.V. Adamovich, M. Recce, and H. Poizner. Virtual reality-enhanced stroke rehabilitation. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 9(3):308 –318, sept. 2001.

[24] N.S.K. Ho, K.Y. Tong, X.L. Hu, K.L. Fung, X.J. Wei, W. Rong, and E.A. Susanto. An emg-driven exoskeleton hand robotic training device on chronic stroke subjects: Task training system for stroke rehabilitation. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –5, 29 2011-july 1 2011.

[25] K. Momen, S. Krishnan, and T. Chau. Real-time classification of forearm electromyographic signals corresponding to user-selected intentional movements for multifunction prosthesis control. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 15(4):535 –542, dec. 2007.

[26] Marcello Mulas. Sviluppo di un esoscheletro per la riabilitazione della mano controllato per mezzo di segnali mioelettrici. Master's thesis, Artificial Intelligence and Robotics Laboratory, Department of Electronics and Information, Politecnico di Milano, 2004.

[27] M. DiCicco, L. Lucas, and Y. Matsuoka. Comparison of control strategies for an emg controlled orthotic exoskeleton for the hand. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 2, pages 1622 – 1627 Vol.2, 26-may 1, 2004.

[28] M. B. Raez, M. S. Hussain, and F. Mohd-Yasin. Techniques of emg signal analysis: detection, processing, classification and applications. In *Biol Proced Online. 2006*, March 2006.

[29] G.R. Naik, D.K. Kumar, and H. Weghorn. Ica based identification of sources in semg. In *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*, pages 619 –624, dec. 2007.

[30] Paolo Belluco, Monica Bordegoni, and Umberto Cugini. Eracle: Electromyography system for gesture interaction. In Michael Smith and Gavriel Salvendy, editors, *Human Interface and the Management of Information. Interacting with Information*, volume 6771 of *Lecture Notes in Computer Science*, pages 391–398. Springer Berlin / Heidelberg, 2011.

[31] G.R. Naik, D.K. Kumar, and H. Weghorn. Performance comparison of ica algorithms for isometric hand gesture identification using surface emg. In *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*, pages 613 –618, dec. 2007.

[32] M. Arvetti, G. Gini, and M. Folgheraiter. Classification of emg signals through wavelet analysis and neural networks for controlling an active hand prosthesis. In *Rehabilitation Robotics, 2007. ICORR 2007. IEEE 10th International Conference on*, pages 531 –536, june 2007.

[33] K. Mahaphonchaikul, D. Sueaseenak, C. Pintavirooj, M. Sangworasil, and S. Tungjitkusolmun. Emg signal feature extraction based on wavelet transform. In *Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON), 2010 International Conference on*, pages 327 –331, may 2010.

[34] N.M. Kakoty and S.M. Hazarika. Recognition of grasp types through principal components of dwt based emg features. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –6, 29 2011-july 1 2011.

[35] N.M. Kakoty and S.M. Hazarika. Recognition of grasp types through principal components of dwt based emg features. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –6, 29 2011-july 1 2011.

[36] MathW orks. *Wavelets: A new tool for signal analysis.*

[37] S.Z. Mahmoodabadi, A. Ahmadian, M.D. Abolhasani, M. Eslami, and J.H. Bidgoli. Ecg feature extraction based on multiresolution wavelet transform. In *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, pages 3902 –3905, jan. 2005.

[38] D. Moshou, G. Papaioannou, I. Hostens, and H. Ramon. Wavelets and self-organising maps in eletromyogram (emg) analysis. In *European Symposium on Intelligent Techniques (ESIT 2000)*, 2000.

[39] B. Hudgins, P. Parker, and R.N. Scott. A new strategy for multifunction myoelectric control. *Biomedical Engineering, IEEE Transactions on*, 40(1):82 –94, jan. 1993.

[40] K. Englehart, B. Hudgin, and P.A. Parker. A wavelet-based continuous classification scheme for multifunction myoelectric control. *Biomedical Engineering, IEEE Transactions on*, 48(3):302 –311, march 2001.

[41] Tong-yi Chen, Zhong-wei Chen, Zhong-hua Gao, Tian-pei Hu, Xiaoming Xu, Yu-pu Yang, Zhang Jian, and Xiao-wen Zhang. Clinical detection and movement recognition of neuro signals, march 2005.

[42] P. Parker, K. Englehart, and B. Hudgins. Myoelectric signal processing for control of powered limb prostheses. *Journal of Electromyography and Kinesiology*, 16(6):541 – 548, 2006.

[43] Zheng Li. Using robotic hand technology for the rehabilitation of recovering stroke patients with loss of hand power. Master's thesis, North Carolina State University, 2003.

[44] L. Lucas, M. DiCicco, and Y Matsuoka. An emg-controlled hand exoskeleton for natural pinching. *Journal of Robotics and Mechatronics*, 16(5):482 – 488, 2004.

[45] A. Wege and G. Hommel. Development and control of a hand exoskeleton for rehabilitation of hand injuries. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3046 – 3051, aug. 2005.

