University of Verona

DEPARTMENT OF COMPUTER SCIENCE
Ph.D. Course in Computer Science - Cycle XXXI/2015

Doctoral School of Natural Sciences and Engineering

Myoelectric Control Architectures to Drive Upper Limb Exoskeletons

S.S.D. ING-INF/05

Candidate:
Davide Costanzi
VR397872

Supervisors:
Andrea Calanca, PhD

Prof. Paolo Fiorini, PhD

October 2021

Abstract

Myoelectric interfaces are sensing devices based on electromyography (EMG) able to read the electrical activity of motoneurons and muscles. These interfaces can be used to infer movement volition and to control assistive devices. Currently, these interfaces are widely used to control robotic prostheses for amputees, but their use could be beneficial even for people suffering from motor disabilities where the peripheral nervous system is intact and the impairment is only due to the muscles, e.g. muscular dystrophy, myopathies, or ageing. In combination with recent robotic orthoses and exoskeletons, myoelectric interfaces could dramatically improve these patients' quality of life. Unfortunately, despite a wide plethora of methodologies has been proposed so far, a natural, intuitive, and reliable interface able to follow impaired subjects' volition is still missing. The first contribution of this work is to provide a review of existing approaches. In this work we found that existing EMG-based control interfaces can be viewed as specific cases of a generic *myoelectric control architecture* composed by three distinct functional modules: a *decoder* to extract the movement intention from EMG signals, a *controller* to accomplish the desired motion through an actual command given to the actuators, and an *adapter* to connect them. The latter is responsible for translating the signal from decoder's output to controller's input domain and for modulating the level of provided assistance. We used this concept to analyse the case of study of linear regression decoders and an elbow exoskeleton. This thesis has the scientific objective to determine how these modules affect performance of EMG-driven exoskeletons and wearer's fatigue. To experimentally test and compare myoelectric interfaces this work proposes: (1) a procedure to automatically tune the decoder module in order to equally compare or to normalize the decoder output among different sessions and subjects; (2) a procedure to automatically tune gravity compensation even for subjects suffering from severe disabilities, allowing them to perform the experimental tests; (3) a methodology to guide the impaired patients through the experimental session; (4) an evaluation procedure and metrics allowing statistically significant and unbiased comparison of different myoelectric interfaces. A further contribution of this work is the design of an experimental test bed composed by an elbow exoskeleton and by a software framework able to collect EMG signals and make them available to the exoskeleton's actuators with minimal latency. Using this test bed, we were able to test different myoelectric interfaces based on our architecture, with different modules choices and tunings. We used linear regression decoders *calibrated* to predict the muscular torque, low-level controllers having torque or velocity as reference, and adapters consisting of a properly dimensioned gain or simple dynamic systems, such as an integrator or a mass-damping system. The results we obtained allow to conclude that EMG-based control is a viable technology to assist muscular weakness patients. Moreover, all the components of the *myoelectric control architecture* – decoder, adapter, controller, and their tuning – significantly affect the task-based performance measures we collect. Further investigations should be devoted to a methodology to automatically tune all the components, not the decoders only, and to the quantitative study of the effect the adapter has on the regulation of the assistance level and of the tradeoff between speed and accuracy.

Sommario

Le interfacce mioelettriche sono sensori basati sull'elettromiografia (EMG) che leggono l'attività elettrica di motoneuroni e muscoli; possono essere usate per inferire la volontà di movimento e per controllare un dispositivo di assistenza. Attualmente, sono ampiamente utilizzate per controllare protesi robotiche per amputati, ma del loro uso potrebbero beneficiare anche persone il cui sistema nervoso periferico rimane intatto e la disabilità è dovuta soltanto all'apparato muscolare, ad esempio per distofia muscolare, miopatie o invecchiamento. In combinazione con le recenti ortesi robotiche ed esoscheletri, le interfacce mioelettriche potrebbero migliorare notevolmente la qualità di vita di questi pazienti. Sfortunatamente, a dispetto del gran numero di metodologie proposte finora, manca ancora un'interfaccia naturale, intuitiva ed affidabile capace di seguire la volontà dei soggetti disabili. Il primo contributo di questo lavoro è fornire una revisione degli approcci esistenti. Nel fare questo, abbiamo scoperto che le interfacce basate su EMG esistenti possono essere viste come casi specifici di una generica *architettura per il controllo mioelettrico* (myoelectric control architecture) composta da tre moduli: un *decoder* per estrarre l'intenzione di movimento dai segnali EMG, un *controllore* (controller) per eseguire il movimento desiderato attraverso un effettivo comando dato agli attuatori ed un *adattatore* (adapter) per collegarli. Quest'ultimo è responsabile di tradurre il segnale da uscita del decoder a ingresso del controllore, modulando il livello di assistenza. Abbiamo utilizzato questo concetto per analizzare il caso di decoder basati su regressione lineare e un esoscheletro per il gomito. Questa tesi ha l'obiettivo scientifico di determinare come questi moduli influenzano le prestazioni di un esoscheletro guidato da EMG e l'affaticamento del soggetto. Per testare e confrontare sperimentalmente le interfacce questo lavoro propone: (1) una procedura per regolare automaticamente il decoder per garantire un confronto equo e normalizzarne l'output tra sessioni e soggetti diversi; (2) una procedura per regolare automaticamente la compensazione di gravità anche in soggetti affetti da severe disabilità, permettendo loro di portare a termine i test sperimentali; (3) un metodo per guidare i pazienti disabili durante la sessione sperimentale; (4) una procedura di valutazione e metriche di prestazione che permettono un confronto statisticamente significativo e senza distorsioni tra differenti interfacce. Un ulteriore contributo è la progettazione di un sistema sperimentale composto da un esoscheletro per il gomito e un ambiente software in grado di acquisire segnali EMG e renderli disponibili agli attuatori dell'esoscheletro con latenza minima. Usando questo sistema, siamo riusciti a testare differenti interfacce basate sulla nostra architettura, con differenti scelte di moduli e regolazioni. Abbiamo usato decoder *calibrati* per predire la coppia muscolare, controllori a basso livello con riferimenti di coppia o di velocità ed adattatori che consistono in un guadagno adeguatamente dimensionato oppure semplici sistemi dinamici, come un integratore o un sistema massa-smorzatore. I risultati ottenuti ci permettono di concludere che il controllo basato su EMG può essere utilizzato per assistere persone affette da debolezza muscolare. Inoltre, tutti i componenti dell'*architettura per il controllo mioelettrico* influenzano significativamente le misure di prestazione basate su task che raccogliamo. Ulteriori ricerche dovrebbero essere indirizzate ad una metodologia per regolare automaticamente tutti i componenti, non solo i decoder, e allo studio quantitativo dell'effetto che l'adattatore ha sulla regolazione del livello di assistenza e del compromesso tra velocità e accuratezza.

Contents

CHAPTER 1

Introduction

The need for assistance to disabled and elderly people is going to increase in the next future: main motivations are the growing number of people suffering from long term or non-recoverable physical impairments (world's population affected by some form of disability increased from around 10% in the 1970s to more then a billion people or about 15% in 2010 [67]), the demographic shift towards an older population that has a higher risk of disability (at present over 65 are 20% of world's population and it is expected to be at least 35% by 2050 [13]), and the shortage of therapists and caregivers assisting disabled people [42]. As robotic devices may help to face all these challenges, great efforts have been put in the development of wearable robotic devices such as exoskeletons and active orthoses [22, 27, 38, 42, 64]. This technology has applications ranging from rehabilitation to assistance and may help to increase life independence of disabled people while reducing the burden for caregivers and the society [13, 27, 54, 70]. In the last decades, research has mainly focused on:

- mechanical design in order to achieve suitable support for assisting forces and kinematic compatibility;
- low-level control design to provide safe and high performance physical interaction with impaired subjects.

Most popular applications include lower limbs rehabilitation and assistive devices and upper limbs rehabilitation devices for people with neurologic diseases [52].Fewer efforts have been spent on assistive devices for people suffering from long-term non-recoverable neuromuscular diseases affecting the upper limbs. This population needs assistive devices able to infer motion volition and restore arm's capabilities. In fact, subjects who have lost upper limb functionalities hope that the advancement of wearable robots could give them the capacity of regaining or acquiring *autonomous* control over their own arms [70]. This technology would lead to enhance autonomy on activities of daily living (ADLs) and thus to dramatically improve the patient's quality of life. In many cases, technology is the only prospective to improve their every day condition, since a biological cure is still a dream for many pathologies.

One of the key components of such assistive technology is represented by myoelectric interfaces (MIs).

Most of myoelectric interfaces use surface electromyography (sEMG) to non-invasively record the signals generated by the myoelectric activity, which is related to the voluntary activation of motoneurons and muscles.The informative content of the recorded signals needs to be properly *decoded* in order to control a robotic device. The *decoding problem* is indeed the core issue of working with sEMG. It deals with converting the collected myoelectric signals into a quantitative representation of movement volition.

Across years, EMG decoding has exploited single or multi-channel electrodes and has integrated information from other sensors, e.g. encoders, accelerometers, gyroscopes and load cells [40, 46]. Machine learning and neuromusculoskeletal (NMS) models have been used to produce more clear and physiological signals [24, 32, 58, 60]. Despite these efforts, most of existing solutions are affected by some issues, making them unreliable or inadequate, at least in some conditions [9]. For instance:

- machine learning techniques usually need long training sessions for re-tuning each time the exoskeleton is worn.
- existing EMG classifiers [9, 19, 46] are not adequate to achieve continuous voluntary control, as are they are typically used to distinguish among a discrete set of desired actions.
- reliable strategies able to online adapt decoding and control strategies depending on the patient condition or the intended task do not exist.
- few approaches have been evaluated on impaired subjects [9, 19, 24, 40, 46], although many of them have been proposed to decode EMG signals for exoskeleton control.

In conclusion, we do highlight that most existing results are just experimental, without any attempt to model the detailed underlying working mechanisms of existing EMG-control architectures [14, 61]. Conversely, existing NMS models could explain these mechanisms but are quite difficult to be trained on a large set of subjects and, due to their complexity, they cannot be expressed in treatable analytical forms [58, 60].

These and other issues, as detailed in section 1.1, have impeded the emerging of a true voluntary control paradigm where the patient can move an impaired arm in a *natural*, *intuitive* and *reliable* way, allowing *simultaneous* and *proportional* control of multiple DoFs, in everyday use [9, 40, 46]. While all these challenges hold for robotic exoskeletons, robotic prostheses are mature commercial products. The reason behind the greater technological readiness of prostheses is that they do not act in parallel with the human body, so they do not strictly need coherence with physiological activations [1].

For this reason the main focus of this review is, throughout the vast literature on myoelectric interfaces, on MIs for exoskeleton control. Three fundamental aspects can be identified:

Interface architecture describes the way sensors and actuators are interconnected in order to apply the desired command.

Interface training describes how the sensor data are interpreted and how the related models are trained.

Interface evaluation defines the metrics for assessing the interface performance.

Logically, the **interface architecture** is a myoelectric control architecture composed by a *decoder* module devoted to the decoding problem, i.e. compute a signal which value is controlled by wearer's volition starting from human interface sensors, a *controller* module defining how to use this reference signal in a low-level control algorithm, and an in-between *adapter* module interpreting the decoded EMG signal and modulating the assistance level.

The **training** modality defines which data the models require to calibrate their parameters and addresses how these data can be collected from the target population.

The **evaluation** metrics for the interface should be related to the system functional performance and should quantify the interface capabilities in the field. These should be *task-based*, measuring, for example, the interface performance in activities of daily living, or its ability in restoring lost abilities and how these compares to healthy people. Measures related to decoding performance, such as model fit error or classification accuracy, are not so important, since they are not measurable *online* – so ignore the active interaction with the wearer [47] – and they cannot assess the interface performance as a whole [66].

Throughout this work we aim to factor in all these aspects having a comprehensive view of the interface, considering not only to the definition of a decoder but also the solution of the collateral issues that prevent the realization of a volitional, myoelectric control interface, usable by muscle weak people. We include practicalities too often disregarded in the literature, but fundamental when working with disabled people. Moreover, we should never forget that these systems has the human in-the-loop and therefore the myoelectric interface is an *human-embedded controller* [1], where the human CNS is an active and adapting component.

In the contest of this thesis, we do not consider multimodal interfaces combining other sensors in addition to sEMG, or that would require a multi-joint robot to be tested. We limited our investigations to interfaces that solely use on sEMG sensors and controlling a single joint, to showcase a more rigorous path for the design of myoelectric interfaces for people suffering from neuro-muscular disorders with a specific focus on muscular weakness pathologies, such as muscular dystrophy, myopathies, or aging.

1.1 Myoelectric interfaces issues

From the user's point of view, an ideal control interface for an assistive device should be:

Natural device need to follow the wearer's volition, with minimal effort;

Intuitive responses of the device in consequence of wearer's inputs are easily predictable;

Reliable behaviour of the device should be consistent during and across use session, it never counteract wearer's intention and experience can only improve overall performance.