[46] M. Bouzit, G. Popescu, G. Burdea, and R. Boian. The rutgers master ii-nd force feedback glove. In *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2002. HAPTICS 2002. Proceedings. 10th Symposium on*, pages 145 –152, 2002.

[47] D. Jack, R. Boian, A.S. Merians, M. Tremaine, G.C. Burdea, S.V. Adamovich, M. Recce, and H. Poizner. Virtual reality-enhanced stroke rehabilitation. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 9(3):308 –318, sept. 2001.

[48] J.M. Ochoa, Yicheng Jia, D. Narasimhan, and D.G. Kamper. Development of a portable actuated orthotic glove to facilitate gross extension of the digits for therapeutic training after stroke. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 6918 –6921, sept. 2009.

[49] J.M. Ochoa, M. Listenberger, D.G. Kamper, and Sang Wook Lee. Use of an electromyographically driven hand orthosis for training after stroke. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –5, 29 2011-july 1 2011.

[50] HyunKi In, Kyu-Jin Cho, KyuRi Kim, and BumSuk Lee. Jointless structure and under-actuation mechanism for compact hand exoskeleton. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –6, 29 2011-july 1 2011.

[51] K. Englehart and B. Hudgins. A robust, real-time control scheme for multifunction myoelectric control. *Biomedical Engineering, IEEE Transactions on*, 50(7):848 –854, july 2003.

[52] MathWorks. *Neural Network Toolbox™Documentation.*

Appendix A

Diagrams

This appendix reports the diagrams of the realized systems. Figure A.1 displays the class diagram of the implemented classes using the classical UML convention. Figure A.2 display the logical flow of the of sample acquisition, network training and online classification. Figures A.3 and A.4 displays the logical flow of the EMG board module and of the feature extraction function.

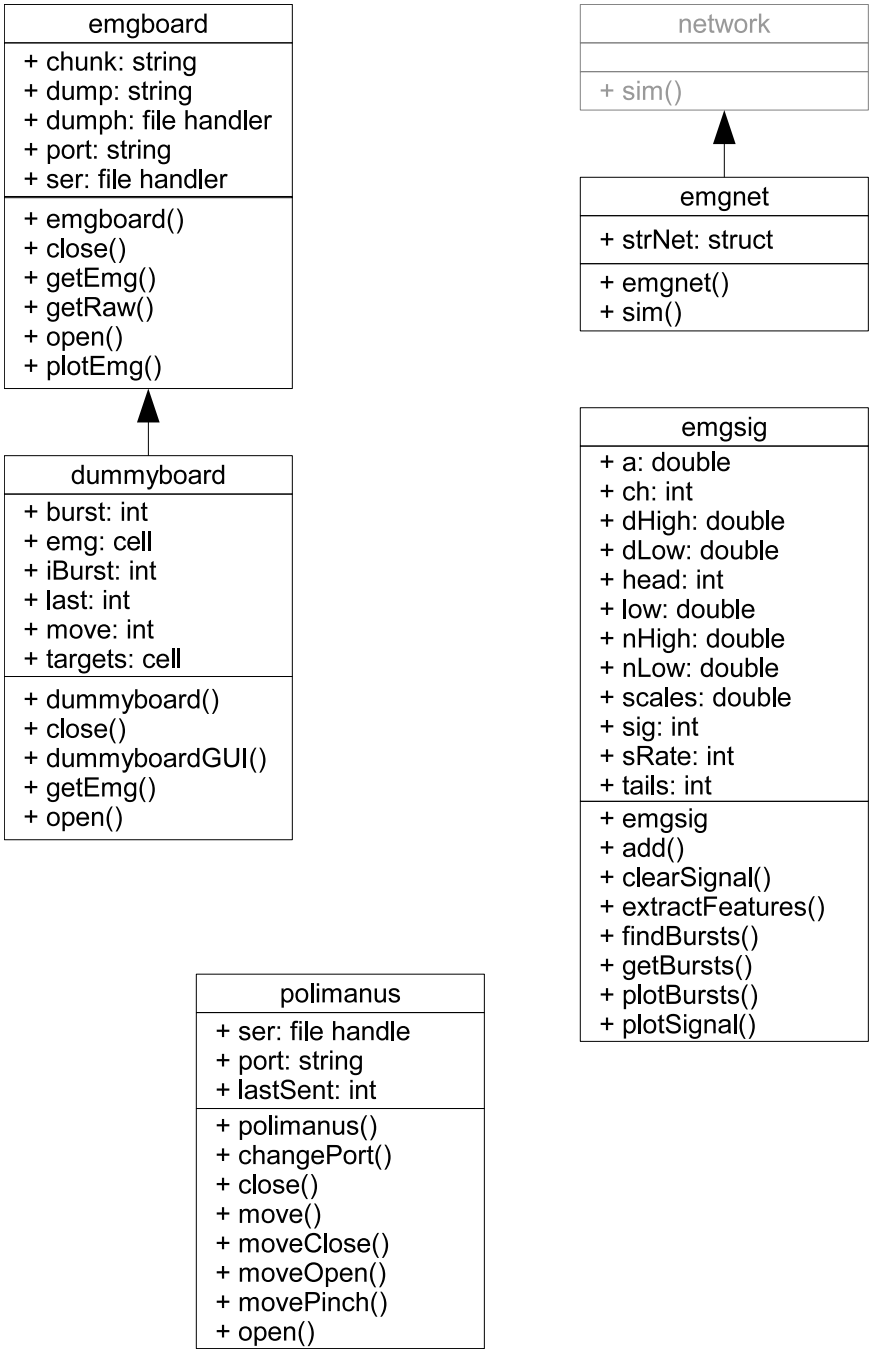


Figure A.1: Diagram of the classes

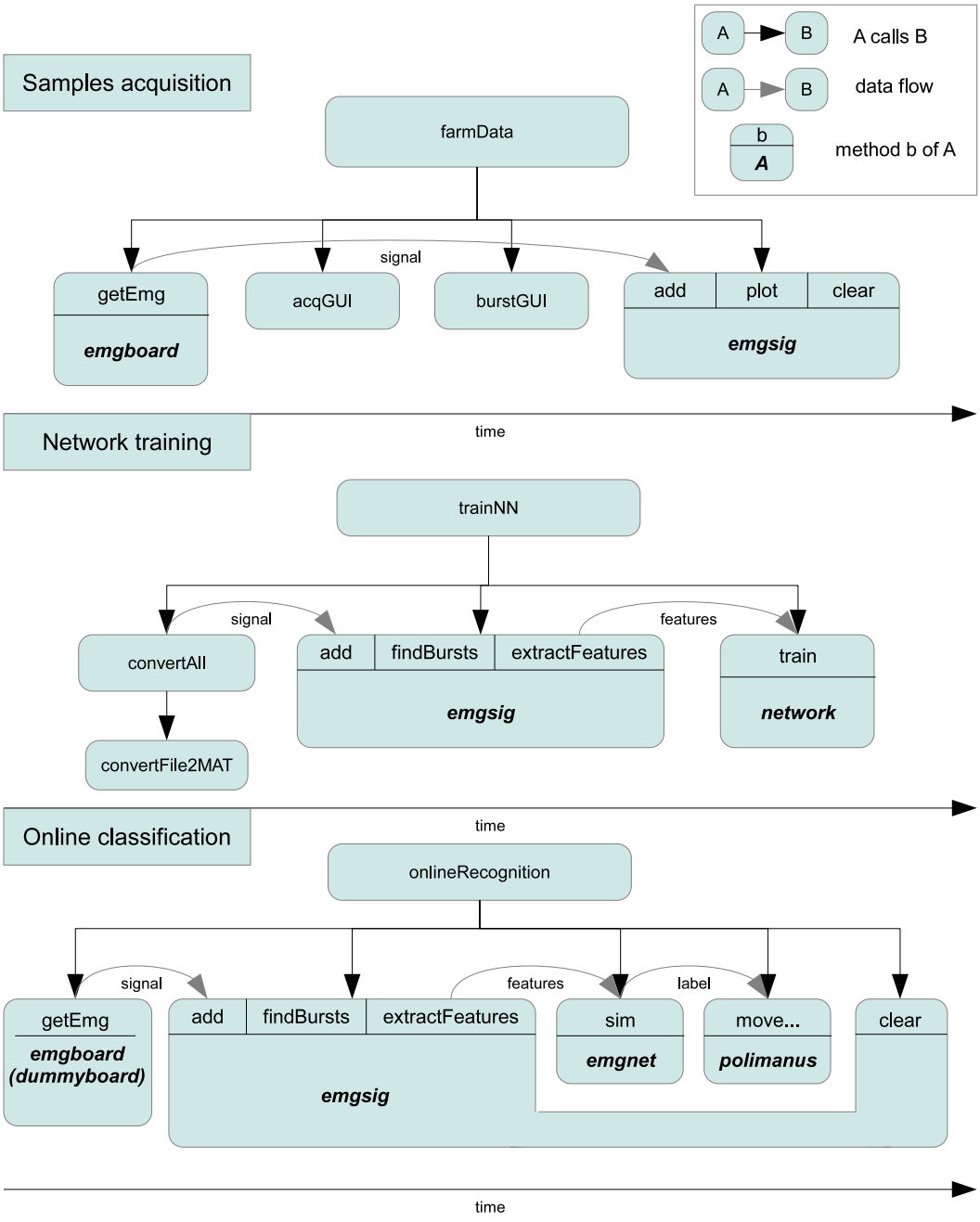


Figure A.2: Diagram of the logical flow of the functions in different stages

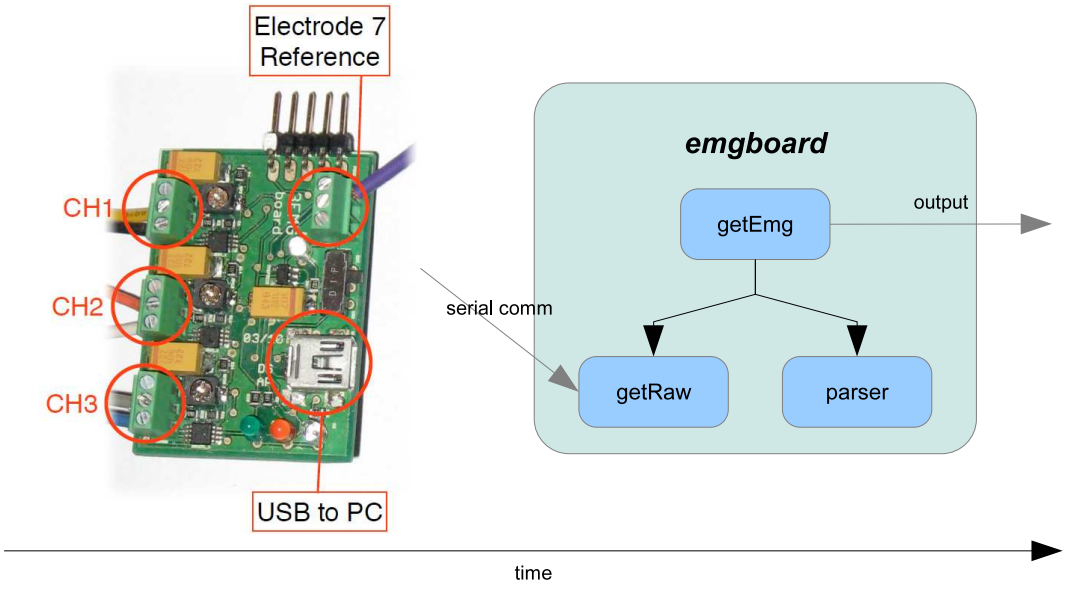


Figure A.3: Diagram of emgboard