Unfortunately, existing control interfaces are inadequate to satisfy these needs, because they are unreliable, cumbersome, counterintuitive and have low information throughput. Myoelectric interfaces also suffer of these issues: clinical applications have revealed that sEMG signals are unreliable, badly conditioned, typically affected by high noise, crosstalk phenomena, sensible to sensor placement, and dependent on time, fatigue, and sweat [9, 41]. Thus, EMG-based control is often the result of several compromises. Notwithstanding the challenges posed by using sEMG, it is the best non-invasive source of biosignals with clearer and richer information content with respect to the ones provided, e.g., by brain–computer interfaces based upon electroencephalography (EEG) [46]. Moreover, no valid alternatives are used in clinics to access PNS signals.

The consequence of an inadequate interface is the rejection of the device [70]: considered that there is an implied overhead to use a wearable robot (e.g. donning and calibration procedures, encumbrance, limitations, etc. . .), the device should provide an advantage in term of time and precision over getting help from a caregiver. Without these requirements of accuracy and velocity, the patient would always prefer not to use the device.

The research on myoelectric interfaces too often disregarded some aspects that actually matter for the application in a real case scenario:

- Most of the interfaces have never been tested on target population, but only on healthy subjects, therefore the adopted training procedures often cannot work with target population, since they require the execution of movements these people are not able to do autonomously [11]. Or, we do not know if the interface can be adapted to disabled people and how, as for example the interfaces based on muscle models [58] that are known only for healthy muscles [68] thus excluding muscular weak subjects.

- There is a lacking of metrics able to accurately measure the "quality" of the interface, i.e. how close it follows movement intention. That is because we do not have access to the actual intention, especially in target population, thus the feeling of the interface is very subjective and depends on the task and the device we use. However, it is fundamental to objectively assess the control interface. Moreover, these metrics should be measured online, because only online testing can display the closed loop interaction between device and human that is where the behavior of the interface arises and concurrent effects due to the other components of the system can be taken into account. It is also important to do these online tests at an early development stage to guide further refinements [9], since high offline performances can possibly not be as high as online ones and vice versa some offline inaccuracies of the decoding can be compensated by the closed loop with human [11, 47, 61].

- In our opinion the myoelectric interfaces for orthoses are more difficult to develop w.r.t. the ones for prostheses. While there are studies showing that to control prosthesis or external robotic arms we can use every fixed mapping of muscle activations to DoFs and the human can adapt [1, 29], for an exoskeleton, that is mechanically linked to the limb, we require more accuracy since residual and robot movements have to be consistent.

These considerations lead to the following requirements for the interface:

- The train methodology should be usable also within the target population.

- The interface should be assessed with online tests and quantitative metrics that measure its performances as part of a complete system.

- There should be an exact matching between the residual movement and the decoded movement.

CHAPTER 2

State of the art – part 1

Wearable robots, control interfaces and EMG

This chapter briefly reviews the literature about wearable robots, with focus on upper-limb assistive exoskeletons (section 2.1) and proposed control interfaces and sensor fusion approaches, sEMG-based but alternatives too (section 2.2). Section 2.3 will introduce the biophysiology that lays behind sEMG signal generation.

2.1 Introduction to wearable robots

This section briefly reviews the literature about wearable robots, with focus on upper-limb devices. From the structural point of view, wearable robots can be divided into prostheses and orthoses [27] (figure 2.1(a)):

prostheses operate in series with the human limbs, substituting them;

orthoses operate in parallel, encompassing the human limb. Orthoses can be further divided into exoskeletons and external devices:

- *exoskeletons* or parallel actuated devices have a structure that follows the human body linkage;
- *external* or peripheral actuated devices use external robots having a structure possibly different from the human limb.

So we have three kind of devices: prostheses, exoskeletons and external devices. Some of them are also *portable*, meaning that they can follow the wearers when they are moving. All the prostheses and some exoskeletons, depending on their purpose, are portable, while external devices cannot be portable.

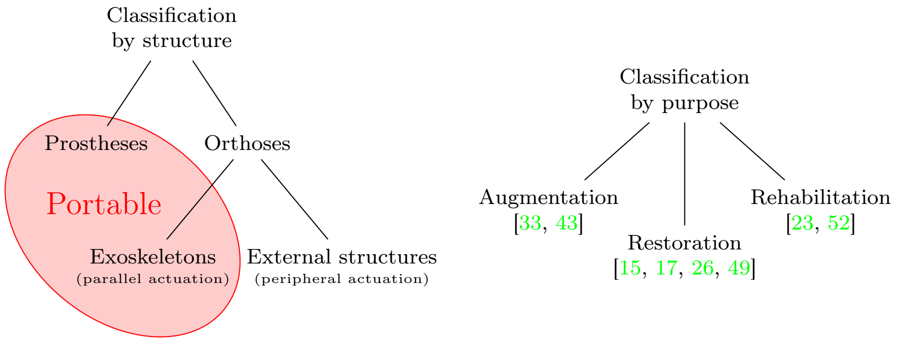
Focusing on exoskeletons, we can classify them also by purpose (figure 2.1(b)):

augmentation these devices are meant to be used by healthy people in order to augment their capacities, beyond human limits: to lift heavy weights, to enhance strength or endurance of soldiers, or to run faster and longer. Examples of exoskeletons are the BLEEX from Berkeley [33] or the Body Extender developed at Scuola Superiore Sant'Anna [43].

restoration these devices are designed to help in ADLs people who have lost normal capabilities due to injuries, degenerative or long term diseases. Their purpose is not to cure the person, but to restore while being worn the lost abilities to improve quality of life of, e.g., people suffering from SCI, paraplegia, tetraplegia, or neuromuscular diseases. Some exoskeletons of this kind are the HAL of Cyberdyne [26], ReWalk [17] or Ekso [15] to restore walking, MUNDUS project [49] to restore arm function.

rehabilitation these devices are supposed to *permanently* restore or improve damaged capacities, also when they are no more worn. They support physicians in giving physical therapies to people usually suffering from diseases with neurologic origin, like stroke, CP or SCI [23, 52]. Typically, they do not require an input from the wearer since they perform predefined tasks though they can be partially actively controlled to increase patient's involvement. There are few commercial exoskeletons of this type, both for lower and upper limbs.

Some exoskeletons which main purpose is restoration can be used also for rehabilitation, e.g. Ekso [15], if the subject can benefit of some long term improvements by wearing it. However, the main characteristic of restoration devices is that they should follow wearer's volition and be portable to help in everyday life.



(a) Classification by structure

(b) Classification by purpose

Figure 2.1: Classifications of wearable robots.

2.2 Sensor fusion for wearable robots control

Here we present sensor fusion methods for controlling wearable robots: we based our work on the review of sensor fusion techniques reported in [46]. The need of appropriate sensor fusion algorithms derives from the observation that existing wearable robots usually lacks the capability to adequately recognize the actions and intentions of the wearer.

In general, a way to overcame the limits is to provide them with different sensors and methods to infer the wearer's intentions. These may include not only methodologies that use myoelectric signals (EMG) or other electrophysiological measurements such as EEG, but also built-in sensors such as joint angle sensors or mechanical sensors placed on other parts of the body. This sensor-based *multimodal information* can give a more complete picture of the user will and multimodal information requires *sensor fusion* algorithms.

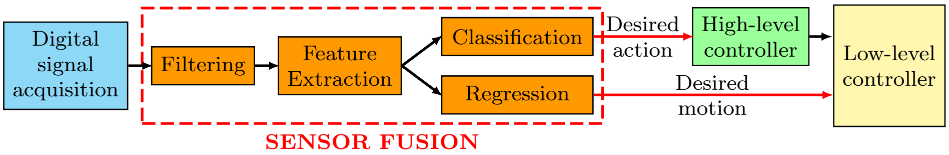


Figure 2.2: Sensor fusion architecture: general process of robotic decision making [46].

The figure 2.2 schematizes the general decision-making process of existing robots, the part of controller where sensor fusion takes place, as described in [46]. First, the input signals coming from sensors are acquired and digitalized, then the sensor fusion/decoding begins and it is generally composed by the following steps:

Filtering – *bandpass filtering* to remove raw digital signal components outside the range of frequencies we are interested in (low-frequency mechanical artifacts and high-frequency aliasing effects). *Notch filtering* to remove electrical noise at 50 or 60 Hz. *Spatial filtering* to remove unwanted signal components in the same frequency band as the useful signal.

Feature extraction – extraction of useful information ("features") from the filtered signals. This can be as simple as *rectification*, but more complex features, such as *spectral power* distribution, are also common. Commonly, features have a sampling frequency lower than the raw signals. *Segmentation*: division of the raw signals into "windows" (see figure 2.3). Features are extracted over the entire window and are output at the end of the window. A window can optionally overlap with the previous one to increase the frequency of features computation.

Classification and **regression** – are alternative to each other and a wearable robot generally utilizes one or the other. Multiple features can be used as inputs simultaneously.

- *Classification*: assigns a discrete label to extracted features (e.g. "hand closing", "leg lifting"). This discrete label generally represents the action that the user wants to perform, and an high-level robot controller is necessary to decide how to react to this desired action. Common implementations are based on supervised machine learning:
  - learn classification rules from a set of previously recorded and labelled training data;
  - the accuracy is defined as the percentage of correct class assignments.
- *Regression*: converts features to continuous values (e.g. joint torques). These values represent either the velocity/torque the user is trying to apply or directly the velocity/torque the robot should apply. Only a low-level robot control is required. Regressors are often implemented through supervised learning, but they can also use manually defined rules. Continuous output values allow smoother control, so the sampling frequency

of features for regression is generally higher than for classification and can be as high as that of the raw signals.

The robot's controller takes the results of classification or regression and converts them into the command given to the wearable robot's actuators, but classification requires more complex (*high-level*) robot control algorithms than regression.

High-level controller – given the high level action the robot should perform, outputs the velocity/torque the robot have to apply in order to do that action, while a low-level controller actually applies that velocity/torque.

Low-level controller – given the desired robot velocity/torque profile, feed the robot with the command (e.g. current, voltage) that ensures the application of it.

The model we adopted for the myoelectric interface in chapter 3 specifies and extends the model we just described. It is tailored to regression methods and focuses on the physical meaning of information exchanged between modules. The decoder module deals with the sensor fusion of myoelectric signals and, since it computes a continuos signal that represent the movement volition as muscular torque, belongs to the category of regressors. However, its output cannot be fed as-is into a low-level controller because it is not the command for the controller, it needs to be amplified since it is the force of a subject with weak muscles and/or transformed into the type of input expected by the controller that can be different, e.g. velocity. That is the purpose of the adapter module, making the connection between decoder and controller explicit.

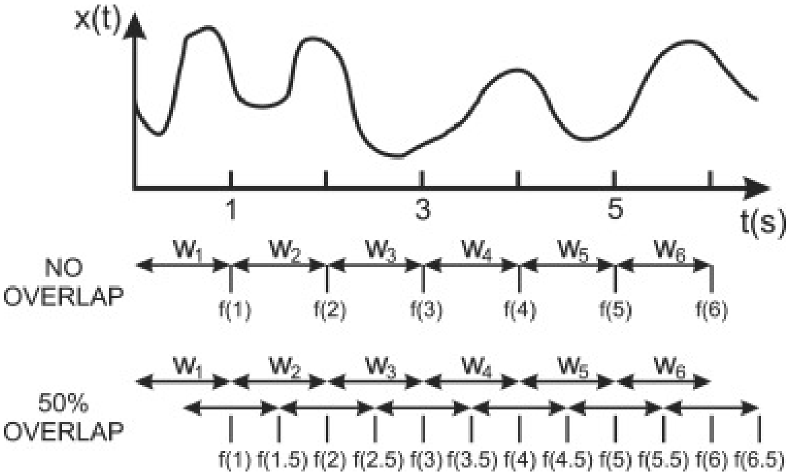


Figure 2.3: Windowing and feature extraction for a continuous signal $x(t)$. The windows is w and $f(T)$ is the feature extracted at time T from the corresponding window. In the first example windows do not overlap, while in the second each window overlaps the preceding one by half, so doubling the frequency of feature extraction.

2.2.1 Unimodal sensor fusion

Unimodal sensor fusion is a "traditional" sensor fusion method adopted for wearable robots, where multiple signals obtained from a single type of sensor are combined together.

	Classification			Regression				
Preprocessing	Bandpass filter			Bandpass filter				
Windowing	150-250 ms, overlapping			No windowing		150-250 ms, overlapping		
Feature extraction	Time-domain, root-mean-square, autoregressive			Rectification and smoothing		Time-domain, root-mean-square, autoregressive		
Sensor fusion	Single non-adaptive classifier	Single adaptive classifier	Parallel classifiers	Simple proportional control	Muscle models	Muscle synergies	Neural networks	State space models
Output	User's desired action			Control signal(s) for robot joint(s)				

Table 2.1: Resuming table of sensor fusion techniques for EMG signals.