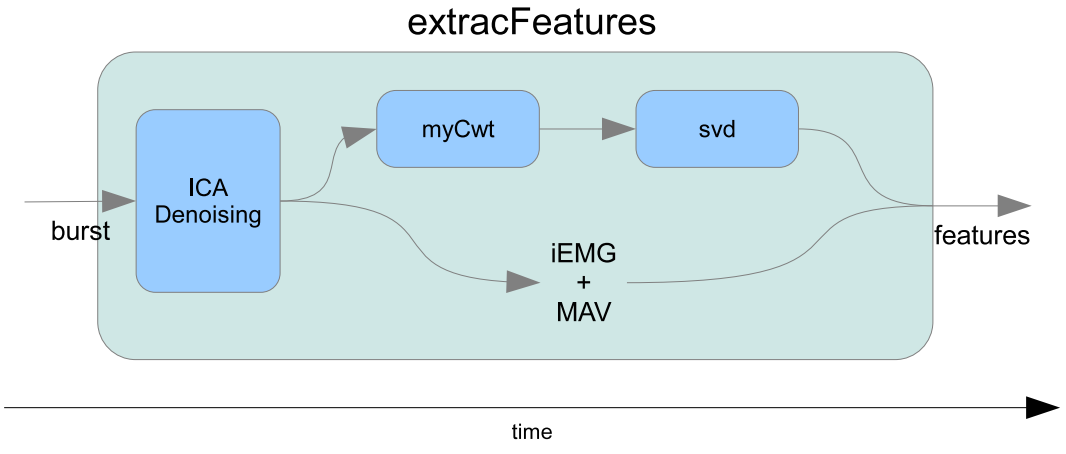


Figure A.4: Diagram of feature extraction

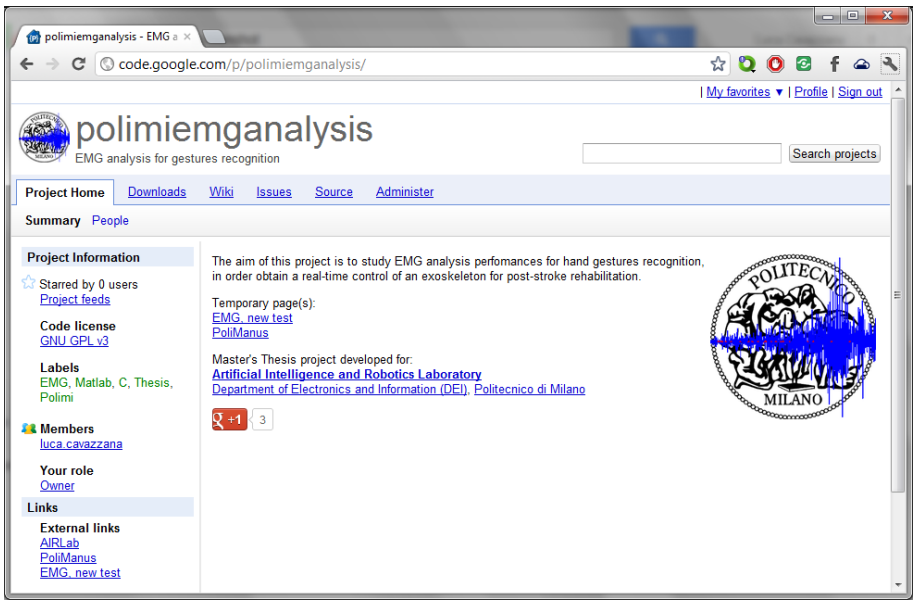
Appendix B

Software documentation

This appendix contains the documentation of the software realized within this project.

For each class is provided an high level description of his functionalities, followed by a list of the properties and the methods with a brief explanation. For each function or method a detailed description is provided, following the style of the official Matlab Documentation. These descriptions can also be accessed through Matlab's built-in `help` and `doc`.

The code is available on a online repository at the address `http://code.google.com/p/polimiemganalysis/`



B.1 Emgnet

Pattern recognition neural network.

This class extends the built-in network, storing the struct converted class in order to avoid the costly conversion every time the `sim` function is called. This way the time required to produce an output is significantly reduced for small networks.

Class Details

Superclasses: network

Constructor Summary

emgnet create a custom emg classifier

Property Summary

strNet network struct

Method Summary

sim simulate emgnet

Method Details

emgnet

Class constructor.

`en = emgnet(net)` receives a network class `net` and converts it in a `emgnet` object.

sim

Simulates the neural network.

This method overrides the original `sim` method. It skips the conversion to struct of the `network` class (since was already performed by the constructor) thus significantly improving execution time.

`y = sim(en,x)` takes the emgnet object `en` and the input vectors `x` and returns the output vectors `y` generated by the network. For more details see `sim`.

B.2 Emgboard

EMG board interface for signal acquisition.

`emgboard` handles the serial communication with the Eracle device, offering a method which pools the board and parses the output, thus returning the signal as a matrix of integers.

Class Details

Superclasses: handle

Constructor Summary

	emgboard	create Eracle interface

Property Summary

	chunk	incomplete sample from last acquisition
	dump	dump filename
	dumpH	dump file handler
	port	port name
Constant	**sRate**	serial sample rate
	ser	serial handler

Method Summary

	close	close serial communication
	getEmg	get parsed EMG signal from Eracle
	getRaw	get raw data output from the serial port
	open	open serial port communication
Static	**parser**	parse EMG board output
	plotEmg	real-time signal plot

Method Details

emgboard

Class constructor.

`eb = emgboard(port,dump)` returns an object which will handle the Eracle board. `port` is the serial port name, while `dump` (optional) is the name of the file where the raw data is saved.

close

Close serial communication.

`status = close()` closes the serial port (and any dump file). Returns 1 on success.

getEmg

Get parsed EMG signal from Eracle.

`[ch data] = getEmg(w)` gets and parses the signal from the EMG board and returns the signal `ch` as $N \times 3$ matrix, where N is the signal length and 3 the number of channels. If `w` is given and not zero the call will be blocking. If specified, `data` is the raw output from the serial.

NOTE: too much time between two consecutive serial reading (ie: long analysis time) could cause the input buffer to fill, thus losing data. This way the parser could be unable to concatenate and parse the two readings.

getRaw

Get raw data input from serial port.

`raw = eb.getRaw(w)` returns the raw output string from the serial board. If `w` is specified and not zero this call will be blocking.

open

Opens serial port communication.

`status = eb.open()` opens serial port communication, returning 1 on success.

parser

Parses EMG board output. `[Static]`

`[ch,chunk] = emgboard.parser(raw,chunk)` parses the EMG board output `raw`, concatenating it with `chunk` if provided. Returns the $N \times C$ matrix `ch` (with N number of samples, C number of channels) and the tail of the last incomplete sample `chunk`.

plotEmg

Real-time signal plot.

`f = eb.plotEmg(f)` plots the signal from the EMG board. It takes as parameter (optional) and returns the handler `f` of the window where the signal is drawn. The chunk from the latest acquisition is plotted in red.

B.3 Dummyboard

EMG board emulator.

`dummyboard` This object will simulate the output of an EMG board: samples from a pre-recorded burst are returned when a gesture is selected through the GUI, and random noise is returned otherwise to simulate muscle's inactive state. The sample rate is the same of the original Eracle board.

Class Details

Superclasses: emgboard

Constructor Summary

dummyboard	create EMG board emulator

Property Summary

burst	index of the selected burst
emg	pre-recorded bursts
iBurst	index of the last outputted sample
last	timestamp last acquisition
move	id of the selected movement
targets	gesture associated to the pre-recorded bursts

Method Summary

dummyboardGUI	gesture selection interface
getEmg	get simulated EMG data
open	load pre-recorded bursts

Method Details

dummyboard

Class constructor.

`db = dummyboard(src)` returns an object which simulates an Eracle board. `src` is the path of the .mat file containing pre-recorded bursts.

dummyboardGUI

Gesture selection interface.

`db.dummyboardGUI()` opens a interactive user interface for gesture selection.

getEmg

Get simulated EMG data.

`ch = db.getEmg()` returns a $N \times 3$ matrix `ch` containing the generated samples, where 3 is the number of channels and N is the number of samples, accordingly to the sample rate and the elapsed time from the last method call. The output consists into samples of a pre-recorded burst when a gesture is selected through the GUI, random noise otherwise.

open

Load pre-recorded bursts.

`db.open()` loads the pre-recorded bursts from the file specified during initialization.

B.4 Emgsig

EMG signal analysis.