This is usually the case for electromyography (EMG) or electrical brain activity signals. Recently, also mechanical sensors have been used in this modality, such as inertial measurement units (IMUs) and pressure-sensitive insoles. This kind of sensors can also be attached to parts of the body not covered by the device (e.g. for a prosthetic leg, can be useful placing an IMU on the intact leg). Other less common sensors are: intraneural electrodes, non-electrical muscle measurements (mechanomyography - MMG and sonomyography - SMG, forcemyography), eye tracking (electrooculography - EOG or camera-based), artificial vision and workload recognition.

Electromyography (EMG) The electromyography is a technique that allows to record the electric signals produced by muscle cells during muscle activation. This signal, visible 100 ms before muscle movement occurs, is suitable both for classification and regression. Table 2.1 summarizes the characteristics of sensor fusion steps adopted with this kind of signal, distinguishing when it is used for classification or regression. Cutoff frequencies for both the approaches are usually between 10-20 Hz as lower bound and 400-500 Hz as upper bound.

This measure can be done invasively (needle EMG) or not (surface EMG), but clearly we have higher signal quality with invasive electrodes. Many commercial prostheses still use *on-off* control, but it is outdated from a scientific perspective so it will not be further considered. Usually, classification is adopted to control prostheses, while regression for exoskeletons.

An important issue we encounter with EMG analysis is that signal characteristics vary significantly over time, so adaptive (supervised or not) or parallel classifiers should be preferred. Currently, the most effective regression techniques are proportional control and torque estimation via muscle models: such kind of models are very complex, so methods for online calibration should be further explored, as for example in [59].

Brain signals The methods described in this paragraph have the aim of recording the electric activity of the brain, generated in response to external stimuli or due to planning/intention of movement: table 2.2 briefly reports some information about these methods. The chosen technique strongly determines the sensor fusion approach that can be adopted:

- EEG approaches are not invasive, but with poor information content, so unsuitable for regression.
- Invasive approaches provide an high informational content and are suitable both for classification and regression, but they are not easily accepted by users so it is manly studied in animals like monkeys rather then humans.

These approaches have the advantage of allowing paralysed users to control a robot, even in total absence of motor activity. Classification of discrete classes by

	Electroencephalogram (EEG)			Electrocorticogram	Intracortical electrodes
	SSVEP	P300	Motor imagery		
Invasive?	No	No	No	Yes	Yes
Requires external stimulus?	Yes	Yes	No	No	No
Bandpass filtering	5–30 Hz	0.5–30 Hz	1-100 Hz	Not critical	Not critical
Sensor fusion approach	Classification	Classification	Classification	Classification or regression	Regression

Table 2.2: Resuming table of sensor fusion techniques for brain signals.

means of EEG is the most studied approach, since the high invasiveness of other media.

- *Steady State Visual Evoked Potential* (SSVEP) and *P300* signals require external stimuli, causing several drawbacks since a screen must be used to present the stimulus. Moreover, they have not yet showed great advantages with respect to an eye-tracking system: both these signals and eye-tracking require eye movement, but the latter has an higher information transfer rate and it suffers less of false positive recognition. SSVEP is the activation of visual cortex occurring when focusing attention on a flickering stimulus and its intensity is proportional to frequency of the stimulus. P300 is a characteristic signal of the visual cortex generated about 300 ms after perceiving an unexpected visual stimulus. The wearer can control the generation of these signal by locking or not locking towards the visual stimuli the system present, so giving the command to the device.

- *Motor imagery* EEG is a promising technology, since it is based on the change induced on brain waves by imagined motion (seen as power increase and/or decrease in some frequency bands) and it is independent from external stimuli: however, it requires user training and only a small number of images can be identified.

- *Electrocorticogram* (ECoG) requires electrodes implanted on the brain surface, the cortex, while *intracortical electrodes* are needles implanted through the brain surface: this last is the most invasive technique (and also the most precise). Although we can find very few studies on these invasive techniques for humans, it is well studied in animals like monkeys and some methods derived from them could teach something for other ones: use of adaptive methods (because electrodes once implanted cannot be moved), demonstrations of the adaptability of users, so robustness of the system, with respect to certain error types and different optimal parameters in offline and online use.

Other sensors Apart form EMG or EEG other sensors are also common. For example mechanical sensors, those that directly measure forces and kinematics, are widely adopted. Commonly these sensors are used as direct input to the controller. These can be multi-DOF force sensors or distributed pressure sensors used for impedance or zero-torque control. They can be useful to switch the device between operating modes, to detect potentially unstable situations that require intervention or to exploit the principle of complementary limb motion estimation (CLME).

Other important interfaces are eye trackers: these systems have the advantage that can be also used by severely impaired people, since few diseases completely block eye movement. Electrooculography (EOG), based on low frequency signals readable from the skin surface near eyes that change depending on the eye orientation, and camera-based tracking are the most common methods to identify eye orientation, but the latter is usually much more expensive and requires more advanced image processing methods.

We can see two examples of commercial sensors belonging to these two classes in figure 2.4.

(a) Myo™ armband – Thalmic Labs Inc.

(b) TobiiPro Glasses 2 – Tobii AB

Figure 2.4: Examples of commercial sensors: Myo™ armband and TobiiPro Glasses 2. (a) *Myo™ armband* is a wearable 9-axis IMU (gyroscope, accelerometer and magnetometer) with 8 medical grade stainless steel sEMG sensors, used as an innovative computer interface. (b) *TobiiPro Glasses 2* is an head-mounted eye-tracking system, featuring binocular tracking and a front camera, mainly used in marketing studies or medical applications.

2.2.2 Multimodal sensor fusion

Multimodal sensor fusion is a method that combines information from different sensor types, in order to overcome the limitations of each sensor taken individually. Figure 2.5 illustrates the different strategies that have been adopted:

Single fusion algorithm – Each modality generates features separately, then all features are given as inputs into a single sensor fusion algorithm (figure 2.5(a)).

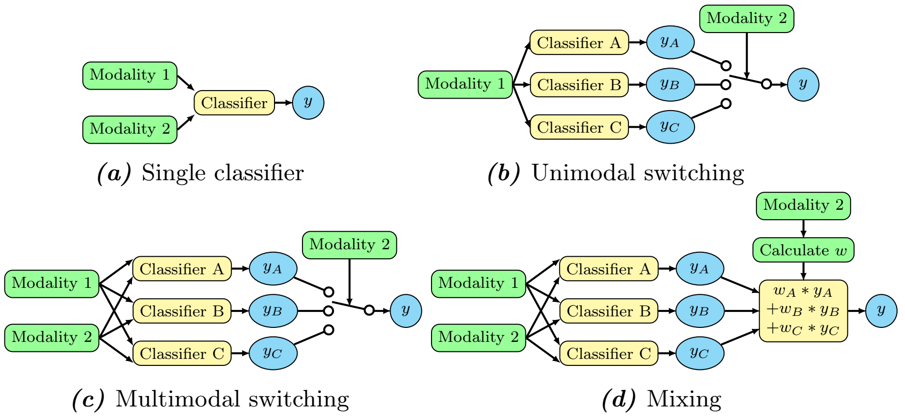
Unimodal switching – One modality is used to trigger the switching between operating modes with different sensor fusion algorithms. Each operating mode has a specific algorithm that uses only a second modality as input, so each one is unimodal (figure 2.5(b)).

Multimodal switching – As for unimodal switching, one modality triggers the switching between operating modes with different sensor fusion algorithms. Each sensor algorithm uses multiple modalities as inputs, so each one is multimodal (figure 2.5(c)).

Mixing – it is a generalization of switching methods, where multiple sensor fusion algorithms run in parallel, each using one or more modalities. The results of these algorithms are then "mixed" together, each weighted accordingly to values determined by one modality (figure 2.5(d)).

Second and third approaches are used when it is easy to distinguish between operating modes (e.g. discrete gait phases), while the last approach is used when operating modes are difficult to distinguish due to, e.g., sensor uncertainty or partial overlap. In wearable robotics, fusing EMG and mechanical sensors is the best explored multimodal sensor fusion, where we can find examples of all four approaches. It is also frequent the combination of eye tracking with either EEG or EMG, mainly for upper extremity applications, with either switching or mixing. There are also examples of EEG combined with either EMG or intraneural muscle signals. Finally, although rare, fusion of more than two modalities does exist (EEG, EMG and IMUs to detect and suppress hand tremor through a neuroprosthesis in [20]).

Possible additional signal sources that could be potentially combined with other modalities are interaction force sensors [28] and accelerometers [6].



(a) Single classifier

(b) Unimodal switching

(c) Multimodal switching

(d) Mixing

Figure 2.5: Approaches for two modalities sensor fusion. Reported for classification, but analogous for regression.

2.3 Biological origin of myoelectric signal

The myoelectric signal originates from the electrical activity of the neuromusculoskeletal system. Upon movement volition, the *central nervous system* (CNS) excites the nerves, transmitting an electrochemical signal to the motoneurons and the muscles that subsequently develop forces transmitted by tendons to the skeleton in order to generate movement. Each muscle contains muscle fibers (figure 2.6) that are responsible for the force generation. They are composed of myofibrillae, which in turn consist of aggregates of *actin*, a thin filament negatively charged, and *myosin*, a thick filament, also negatively charged, having a globular head containing one ATP molecule [12, 21].

The nerve action potential from the lower motoneuron causes a release of *acetylcholine* (Ach) at its neuromuscular junctions, each connected to a fiber. This sends a charge through *transverse tubules*[1] as muscle fiber action potential to the *sarcoplasmic reticulum*[2] and allows the latter to release positive *calcium ions* (Ca^{2+}) into the space between actin and myosin. Calcium ions bond with the actin filament making the myosin filament attracted to the now positively charged actin filament: the

[1] extension of the cell membrane creating a tridimensional system which penetrate the cell's centre and is responsible for the spreading of the action potential alongside the muscular fiber

[2] membrane-bound structure with a main function to store calcium ions

bonding of the myosin head to the actin causes the sliding of the filaments and, thus, the contraction.

Furthermore, the group of muscle fibers innervated by the same nerve axon and its branches are called together *motor unit* (MU) and they have homogeneous properties. The muscles can contain from few hundreds to thousands of motor units and the innervation ratio (number of muscle fibers per motor unit) varies largely among muscles [21]. The action potential on the nerve activates the corresponding motor unit, causing a *motor unit action potential* (MUAP) and consequently the fibers contraction. The action potential propagating through the muscle fibers of a motor unit generates extracellular currents that extend from the membrane to the skin surface, where one or more electrodes can pick them up. The extracellular currents recording provides the source for the electromyography (EMG) [12, 21], that use (surface) electrodes to convert the electric potential generated by the muscles into an electric signal, the myoelectric signal.

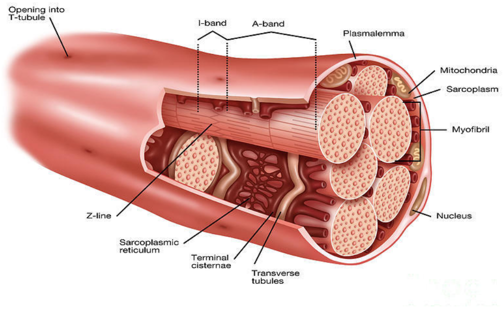


Figure 2.6: Structure of a muscle fiber.

The myoelectric signal results from the summation of MUAPs and its amplitude increases as the number of motor units recruited and the firing rate (i.e. the frequency at which motor units are activated) increase. The physiology of the muscle fiber determines the amplitude, shape and time course of each muscle fiber action potential, which together determine the characteristics of the MUAP and the EMG signal. In addition to recruitment (which motor units are activated) and firing rate, other features affecting the EMG signal include variations in muscle fiber length, fiber type composition, contraction velocity, muscle partitioning, variation in the distribution of sensory receptors, tissue filtering, and muscle temperature [21].

The complex interaction between the neuromuscular and the musculoskeletal systems that leads to the motor function and the myoelectric signal can be modelled as in figure 2.7, where the transformation of the neural excitation into force comprises two dynamic systems. The neuromuscular part is represented by the *activation dynamics* block that given the neural excitation input $u(t)$ of the motoneuron generates an internal muscle tissue state $a(t)$ related to the Ca^{2+} concentration. On the other hand, the musculoskeletal part is represented by the *muscle contraction dynamics* block that leads to the generation of the muscular force $F^m(t)$ [71].

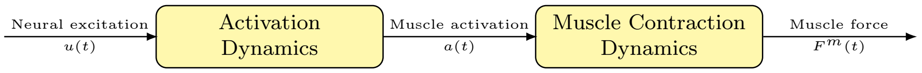


Figure 2.7: Muscle tissue dynamics.

CHAPTER 3

State of the art – part 2

A review of existing myoelectric interfaces

This chapter focuses on sEMG decoding problem and to the proposal of a logical decomposition of the myoelectric interface (section 3.1). In section 3.2 we will describe the interfaces considered to validate the calibration and evaluation methodologies described in chapter 4 during the experiments presented in chapter 5.

3.1 Myoelectric control architecture

The model we propose to describe a generic myoelectric interface is composed by three cascading modules with a specific role: *decoder*, *adapter*, and *controller* as represented in figure 3.1. Here we consider an *unimodal* sEMG *decoder* that map m emg_{c_j}s raw sEMG signal coming from m c_js channels to a signal d encoding the desired command for a specific wearable robot DOF. Internally, the decoder can be divided into two sub-modules, the first module extracts features f_{c_j} from the raw sEMG signals, the second maps these features into a decoded signal.In the case of regression-based decoding f_{c_j} and d are continuos valued signals, whereas in the case of classification-based decoding f_{c_j} and d are discrete values. In order to allow a continuos, proportional voluntary control paradigm, this thesis focuses on the former case.

The second module is the *adapter* that converts the decoded signal into a reference c_{ref} for the low-level controller. Its role is fundamental, since it allows to divide the problem of decoding the volition from sEMG signals and the control problem: using the same decoder, trained e.g. to decode isometric muscular torque τ_{mus}, we can control a desired physical quantity (e.g. torque, velocity, or other) using appropriate controllers. The adaptor *adapts* the decoded signal to the desired control reference, providing different levels of assistance.It can be implemented in the form of a linear gain, or more complex functions can be used, such as linear or non-linear admittance models[39].

The third module is responsible to *control* the robot's actuators and can be any of the controllers proposed in literature to move active exoskeletons. Its role is to generate a command τ_m for the motors such that the difference between the reference c_{ref} and the controlled quantity is minimized, while ensuring a safe interaction with the human wearer.

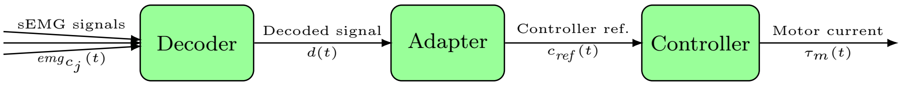


Figure 3.1: Myoelectric interface model.

3.1.1 Interface control paradigm

The terminology used below is drawn from literature on robotic prostheses [19], where the problem of volitional controlhas been more extensively explored. Therefore, there exists is a standard widely accepted terminology. There are four important attributes that can be given to a decoder, defining the control paradigm adopted by the interface. In order to be more precise while referring these terms in the following we report our intended meaning in the context of orthoses:

sequential the decoder can control one joint or discrete action at time and there exists a specific procedure, e.g. co-contraction, to switch *sequentially* between the various modalities, joints, or actions.

simultaneous the decoder can control many, if not every, joint at the same time, without any specific distinction or order between them. A decoder can be considered simultaneous even if the output is a discrete action: in this case, either the action itself involves more joints, or multiple actions can be performed at the same time or mixed. For example, actions involving different groups of joints such as reaching and grasping. What defines a decoder as *simultaneous* is that there are not explicit commands to change modality: the transitions are seamless.

pattern-based the decoder attempts to detect specific activation patterns from multiple channels in order to trigger a specific action or activate a specific group of joints, that is associated to the detected pattern. They are *classification*-based decoders.

proportional the decoder allows the user to modulate the intensity of the command; the information extracted from sEMG signal takes values in a continuous range, or numerous discrete levels, rather than a simple on/off signal. They are *regression*-based decoders.

A decoder obviously cannot be both sequential and simultaneous; systems where the command output is a discrete action are commonly pattern-based, sequential or more commonly simultaneous, while system controlled at joint level are more suitable for simultaneous proportional control. Here we are interested in simultaneous decoders that control directly device joints, but not in pattern-based ones.

3.1.2 Regression-based decoders for sEMG

Regression-based decoders extract form the raw sEMG a continuous valued signal, usually a torque profile or a position trajectory. They are mainly used as control interfaces for exoskeletons. They can be divided into two groups, depending on the way they model the muscular system and whether they consider the sEMG as an input or an output of the muscular system.

Forward methods – these methods consider the sEMG as the control signal of the muscular system. They model, explicitly or implicitly, the natural muscle's sEMG-to-force relationship or use a non-biological approximation, as linear proportional, then the torque output of the model is the command for the device: this is the way existing myoelectric interfaces for exoskeletons work [22, 38, 42].

- Explicit modelling [58] usually requires a complete musculoskeletal model to compute the actual musculotendon and torque-arm length. Then a muscular model computes the generated torque (common are Hill type models and Hammerstein-Wiener). The muscle fibre has three components: an active component, an elastic one, and a viscous dumper. The tendon model can be stiff or compliant. Unfortunately, not all the parameters involved have physical meaning.
- Implicit modelling can be done through various classic regression methods: linear regression, non-linear kernel methods, neural networks [24]. Recently also convolutional neural networks have been proposed [5].
- Non-biological modelling use simple mathematical relationships, known to be much more simpler then the actual model, to map the sEMG value direct proportionally to the torque command. Linear proportional models are the most used, as in [36]. Differently from explicit and implicit modelling, these approaches renounce to an accurate prediction of the biological torque for rough estimate. In return, they require less data and less time to be trained.

Backward methods – these methodologies consider the sEMG signal as visible output of the underlying process of voluntary movement, not as a command to give to the device. The objective is to estimate the state of this process:

- State-space models use a model for the dynamic evolution of the state (usually position and velocity) and an observation model to relate the observed sEMG to the estimated state. Usually the state estimation uses a Kalman filter.
- Muscle synergies rely on the principle that the human motor system can only control groups of muscle. Non-negative matrix factorization (NMF) is usually employed to identify these groups.

Some major drawbacks of forward methods:

- sEMG-to-force relationship is non-linear, dependent from joint configuration, electrode placement, and fatigue [30].
- The physiology that lays behind muscle force generation has not been completely clarified yet [63].

- Explicit modelling, i.e. model-based approaches, requires the estimation of many parameters [56, 58].
- Implicit modelling, i.e. model-free approaches, requires a great amount of diversified data and, since the inner workings of these methods are hidden, they are not easily *generalizable* [58].
- Non-biological models, as linear proportional models [36], are promising since they require a simple calibration, but they are not widely studied.

Issues of backward methods:

- State-space models usually require a model of the motion, specific for the task (e.g., trajectories, targets, etc.), to estimate the evolution of the state.
- As the forward methods, they require model training, so an effective method to perform it on target population must be developed.

3.2 Linear decoders

We selected a subset of the decoders proposed in literature and made a case of study, testing the closed-loop behaviour in combination the control paradigm described in section 3.1. We focused on *proportional* decoders that can be described as a functional map between m emg_{c_j}s raw sEMG signal coming from m c_js channels and a continuous signal encoding the volition d, e.g. muscular torque τ_{mus}. Logically, it can be divided into two phases:

(1) Transformation of raw sEMG signals $emg_{c_j}(t)$ into feature signals $f_{c_j}(t)$, that should capture the underlying muscles activation level, i.e. the control information carried by the sEMG signal.

(2) Transformation of the features $f_{c_j}(t)$ coming from all the channels into the volition signal $d(t)$, using a *time-invariant linear mapping.*

It has been shown for prostheses control that a *time-invariant* map between features and output command along with a closed-loop feedback to the wearers, enable the brain's ability to create an inverse model of the assistive device [29], allowing an *human-embedded* controller where the user directly controls the device into its task space [1]. Interface-specific synergies can also be developed for a myoelectric control robust against sEMG signals intrinsic variability [30].

In our case of study we decided to follow the literature [36, 58], for which the volition $d(t)$ is encoded as muscular torque $\tau_{mus}(t)$. We think this as a natural and effective choice for two reasons:

- muscles themselves “output” torques;
- residual muscular torques can be measured also in target population, i.e. it is possible to build a mapping with sEMG.

It is responsibility of the subsequent modules to amplify and adapt this signal to control the device.

On the other hand, there are many approaches where the quantity controlled is the velocity [46], or where the decoder's torque is converted into a velocity reference through an admittance model [39]. The decoders models we present (subsection 3.2.2) can be used – in theory – to generate velocity estimations instead of torques: they are generic linear regression models, their output depends on the data they are trained with. However, such a dataset linking sEMG signals to desired velocity, cannot easily collected from target population.

In any case, the description of the myoelectric control architecture is not affected by the actual meaning of the $d(t)$. It suffices to be a voluntary controlled signal expressed in an arbitrary unit. The problem of decoding the volition from the sEMG signal is divided from the interpretation of it as device command, that is the control paradigm: selection of the assistance level and conversion to the required measure unit, *adapter*, and device low-level control, *controller*.

3.2.1 Feature extraction

Feature extraction is required for sEMG signals elaboration since using the raw myoelectric signals is impractical: they are intrinsically noisy and have an high frequency dynamics [4]. Roughly speaking, the information on muscle activation level is encoded as amplitude and frequency content of the measured sEMG signal [18]. Each motoneuron discharge causes an *action potential* (AP), an electric signal with a stereotyped time course; the summation of the APs coming from all the active motor unit propagates from the muscles through the tissues reaching the skin, where the electric signal is read by the electrodes. The firing rate ranges between $5 \sim 10$ (minimum activation) and $60 \sim 100$ Hz (fully activated), while the amplitude is in the order of hundreds of µV and depends on the size and the number of active muscle fibres [16]. The raw sEMG signal elaboration starts *bandpass filtering* it in order to remove low frequency mechanical artefacts and high frequency aliasing effects, between 10-20 Hz and 400-500 Hz [46]. *Notch filters* or *spatial filters* can also be applied, the firsts to remove line interference (50 Hz in Europe, 60 Hz in North America), the latter to remove unwanted signals in the frequency of interest (e.g. hearth electric pulses – ECG). Many features have been proposed both for classification and regression, ranging from simple *rectification*, *linear envelope* [36], *moving-average* filters [34], to more complex features, such as *spectral power* distribution (see [46, 51] for details). Windowed signal features usually adopted for classification [51, 69]. In detail, some of the most common features used in literature:

Linear Envelope – the filtered sEMG is fully rectified $rect_{c_j}(t) = |emg_{c_j}(t)|$, and then passed through a second order critically damped low-pass Butterworth filter, usually with a cutoff frequency of $2 \sim 3$ Hz, to obtain a smooth signal [36].

Activation dynamics – a second order dynamic system is used to filter the signal, in order to resemble the muscle twitch response or other muscles' dynamic characteristics [34, 58]

Windowed features – the raw signal can be *segmented*, i.e. divided into "windows", then features are extracted over the entire window and are output at its end. A window can overlap with the previous one to increase features sampling rate, which is particularly useful in regression. The functions applied to the windows can be time-domain, e.g. RMS, standard deviation or mean absolute value, or frequency domain based Fourier or wavelet transform [44, 50, 51]. Window lengths are usually comprised between 100 ms and 500 ms.

Although windowed features are more common in classification-based approaches [51], there are also some examples where these have been used for regression [46].

The last passage, before using the features, usually requires normalization with respect to maximum voluntary contraction (MVC) values, to minimize variability due to electrodes placement [4], and thresholded to remove unwanted baseline noise.

3.2.2 Linear model decoders for sEMG

After feature extraction, the signals coming from different sEMG channels should be combined to obtain the signal $d(t)$, the decoded movement volition. We consider decoders that use *linear* models to map the feature to the decoded signal. The parameters of these models cannot be manually selected, but should be selected by fitting the model to predict a target quantity, in our implementation the muscular torque. It is worth notice that these are generic *non-biologic* models that make no assumptions on the nature of the signals involved.

Linear decoders are here described by building blocks that can be composed in different way to obtain various of them. Table 3.1 summarize the blocks considered. *Regressors* are the main blocks as they compute the output using a linear model of the sEMG features. *Transformers* modify the input features, e.g., by reducing their dimensionality or delaying them. *Classifiers* can identify the intended motion direction and select between different regressors depending on the class of motion identified.

The advantage of linear models over more complex but more precise non-linear models, is that parameters values can be obtained by rapid gradient-based optimization procedures or *ordinary least-square* (OLS) regression [53] in a matter of seconds, reducing the cost of re-training only to training dataset acquisition.

Regressors The *plain proportional* decoder uses two sEMG channels and was proposed by Lenzi [36]. The *multi-channel proportional* decoder uses $m > 2$ sEMG channels, multiplying the features by a linear coefficients vector and adding an offset. To the best of our knowledge, decoders based on *Autoregressive with Exogenous inputs* (ARX(p, b)) models have been applied for sEMG decoding by Bobet and Norman [7], but never used for exoskeleton control. They are composed by an autoregressive and an exogenous input part of order p and b, respectively. The model can also include an input delay of d samples.