This class encapsulates the EMG signal and provides the methods for burst detection and feature extraction

Class Details

Superclasses: handle

Constructor Summary

emgsig	EMG signal analysis

Property Summary

a	ica weights
ch	detected bursts' dominant channel
dHigh	highpass poles
dLow	lowpass poles
heads	detected bursts' head
low	lowpass signal
nHigh	highpass zeros
nLow	lowpass zeros
sRate	sampling frequency
scales	wavelet scales
sig	signal samples
tails	detected bursts' tail
xWAV	integration limits (used by myCwt)
yWAV	integral wavelet (used by myCwt)

Method Summary

add	add new signal samples
clearSignal	flushes old signal samples
extractFeatures	extract bursts' features
findBursts	detect muscular activity
getBursts	returns EMG samples
plotSignal	plots EMG signal
setSignal	set EMG signal

Method Details

emgsig

Class constructor.

`es = emgsig(rate)` returns a `emgsig` object used for feature extraction. The parameter `rate` is the sample rate of the EMG board.

add

Add new signal samples.

`len = es.add(ch)` appends the new signal chunk `ch` to `es.sig`. Returns the final signal length `len`.

clearSignal

Flushes old signal samples.

`feats = es.clearSignal()` removes the signal samples no longer useful for online recognition.

extractFeatures

Extract bursts' features.

`feats = es.extractFeatures(varargin)` returns a cell-array where each elements contains the features of the bursts detected by emgsig.findBursts. If verb—'ica'— is specified among the optional values ICA denoising is preformed before signal analysis.

findBursts

Detect muscular activity.

`n = es.findBursts()` analyses the EMG activity to detect burst. Returns the number of bursts found.

getBursts

Returns EMG samples.

`bursts = es.getBursts()` returns a cell-array where each elements contains the samples of the bursts detected by emgsig.findBursts.

plotSignal

Plots EMG signal.

`f = es.plotSignal(f)` plots the signal, highlighting the bursts detected with emgsig.findBursts. If provided, `f` is the handler of the window where the method will plot.

setSignal

Set EMG signal.

`len = es.setSignal(sig)` replaces the EMG signal with the one provided with `sig`. Returns signal's length `len`.

B.5 Polimanus

Polimanus interface.

The class `polimanus` handles serial communication with Polimanus exoskeleton, providing high-level methods to control it. Drivers from FTDI site may be needed.

Class Details

Superclasses: handle

Constructor Summary

polimanus	create Polimanus interface

Property Summary

lastSent	last sent position
port	port name
ser	serial port handler

Method Summary

changePort	changes polimanus port
close	close serial port
move	perform generic movement
moveClose	perform close hand movement
moveOpen	perform open hand movement
movePinch	perform precision grasp
open	open serial port

Method Details

polimanus

Class constructor.

`pm = polimanus(port)` creates an handler for the Polimanus exoskeleton on port `port`.

changePort

Changes the port of Polimanus.

`pm.changePort(port)` sets the new port of Polimanus to `port`.

close

Close serial communication.

`pm.close()` closes the serial port communication.

move

Perform generic movement.

`pm.move(p1,p2,s1,s2)` commands a generic movement, moving servo1 to `p1*180/256` degree and servo2 to `p2*180/256` degree with speeds `s1` and `s2`. All parameters are values within 0 and 255.

moveClose

Perform close hand movement.

`pm.moveClose(s)` commands hand closing with speed `s` (within 0 an 255).

moveOpen

Perform open hand movement.

`pm.moveOpen(s)` commands hand opening with speed `s` (within 0 an 255).

movePinch

Perform precision grasp movement.

`pm.movePinch(s)` commands pinch movement with speed `s` (within 0 an 255).

open

Open serial communication.

`status = pm.open(varargin)` opens serial port communication, returning success status. If the string `log` is within the optional arguments the serial communication is dumped on disk.

B.6 Functions

B.6.1 Analysis

extractFeatures

Extract features from a signal segment.

`f = extractFeatures(sig, scales, yWAV, xWAV)` returns the feature vector `f` (integral EMG, absolute mean value, wavelet coefficients after SVD) of `sig`. `yWAV` and `xWAV` are the values of the selected mother wavelet obtained with `intwave`, `scales` are the scaling coefficients.

ica

Performs ICA denoising.
`[s, a] = ica(emg, aStart, aOnly)` performs ica denoising, extracting the independent sources and reconstructing the original signal after a dynamical thresholding of the weights. It takes as input the $L \times 3$ signal `sig`, where L is the length of the signal, and the (optional) mixing matrix `aStart` as initial value. Returns the denoised signal `s` and the computed matrix `a`. If `aOnly` is provided and equal to 1 `s` is not computed and only `a` is returned.

myCwt

Real or Complex Continuous 1-D wavelet coefficients.

This is a customization of the original continuous wavelet transform function.
`[coefs,varargout] = myCwt(sig,scales,yWAV,xWAV,plotmode,xlim)` receives as input the signal `sig`, the scales `scales` and the already built mother wavelet (using intwave) through the parameters `yWAV` and `xWAV` (where `[yWAV,xWAV] = intwave('wname')`). Since the function does not have to call `intwave`, execution time is significantly reduced. For more details see cwt.

B.6.2 Training

farmData

Acquires training sets.