Transformers Common *dimensionality reduction* methodologies applied to sEMG signals are Principal Components Analysis (PCA) [2] and Non-Negative Matrix Factorization (NNMF) [57]. These techniques are useful to lower the decoder's sensitivity to sEMG signals variability, considering only the most significant $n < m$ features. We know that the sEMG signal anticipates the actual torque generated by muscles up to 100 ms [10, 45] and this can be exploited by building models that fit sEMG to a delayed version of the torque: only decoders based on muscle models explicitly consider this delay [37].

Classifiers They are mostly based on Linear Discriminant Analysis (LDA). They have successfully been used by Hahne [25, 65] and Scheme and Englehart for simultaneous proportional of control hand prostheses [61]. In these works, the velocity of the selected motor function is proportional to an dimensionless value, such as the sum of the feature values. Nevertheless, the classifier could be used in a multimodal system to choose between different regressors trained for the specific direction.

Table 3.1: Summary table of linear myoelectric decoder modules. A simple linear decoder uses one of the regressors in combination with one or more of the optional transformers. The classifier module can be used to choose between different parameter sets of the selected decoder. Different decoders arise from different compositions of these modules. The number of parameters that are user-specific are computed considering a single DOF system: normalization factors of the features are not accounted for.

	Decoder module	Formula*	Input channels m_{in}	Output channels m_{out}	*#parameters* (1 DOF)
Regressors	Plain proportional	$D_{PP}(t) = g_e \bar{f}_{c_e}(t) - g_f \bar{f}_{c_f}(t)$	2	1	3
	Multi-channel proportional	$D_{MC}(t) = \boldsymbol{w}^T \bar{\boldsymbol{f}}(t) + w_0$ with $\boldsymbol{w}$ containing the m_{in} regression coefficients	≥ 2	1	$m_{in} + 2$
	ARX(p, b) proportional	$y(k) = a_0 + \sum_{i=1}^{p} a_i y(k-i) +$ $\sum_{i=0}^{b-1} \boldsymbol{b}_i \bar{\boldsymbol{f}}(k-d-i) + \varepsilon_k$ with a_is for AR(p) and $\boldsymbol{b}_i$s m_{in}-dimensional vectors for EX(b)	≥ 2	1	$p + bm_{in} + 3$
Transformers	PCA(n)	$\boldsymbol{f}_{red}(t) = \boldsymbol{W}^T(\bar{\boldsymbol{f}}(t) - \overline{\boldsymbol{x}})$ with $\boldsymbol{W}$ the $m_{in} \times n$ matrix of the first n principal components and $\overline{\boldsymbol{x}}$ the sample mean	$> n$	n	$(n+1)m_{in}$
	NNMF(n)	$\boldsymbol{f}_{red}(t) = (\boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T\bar{\boldsymbol{f}}(t)$ with $\boldsymbol{W}$ the $m_{in} \times n$ matrix of the n non-negative factors	$> n$	n	nm_{in}
	d-samples delay	$\boldsymbol{f}_{out}(k) = \bar{\boldsymbol{f}}(k-d)$	≥ 1	m_{in}	1
Classifiers	LDA(2)	$[c_e \vert c_f]^T = \boldsymbol{W}^T \bar{\boldsymbol{f}}(k)$ with $\boldsymbol{W}$ the $m_{in} \times 2$ matrix of the discriminant 2 directions $class = \begin{cases} ext, & \text{if } c_e > th \text{ and } c_e > c_f \\ flex, & \text{if } c_f > th \text{ and } c_f > c_e \\ no\ move, & \text{otherwise.} \end{cases}$	≥ 2	$3^{\dagger}$	$2m_{in} + 1$

* $\bar{\boldsymbol{f}}$ is the features vector normalized and thresholded.

† these are the output classes: *flexion*, *extension*, and *no move*.

3.3 Control paradigms

1*-DOF *controllers and adapters

The controller module defines the interaction with the environment. When combined with the adapter, the two modules define also the interaction with the user. We call this compound the *control paradigm.*

The decoder's output $d(t)$, related to the volitional muscles activation, is subject to different interpretations depending on which modules define the control paradigm. In general, we can identify two general approaches to the control paradigm: when the controlled physical quantity is a *velocity* (or position); or when it is a *force.*

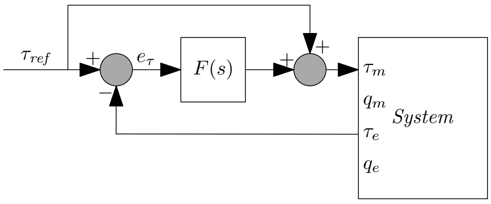


Figure 3.2: Block schema of basic force control (explicit). The feed-forward path improves the accuracy.

In the following, the decoder's output $d(t)$ is regarded as a torque prediction, but similar modules can work also when the decoder works differently.

3.3.1 Force control and gravity compensation

Some of the controllers we present rely on a low-level *force controller* $F(s)$, some others on velocity (or position) controllers. Depending on the actuation mechanics, the force controller $F(s)$ can work in open loop (implicit force control) or in closed loop (explicit force control), where a force sensor is used to mask the inherent actuator dynamics [8, 72] (see figure 3.2).

Force control enables gravity compensation of the exoskeleton and the arm in it. This basic control paradigm allows the movement even without a decoder since gravity is the major force muscles have to overcome. We argue that gravity compensation is the best control paradigm in term of velocity and accuracy, as it allows a natural, direct, control taking full advantage of residual forces, if they are enough. When in free motion and without external loads, the residual forces have just to overcome frictions and inertia. Also position controllers can benefit of improved accuracy from gravity compensation, since it removes a known force offset. We consider only the compensation of known weights, i.e. exoskeleton and subject's arm, and not of external loads which are, generally, unknown. In case of not compensated external loads, an active decoder is required.

Unfortunately, accurate identification of gravity compensation parameters is extremely challenging because of human/robot joint misalignments.Gravity compensation is not robust to kinematic misalignment between the wearer and the exoskeleton and in the case of severe pathologies even a slight overestimation or underestimation of the gravitational load may result in an uncomfortable system for the user. However, preliminary tests showed that slight errors in the gravity compensation are acceptable when they result in under-compensation, while over-compensation or, worse, mixed under and over compensation intervals in the workspace are obstructive. Indeed, over-compensation is very unnatural and fatiguing, requiring the wearer to constantly contrast the device, even in pro-gravity movements. To describe under-compensation in this work we use the formula $\alpha g(q)$, with $\alpha \in [0, 1]$ as near as possible to 1 and $g(q)$ represents the gravitational forces acting on the joint.

3.3.2 Adapter modules

We identified tree kind of adapters: *gain* adapter, *integral* adapter, and *admittance* adapter. They are linear dynamical system of increasing complexity interleaved between the decoder and the controller, that is a *low-level* controller.

A *gain* adapter is a gain – possibly dimensioned – that multiply the decoder's output: $A(s) = \beta D(s)$. It can be viewed also as a 0-order system (no dynamics). We can use it, for example, to connect the decoder to a force-based controller, then it is a dimensionless gain that adjust the assistance level, or to a velocity-based controller, where in addition to the regulation of the assistance it also changes the measure unit.

An *integral* adapter, within the scope of this work, is a system composed by a gain followed by an integrator: $A(s) = \frac{\beta}{s} D(s)$. In general, it is a 1-order system. This approach results for the user in the control of the entity of the variation of adapter output. The β parameter is both a conversion coefficient and the regulation of the level of assistance.

An *admittance* adapter is a virtual mass-damp system controlled by the decoder's output; the velocity (or the position) of this virtual system is used as reference for the controller. When controlling the velocity is a 1-order system, as an *integral* adapter: $A(s) = \frac{1}{Js+b} D(s)$. When controlling the position, it introduces a second integrator becoming a 2-order system (double integral): $A(s) = \frac{1}{Js^2+bs} D(s)$. This approach results for the user in the control of the force that moves a damped mass and it has two parameters: J and b. Here, the conversion function of the module is evident, from forces to velocities (or positions), but the assistance level regulation is also present. In fact, it is substantially hidden in the mass value J: as it decrease the assistance level increases and vice versa. To maintain the same damping effect, independently from the assistance level, the b parameter should be decreased/increased at the same ratio of the J parameter reduction/augmentation.

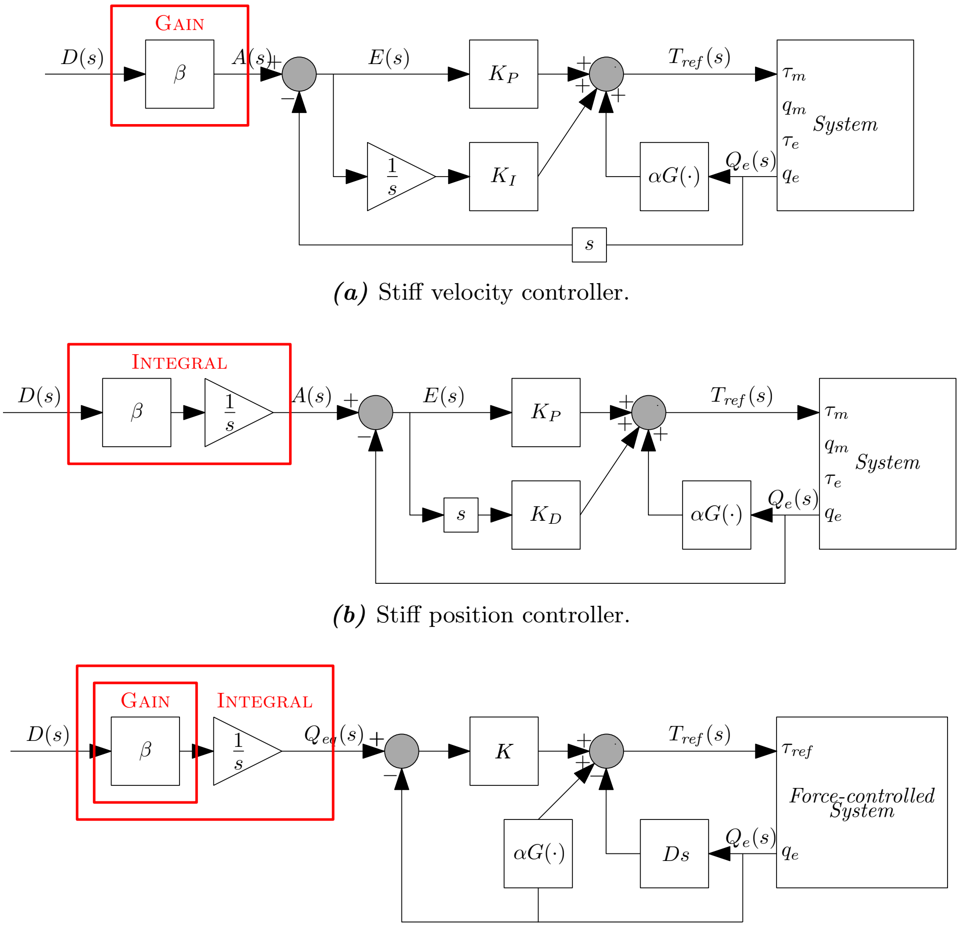
3.3.3 Velocity-based control paradigms

These control paradigms maps the decoder's output into a *velocity* command: their block schemas are reported in figure 3.3. The *stiff velocity controller* uses a classical PI velocity controller combined with a gain adapter, to make the decoder's output a properly dimensioned velocity reference. Similarly, the *stiff position controller* uses a classical PD position controller combined with an integral adapter, to make again the decoder's output a properly dimensioned velocity reference. The control paradigm emerging from the two controllers is exactly the same, but there are some minor implementation differences in the control properties.

The *impedance controller* has as reference the (velocity of the) equilibrium point of a virtual spring, with fixed stiffness and damping. It uses a low-level force controller to provide the force required by the virtual spring model. The adapter to the decoder can be either an integral adapter if the controller wants a position reference, or a gain adapter if it requires a velocity reference Compared to the stiff velocity/position controllers, the impedance approach leads to a softer control action since the internal loop is closed on the force and not on the velocity, allowing the user to deviate form the reference velocity and provides a safer interaction with the surrounding environment.

3.3.4 Force-based control paradigms

These control paradigms maps the decoder's output into a *force* command: their block schemas are reported in figure 3.4. The *direct force controller* uses a gain



(a) Stiff velocity controller.

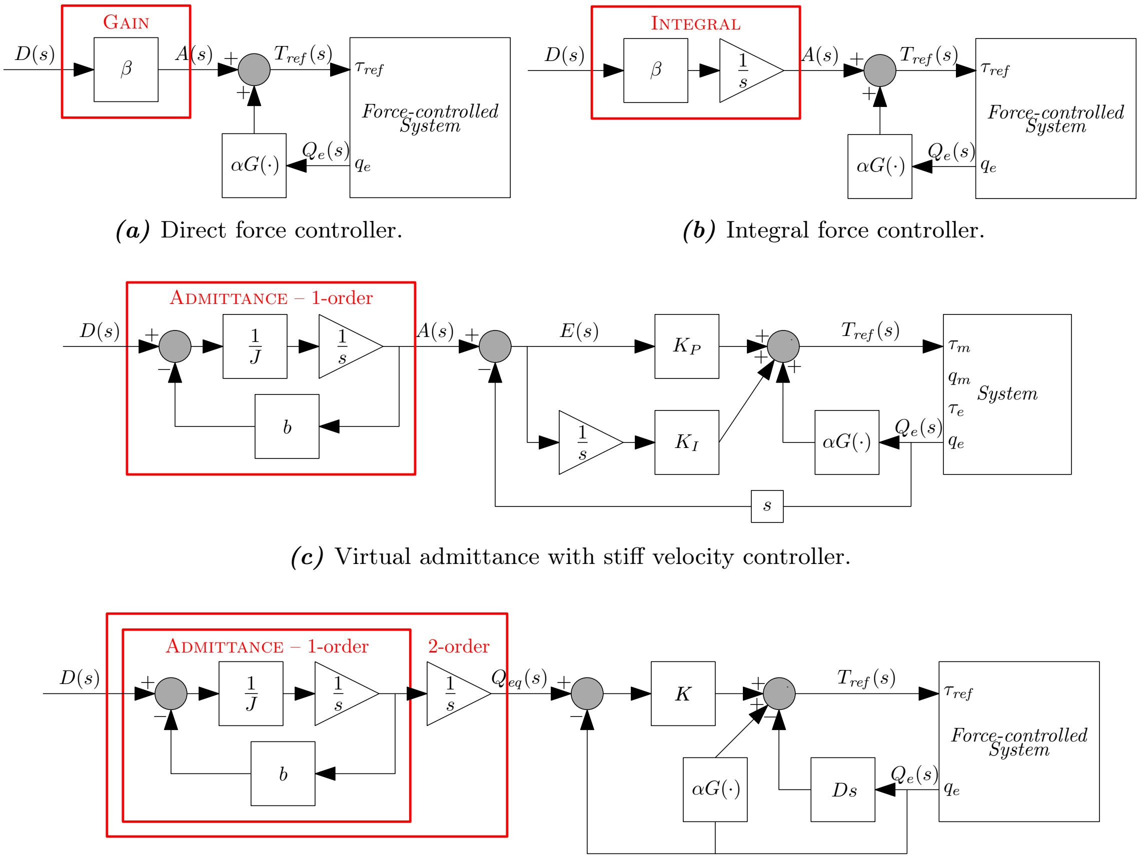
(b) Stiff position controller.

(c) Impedance controller.

Figure 3.3: Block schemas of velocity controllers.

adapter which output is added to the gravity compensation force reference. The *integral force controller* interprets $d(t)$ as the derivative of the force reference. The adapter is an integral adapter with scale factor β_f. The advantage of this controller over the plain version is that in this control paradigm no contraction generate a constant force command since we control the variation of the force.

The *virtual admittance controllers* allow to translate the decoder's force output into a velocity reference, so that a velocity-based controller can be employed. The decoded signal drives an admittance adapter with a *virtual* mass-damping system and the resulting velocity/position is used by a velocity/position controller. In the first case, it is a first order system, in the latter a second order.



(a) Direct force controller.

(b) Integral force controller.

(c) Virtual admittance with stiff velocity controller.

(d) Virtual admittance with impedance controller.

Figure 3.4: Block schemas of force controllers.

3.3.5 Mixed controller: force and velocity input

Some preliminary test revealed that direct force controller can be imprecise, while velocity-based control can be "slow" reacting (maybe because of the integration hidden in the passage to position). Is it possible to mix them in order to obtain a better trade off between precision and reactivity? This lead to the idea of "mixed" controllers. These accept two inputs: a velocity (or position) and a force. The two inputs can in general have different sources, but since we have only one output from the decoder, we can split the signal, connect it to two different adapters, each connected to an input. So the single decoder's output $d(t)$, can be viewed as a force when connected to the *force-adapter*, or as a velocity when connected to the *velocity-adapter*. This architecture is illustrated in figure 3.5.

The base of mixed controllers is the impedance controller: beyond the equilibrium point input, a force reference can be added to the virtual spring model output. The torque reference for this mixed controller is:

$$\tau_{ref}(t) = K(q_{eq}(t) - q_e(t)) - D\dot{q}_e(t) + a_f(t) \tag{3.1}$$

where $q_{eq}(t)$ and $a_f(t)$ are, respectively, the equilibrium point and the force reference, $q_e(t)$ and $\dot{q}_e(t)$ position and velocity on the environment side, K and D the virtual spring model coefficient and dumping.

The combination of *impedance* and *direct force controller* – figure 3.6 – gives rise to a controller that is equivalent to the impedance controller when $d(t) = 0$, while there is an additional force $\beta_f d(t)$ in the direction of the variation of q_{eq} when $d(t) \neq 0$. On the other hand, it is easy to show that the combination of *impedance* and *integral force controller* is equivalent to the *impedance controller*, with its β_v' parameter set to $\beta_v + \frac{\beta_f}{K}$.

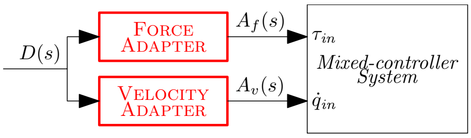


Figure 3.5: Architecture of a mixed controller.

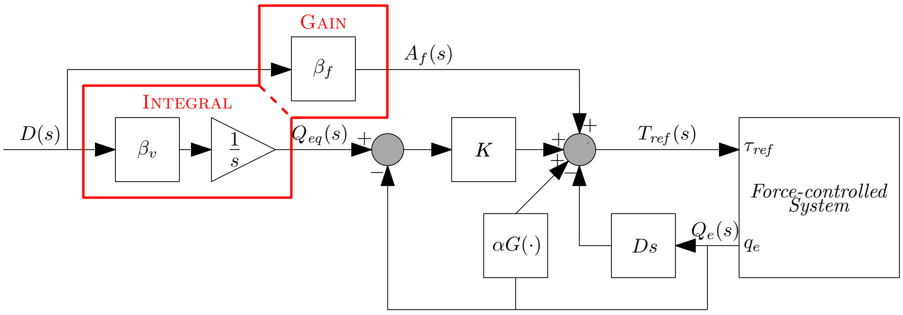


Figure 3.6: Block schema of the combination of impedance and force controller.

CHAPTER 4

Contributions

The review of sEMG signal processing, decoding, and control paradigms in the previous state of the art chapters 2 and 3 was fundamental to define a generic *myoelectric control architecture*, a guideline within the wide and inhomogeneous topic of myoelectric interfaces. We focused on a specific case of study, linear regression decoders and 1-DoF controllers, as baseline models to show the emerging behavior of the closed-loop system, analysing also "contour properties" too often neglected, such as parameters tuning.

Although the research on this topic is extensive, to the best of our knowledge every existing research work neglects one or more fundamental aspects. As a matter of facts, none of the proposed solutions is *complete*, i.e. covers the full path from the sEMG signal acquisition to device control and is suitable for motor impaired people. Moreover, existing solutions are never – a part from few [39] – tested on impaired subjects. Furthermore, a comparison between the different myoelectric interfaces is still missing in the literature. Such a comparison is prevented by the differences between the interfaces and their dependence on specific hardware. To fill this gap in our work we defined a comprehensive *myoelectric control architecture*, with interchangeable components, and a test bed complete in all its parts, from sEMG signal acquisition to actuation, but also we developed a training procedure for physically impaired subjects and performance evaluation methods.

We will look for the best combination of myoelectric control architecture components, built with the decoders, adapters and controllers we reviewed in chapter 3. In section 4.1, we describe the procedure we adopted to acquire training data, while section 4.2 explains the methodology adopted for interface performance evaluation. We know the training dataset collection procedure has some limitations on its applicability to people with too low residual forces. Therefore, in section 4.3 we propose an alternative *computer-guided* data collection procedure to overcome this limitations; it is only proposed since it has never been tested within our experimental framework.

4.1 Isometric torque dataset collection

The objective of this procedure is to collect a dataset relating sEMG and isometric torque at the elbow, recording their values at the same time. The procedure requires a position-controlled elbow orthosis equipped with a torque sensor, as the device described in chapter 5, and an sEMG sensor. The device is placed with an active controller to a fixed comfortable position, usually an intermediate point between full extension and full flexion. Then, the wearer has to repeatedly try to extend and flex the elbow, in contrast with the device, following an on-screen torque reference. The sEMG is measured with the sensor, while the muscular isometric torque is obtained, reversed in sign, removing the weight from the torque measured by the sensor. This is isometric because no movement happens.

We highlight the importance of collecting a good dataset to capture the actual underlying relationship. We found out that to guarantee consistent high quality datasets it is fundamental to provide a feedback of the measured torque measured as well as a guide. Otherwise, the quality of the regression can vary a lot, depending on the dataset. So we provided a program that gives a visual torque reference together with sensor feedback: details on the implementation are described in subsection 5.2.3.

Although this procedure is easy to implement and use, it has some limitations that, hopefully, the alternative *computer-guided* procedure could solve (section 4.3):

- can only be used by people with a certain amount of residual force;
- the relation recorded between sEMG and torque can change in online execution because of non-isometric conditions and closed-loop interaction.

4.2 Performance evaluation tasks

Task-based performance evaluation allows us to give a significative – from the user's prospective – score to the specific myoelectric we are testing. To do this we defined three generic tasks, each described by an human-level objective, a set of performance metrics and a reference trajectory, representing the *golden standard* for that specific task. The first two are target-oriented tasks, an adaptation of the one proposed in [11] to include the Fitts' law tasks of [61], while the third is a periodic task.

First two tasks have been implemented by the *performance evaluation interface* (PEI) used during the experiments. For details see subsection 5.3.1.

4.2.1 Fixed Target (FT)

Fixed Target (FT) task tests the ability to repetitively reach and stop on a target that do not change its position and size. The task consists in reaching a target, stay there until it is *acquired* i.e. the end effector stay in its area (a box around the target position of width W) at least for the dwell time T_{dwell}, then the repetition ends and the next target is the starting position and also the return path is evaluated: this procedure is repeated N_{rep} times. If a repetition fails, the task can continue after that the subject has reached the failed target since it is the starting point of the next repetition. Target distance and ta size are the same for all the repetitions, both forward and backward, so "index of difficulty" ID is constant and computed as [61]:

$$ID = \ln\left(1 + \frac{D}{W}\right) \tag{4.1}$$

where D is the distance between starting and target position and W the target width.

We formally defined the following performance measures (i refers to the repetition number):

Throughput $TP_i = \frac{ID}{MT_i}$ where MT_i is the time spent to acquire the target of difficulty ID (fixed).

Efficiency $E_i = 100 \cdot \frac{SP}{AP_i}\%$ where SP is the shortest path length between starting and target positions (fixed), computed as target distance minus target width $D - W$ and AP_i is the length of the path actually travelled excluding the *stopping distance* SD_i.

Overshoot O_i counts the target reaches, i.e. the end effector enters the area within the target width, for which the target was lost before *acquiring* it.

Stopping Distance or **Target Path** SD_i is the path travelled within the target width during the dwell time, the last target reach that lead to target acquisition. To allow comparison between tasks with different target width, we introduced a scaled version of this measure, that is $\frac{SD_i}{W}$, the plain value scaled by the target width.

Completion Rate $C = 100 \cdot \frac{\#completed}{N_{rep}}\%$ is the percentage of test completed before the task end.

In addition to these task performance measures, we considered also an effort measure linked to the EMG signal. It is widely accepted [21] that the integral of EMG envelope is a measure of the muscular effort and fatigue, so we adopted this measure computing for each repetition the sum of the integrals on each EMG channel for the duration of the task and called it *IEmg*. However, this measure is penalizing of tasks longer in duration, since low effort long task can have an higher IEmg than intensive high effort short task, so we introduced an alternative measure that is the mean integral of EMG envelope and called it *mean IEmg*, that is IEmg divided by repetition duration.

The reference trajectory is the state vector of a discrete LTI system convergent to the target position. For details please refer to [11].

4.2.2 Moving Target (MT)

Moving Target (MT) task tests the ability to reach and stop on random not moving targets, each its position and width. The task consists in reaching N_{rep} targets: there is a time limit of T_{max} for each target and once a target is reached and acquired, the next is presented. Position and width of targets are randomly chosen within fixed intervals in such a way that two consecutive targets there is a minimum distance D_{min}. As for *FixedTarget*, failing to reach and acquire a target means failing the task and the performance measures are the same. The reference trajectory for *MovingTarget* is computed as for *FixedTarget* except that the target position changes over time.