`farmData(port)` guides the patient through the acquisition of sample movements, saving the data in a subfolder of the current path, plus a gest.mat file containing gestures name, ID and #repetitions. If specified opens the board on `port` (otherwise the default port is used).

trainNN

Trains pattern recognitino nn.

`[nets, trs] = trainNN(folder, nnn, burstRatio, varargin)` loads the samples stored into `folder` and trains a number of neural network equal to `nnn` using only the initial segment of the detected burst whose length is expressed as percentage over the full burst (default: 100%). if 'ica' is specified as optional parameter ICA denoising is performed before training.
Returns the trained `nets` and their training records `trs`.

B.6.3 Online recognition

onlineRecognition

Performs online recognition.

`onlineRecognition(net, varargin)` performs online gesture recognition. It takes as input the network `net` (`network` or `emgnet` class) and uses it to continuously classify the gestures acquired from the EMG board. If `'ica'` is provided among the optional parameters ICA denoising is performed during the analysis. `'plot'` enables the graphical feedback (otherwise only the textual one is provided).

B.6.4 GUIs

acqGUI

Get gesture names.

`[n, nrep, gests, name] = acqGUI()` displays a graphical interface where the user can select the number of gestures `n`, the number of repetitions `nrep`, their name `gests` and the `name` of the patient.

burstGUI

Acquisition walkthrough GUI
`port = burstGUI(gName,rep)` launches a graphical interface which allows the user to signal to the calling function the beginning and the stop of the movement. Displays the gesture `gName` and repetition `rep`.

portGUI()

Returns the selected serial port.

`port = portGUI()` launches a graphical interface for serial port selection. Returns a string containing the name of the selected port.

B.6.5 Utilities

convertAll

Gets datas from folder.

`c = convertAll(np)` returns a cell-array containing the signals parsed from the files in `folder`.

convertFile2MAT

Extrat emg data from file.

`sig = convertFile2MAT(f)` returns the data vector `sig` extracted from the text file `f`.

parseRaw

Convert raw data file.

`ch = parseRaw(folder)` opens the files stored into `folder`, parses the contained raw signal (as taken from the EMG board) and separates the channels into separate files.

plotAll

Plot all acquisitions.
`plotAll(folder, findB)` plots all acquisitions stored in the files in `folder`. If `findburst` exists and not null the signal is segmented too.

plotEmgFile

Plots EMG data from file.

`f = plotEmgFile(patient, seq, gesID, gesName)` plots EMG signal saved in files folder/ch#/gesID-seq-gesName.txt and returns figure hadle f.

remakeRaw

Rebuilds raw input from parsed channels.

`remakeRaw(folder, file)` rebuilds the raw input file (as outputted from the EMG board) from the samples stored into ./patient/ch#/file

saveBursts

Save raw bursts or features on disk.

`saveBursts(type, folder)` extracts bursts from the files into folder/ch#, performs segmentation and saves the raw data (`type = 'raw'`) or their features (`type = 'feats'`) into folder_type.mat. The file will contain a $2 \times N$ cell matrix, on the first row the signals/features, on the second one the gesture ID.

Appendix C

Wavelet analysis

Principles of continuous wavelet transform

Many signals, such as the EMG, have a non-stationary nature, varying in amplitude and frequency features over time. The classical approach based on classical Fourier analysis consists into breaking the signal into sine waves of different frequencies, producing a spectral representation unable to catch the transient features and losing their temporal localization, in a result which offers poor information.
An alternative technique developed in 1946 is the short-time Fourier transform (STFT), which is obtained applying the FT over small intervals windowing the signal.

$$F(\omega,\tau) = \int f(t)\, w(t-\tau)\, e^{-j\omega t} dt \qquad \text{(C.1)}$$
$$w(t) = \sqrt{\frac{\alpha}{\pi}} e^{-\alpha(t-\tau)^2}$$

This way can we map $f(t)$ over ω and τ, providing a bidimensional representation which gives us both spectral and temporal localization of the signal's features. The drawback is the fixed size of the window forces the same resolution for all the time-frequency plane, while some applications may require a more flexible approach which allows to determine more accurately either time (shorter windows) or frequency (wider windows).

Continuous wavelet analysis is a technique developed in the early 1980s, rapidly developed thanks to his flexibility and the wide range of application, replacing the Fourier transform in many fields.
This technique analyses the signal measuring the similarity (by means of

inner product) between the signal and a function Ψ called *mother wavelet* (while in FT the analysing functions are complex exponentials and in STFT are windowed complex exponentials) scaled and shifted by different values, thus obtaining a two-dimensional representation (time-scale) of a one-dimensional signal.

$$C(a,b;f(t),\psi(t)) = \int_{-\infty}^{\infty} f(t)\,\frac{1}{\sqrt{a}}\psi^*\left(\frac{t-b}{a}\right)dt \tag{C.2}$$

Frequency features are closely related to the scale of the wavelet: as the coefficient a decrease the wavelet is compressed, thus showing more rapidly changing features which will match high-frequency dynamics of the analysed function leading to bigger values in the high-scale coefficients of the transform. On the contrary higher scales will stretch the wavelet, matching low frequency features. As we can note from equation C.2 the wavelet is also multiplied by $1/\sqrt{a}$, in order to obtain to have an analysing signal with the same energy at every scale.