4.2.3 Tracking (Tk)

Tracking (Tk) task tests the ability to follow a trajectory, with variable amplitude and/or pace. The trajectory to follow $x_r(t_k)$ is composed by N_{slots} sinusoids lasting T_{slot} seconds, each having its own amplitude $A_i \in [A_{min}, A_{max}]$ and frequency $F_i \in [F_{min}, F_{max}]$. At the end of each time slot, a binary random variable decides whether the amplitude or frequency must be changed on discrete steps within admissible ranges with a smooth slot transition. A variant of this task can fix the value of the amplitude or the frequency.

The performance measures defined for this task measure how the actual and the reference trajectories are close and include:

- RMS error on positions – $RMS_{pos} = \sqrt{\sum_{k=0}^{N} \frac{(x(t_k) - x_r(t_k))}{N}}$;
- on velocities – $RMS_{vel} = \sqrt{\sum_{k=0}^{N} \frac{(\dot{x}(t_k) - \dot{x}_r(t_k))}{N}}$;
- maximum cross correlation coefficient – $MCC = \max_d r_{x,x_r}(d)$;
- time delay – $k_d = \arg\max_d r_{x,x_r}(d)$.

These values are computed both on the overall trajectory and per time slots, grouping them by amplitude and frequency and the reference trajectory for *Tracking* task is the sinusoid itself.

4.3 Proposed computer-guided dataset collection

Automatically tune the decoder models require training data that explicit the association between myoelectric signals and intended motion. If the subject is able to autonomously move, we can record at the same time sEMG and either the forces or the velocities of a movement set (e.g. [3]), otherwise we need another methodology. In section 4.1, we described a procedure that allows to collect isometric torques, but it has some limitations, first of all it requires a minimum amount of residual forces. If theses are too low, alternatives must be found.

Corbett et al. [11] proposed a methodology to acquire data for a myoelectric and gaze controlled arm exoskeleton to be used by SCI subjects: some muscles are still under voluntary control and viable sources of sEMG signals, but autonomous arm movements are impossible. To map the myoelectric signals to the desired motion, the user volition was fixed by imposing the exoskeleton to follow a predefined trajectory for a reaching task and the wearer, knowing beforehand the device movement, was instructed to "follow" it. In this way we associate the trajectory of the robot to the muscles activation pattern. This methodology could also be used to acquire training data from people suffering from muscular diseases. In addition, in our *computer-guided* procedure we propose some improvements:

- periodic tasks are added to record the mapping in a wider range of situations;
- iterative refinement of the decoder, new data are acquired progressively decreasing the predefined trajectory control and increasing the decoder control, trained with the previous iteration data;
- the tasks used to evaluate the system performances are the same used for data collection, since "optimal" trajectories and functional scores are defined.

The motivation behind these extension are: the reaching tasks alone could not cover the totality of ADLs movements, but in combination with periodic tasks the coverage should increase. The iterative training account for differences between the myoelectric signals generated when the subject is "passive" and when the movement is actively controlled by the generated signals (biofeedback). The definition of performance metrics upon the same task used for data collection gives a standard way to evaluate the quality of the movements allowed by the system and a mean for comparison with healthy people movements, that we consider the *golden* standard.

CHAPTER 5

Experiments

This chapter describes the experimental sessions performed on myoelectric control. Section 5.1 describes the preliminary experiments to assess the characteristics of the sEMG signals and our mechatronic setup, with a focus on control loop delays. Then, section 5.2 describes the tuning procedures for gravity compensation and how we collected torque data to train the myoelectric decoders. At last, section 5.3 describes the set of experiments we devoted to compare myoelectric interfaces.

In figures 5.1 we can see the setup from various points of view. In particular, figure 5.1(d) shows how it is worn by a user. On the upper arm we can see the Myo™ armband by Thalmic Labs Inc.[1], used as sEMG sensor: the raw data are sampled at 200 Hz and then streamed via Bluetooth® at 100 Hz. The mechatronic components – starting from the furthest part – are: the DC brushless motor, the encoder on the motor, the torque sensor, the elastic link (a metal spring, optionally replaceable with a rigid link, as in 5.1(c)), the encoder on the environment and the support for the right forearm of the wearer. The control software is developed within the ROS framework and the control loop runs realtime at 5 kHz, using the Open EtherCAT Society's Simple Open Source EtherCAT Master (SOEM) C library[2] and the Series Elastic Library (SELib) C++ library[3].

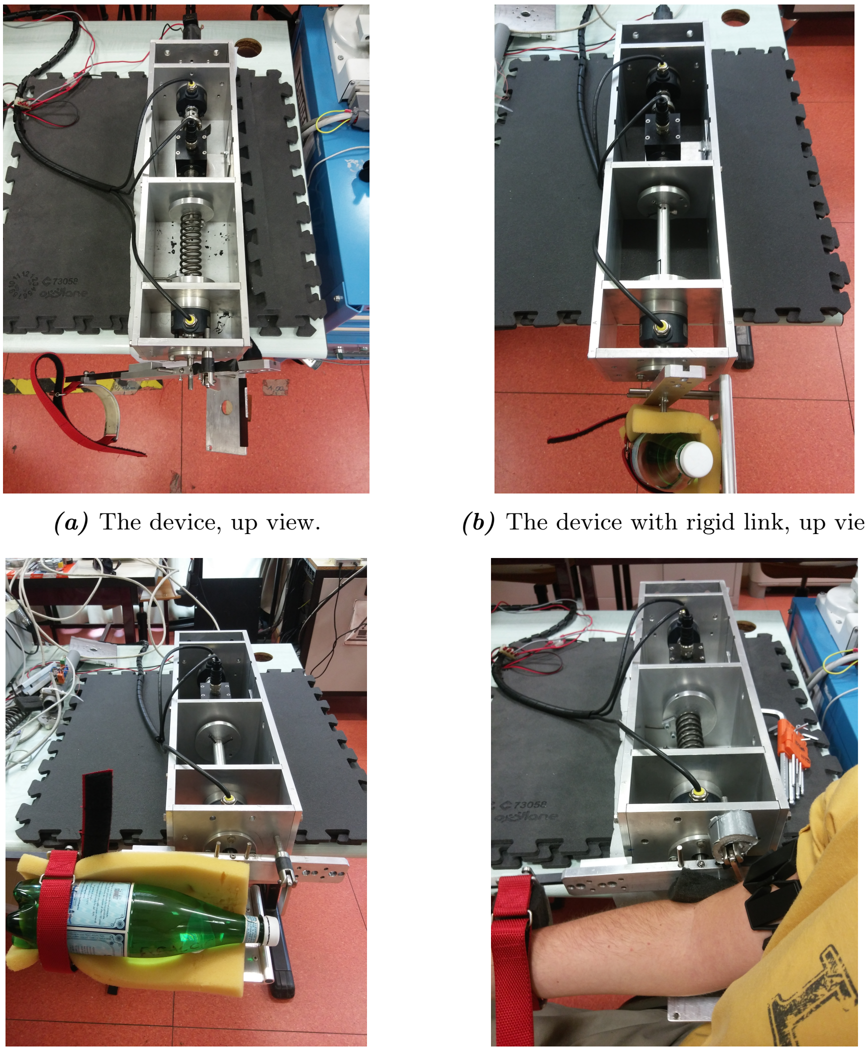
5.1 Analysis of delays

A set of preliminary tests have been performed with the main focus of analysing the delays induced by the various components acting on the control loop, starting from

[1] Thalmic Labs Inc., Myo™ armband https://www.myo.com/, link consulted on October 15th, 2018

[2] Open EtherCAT Society, Simple Open Source EtherCAT Master (SOEM) v1.3.1, GitHub repository https://github.com/OpenEtherCATsociety/SOEM

[3] ALTAIR Laboratory, Series Elastic Library – development version, SELibDev, GitLab repository https://gitlab.com/altairLab/elasticteam/SELibDev, based on Andrea Calanca's Series Elastic Library (SELib) v1.1, GitLab repository https://gitlab.com/calanca/SeriesElasticLibrary



(a) The device, up view.

(b) The device with rigid link, up view.

(c) The device with the sample bottle.

(d) The device worn by a user.

Figure 5.1: The device used for the experiments.

the myoelectric signal generation down to the physical interaction with the user. A general schema of an exoskeleton driven by myoelectric signals is reported in figure 5.2. Two control loops act on the system: the *natural* sensory-motor biological loop, commanded by the user's brain, and an *artificial* nested loop, commanded by the decoder, the adapter (not shown), the low level controller, and the orthosis.This loop aims at supporting the user intended motion, as ultimately it is commanded by the user's brain via EMG interface. Each closed loop system is affected by delays and, as a conservative guideline, we argue that an artificial control loop should not introduce delays that differ too much form ones on the biological loop, i.e. the *electromechanical* delay (EMD), that is the inherent delay between the myoelectric signal and the torque due to the muscle-tendon-articulation dynamics. The EMD in literature is reported up to 100 ms [10, 45] in healthy subjects, but it can be higher in motor impaired people. Analysis of the sEMG-torque data sets we collected, revealed a value between 180-200 ms.

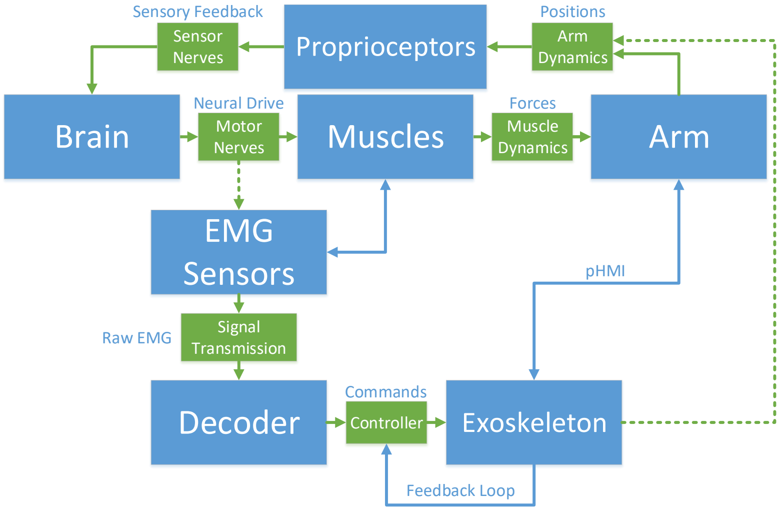


Figure 5.2: Schema of the control loops acting on a myoelectric-controlled arm orthesis: two nested loops are actually responsible for the arm-device system behaviour, an outer loop closed by the wearer's brain and an inner one closed by the low-level control algorithm. The green blocks are the components responsible for the loop delays.

Although the CNS is able to quickly adapt its behaviour in response to external perturbations [36] and constant delays, it has been shown that delays above 500 ms cause the lose of ownership perception about the limb or the device [55], as well as randomly variable delays. Therefore to be conservative, the maximum acceptable control delay for an artificial loop should be around 200 ms. As every component acts on the control and can introduce delays and phase lags, they cannot be designed separately but as part of a larger system. In our case, major delays are induced by (1) data transmission and (2) low level controller dynamics, while decoding phase (a part from delays introduced on purpose) has minimal impact.

5.1.1 Analysis of delays: Data transmission

The Myo™ armband specifications given by the manufacturer do not include details on AD-conversion and transmission delays, so they must be retrieved covered by experiments. Because of the Bluetooth Low Energy (BLE) transmission limitations, the sEMG signals are sampled consistently at 200 Hz but they are sent two by two at an effective frequency of 100 Hz [48].

A first experiment was performed to measure the latency that we found to be between one or two samples. It is not possible to know this delay in normal use since the samples are sent without timestamp.

A second experiment analysed the transmission jitter, computing the distribution of the time interval between two consecutive arrival on a trial lasting 124.02 s. We found it has a distinctive three peaks shape (figure 5.3(a)). The mean value of the delay between two consecutive samples is 10.025 ms, near to the expected 10 ms, but the variability is very high, with a standard deviation of 4.773 ms, as it is typical in transmission bursts. After some tests, we decided to regularize the stream by storing the last packet received and picking it up at a constant rate – a 1-sample buffer. Figure 5.3(b) shows the bar plot of the sEMG samples received and stored in the buffer between two consecutive data request. In conclusion the latency due to data transmission can be considered to be 10 ms due to the buffering, plus a variable amount of time between 10 ms to 20 ms due to AD-conversion and BLE transmission.