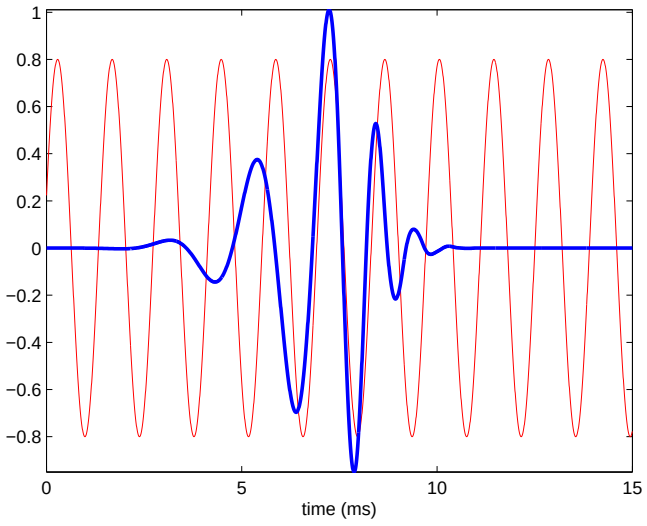
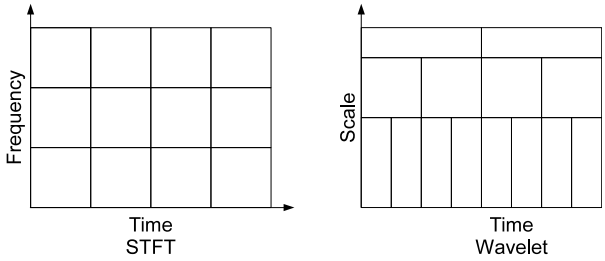


Figure C.1: Example of a 'db8' wavelet over a sinusoidal signal

Shifting is obtained delaying the analysing function. Since the wavelet has finite energy this will naturally implement signal windowing, allowing us to have a temporal localization of the features. Moreover scaling allows us to modify the size of the window stretching or compressing the wavelet,

giving us a flexible tools which allows a finer temporal localization of high-frequency features (low scale) and a more precise frequency localization of slowly changing dynamics (high scale). This is one of the most interesting advantages of wavelets with respect to STFT, where the resolution if fixed.



Another important element in wavelet analysis is the shape of the mother wavelet. There are different wavelet families, each one with different features which make it more or less suitable give the nature of the signal and the available computational power: a function with rapidly changing features will be more suited to look discontinuities in the signal, while if you are interested in smooth oscillations a wavelet closely matching this behaviour may be appropriate. As a general rule the more a wavelet resembles the signal the more effective the analysis.

Discrete wavelet transform

To reduce the amount of computation required by CWT a new method was conceived based on performing the analysis only on dyadic positions and scales, resulting in a much more efficient transform. This technique, called *discrete wavelet transform* consist into applying two complementary filter to the analysed signal, thus generating two different outputs: the first one will contain the low frequency features (also called *detail*), while the second one will contain the high frequencies (*approximation*).

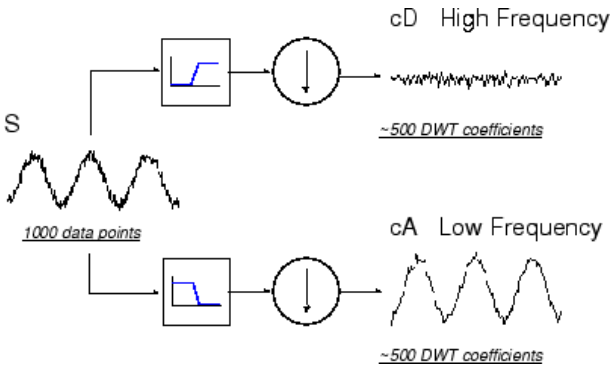


Figure C.2

To reduce the number samples, that otherwise will grow slowing the computation, the signal is then downsampled. This process is then iterated repeating the analysis on the approximation signal until downsampling reduces it to a length that make impossible further iterations.

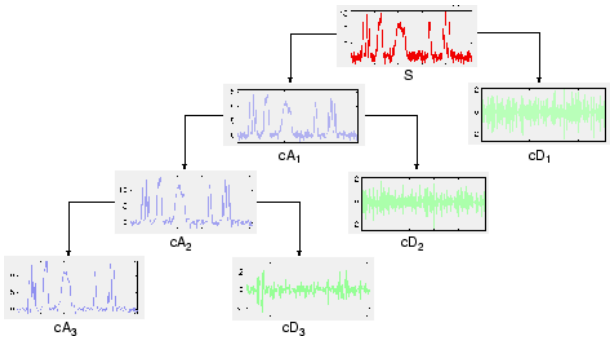


Figure C.3

The advantage of DWT, if compared to CWT, is that halving the number of samples after each iteration the complexity of the transform reduces significantly while maintaining a similar analysis capabilities, making it more suitable for applications such as signal (or image) analysis, denoising and compression.