5.2 Preparatory experimental procedure

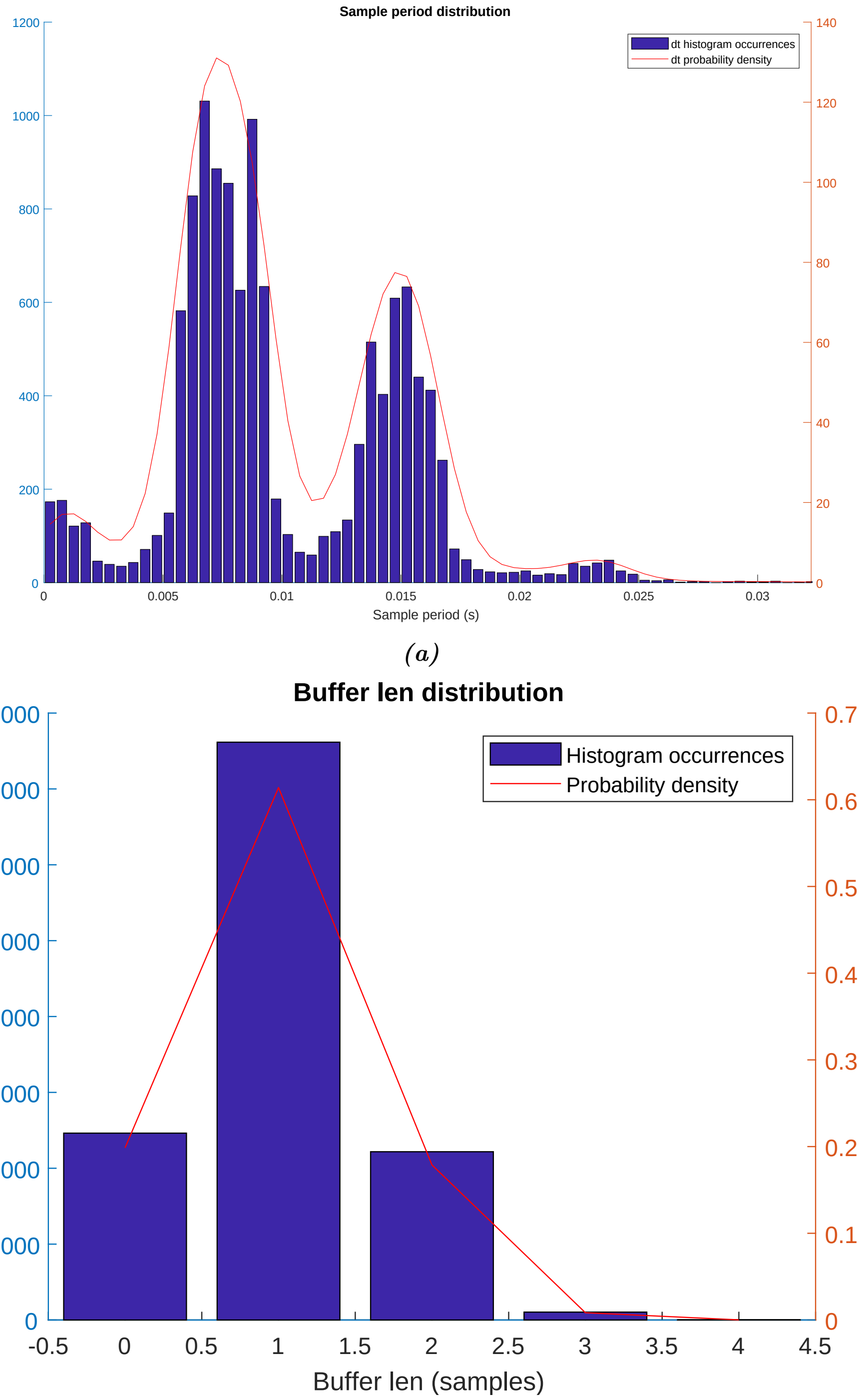
This section describes the preparatory experimental procedure we followed to set the myoelectric interfaces during experiments. It is the initial phase common to each experimental session since it is needed to tune subject specific parameters for controllers and decoders. We decided to perform the tuning procedures at each donning of the device, to guarantee fair and equally optimal conditions in each session. In particular, we noticed that even subtle changes in the wearer/device relative position can negatively affect the gravity compensation and little differences in electrodes placement can alter the EMG signals/torque relation: re-tuning minimize these effects. However, little misalignments can also happen within the same experimental session, plus fatigue and sweat change electrodes efficacy and decoder's performance, so experimental sessions were limited in time, no more than 3 hours, and always in the morning.

We divided the parameters into two sets: fixed parameters to be manually tuned once and session depending parameters, such as gravity compensation parameters for the controllers and regression coefficient for the decoders. The preparatory procedure allow to tune the session dependent parameters.

We divided the preparatory procedure into four phases: device donning (5.2.1), tuning of gravity compensation for controllers (5.2.2), tuning of regression coefficient for decoders (5.2.3), and selection of the gain between decoder and controller (5.2.4), that is the *adapter*. The gain is the only session dependent parameter which does not have an automatic procedure for tuning, but we will give an heuristic to find appropriate values.

5.2.1 Phase 1: Device donning

As one can see in figure 5.4(a), the device used for the experiments has two supports, the first for the forearm to be fasten with a strap and the second for the elbow, held in place by its own weight. The subjects should seat having the device on their right side to wear it on their right arm (figure 5.4(b)): it is also possible to reverse the setup to use the subject's left arm. It is important to adjust the seat and the bench so that the elbow flexion axis is aligned with the device revolute joint and the elbow and the shoulder do not change position across the entire wearer's range-of-motion



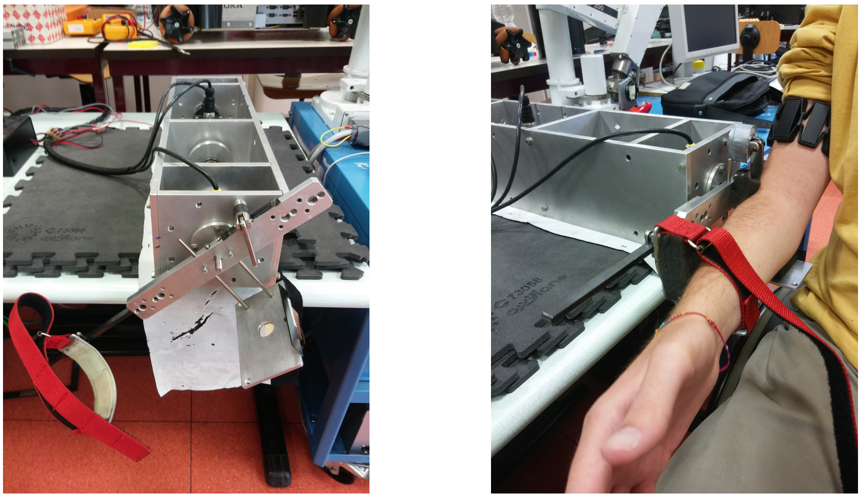
(a)

(b)

Figure 5.3: BLE communication analysis of the Myo™ (a) BLE transmission jitter of sEMG samples. (b) Distribution of sEMG samples buffer lengths.

(ROM). The ROM can be different from subject to subject, so we should take note of its extremes as measured by the relative encoder.

The Myo™ armband is worn *before* donning the device, placed halfway the upper arm right above the biceps belly and properly tightened to allow the electrodes pick up a good signal. For repeatability, we always placed the sensor with the charge plug-in externally and pointing towards the elbow.



(a) The device empty, front view.

(b) The device donned.

Figure 5.4: Device donning: (a) the device before donning; (b) the device worn by a subject, on the upper arm there is the Myo™ armband sensor.

5.2.2 Phase 2: Gravity calibration

We devised an automatic procedure to tune gravity compensation parameters. First, some data are collected to relate gravitational forces and positions. To this aim, a position controller moves the elbow along a predefined trajectory spanning the ROM a few times, while torques and positions are measured. The arm moves at slow constant velocity from down to up, inverts motion direction, then moves up to down at the same constant velocity, inverts motion again and so on (figure 5.5). The measures are stored only during the constant velocity parts, when there is no acceleration. We designed the trajectory to guarantee that the motion is inverted with a smooth transition from and to the constant velocity parts, with a continuous and derivable acceleration in each instant. This avoids abrupt movements that could lead to oscillations, while never exceeding ROM extremes and a maximum acceleration. Moreover, this fluid movement helps the wearer staying relaxed to minimize even small voluntary forces that cannot be modelled.

We also explored an alternative data collection procedure which measures static torques at specific positions, but we had some issues. We found out that the "stop-and-go" trajectory requires a settle time once reached the measure point, both to allow the wearer relax and to collect multiple torque data to filter out the noise. However, to have a good dataset there should be many measure point and this is time consuming: the constant velocity method is faster. Moreover, it makes easier to keep the arm relaxed.

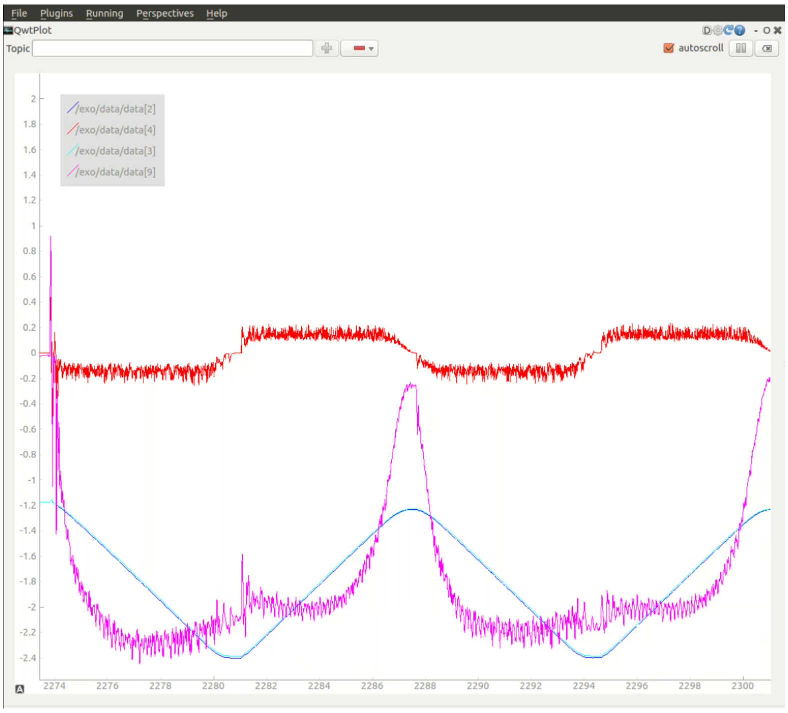


Figure 5.5: A `rqt_plot` showing the trajectory for gravity calibration: environment and motor encoder positions, environment velocity and torque measurement.

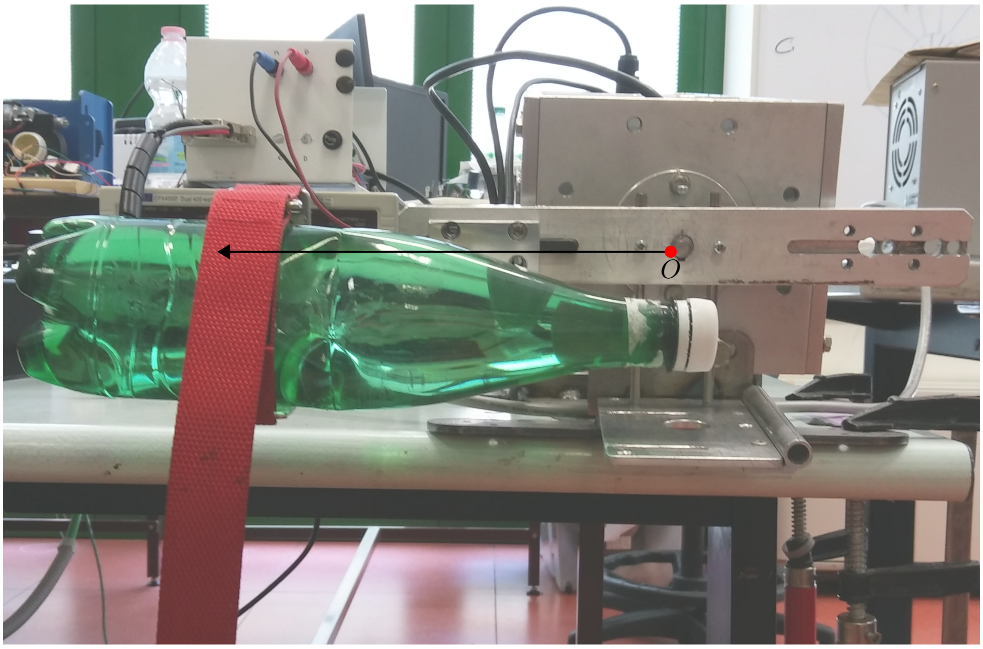


Figure 5.6: Device's reference system and gravity compensation: front view of the device with the sample bottle.

Assuming that the arm is completely relaxed and behaves like a pure mass (imagine it is the bottle in figure 5.6), that the elbow moves at constant velocity, and the motor is the only external force, when the velocity is constant there is no acceleration.Under this condition, the following sum of torques should be zero:

$$\tau_s + g(\theta_e) - d\,\mathrm{sign}(\dot{\theta}_e) = 0 \tag{5.1}$$

where τ_s is the torque read by the sensor and due to the motor, $g(\theta_e)$ is the torque due to the weight of the system as a force acting on the center-of-mass of the arm and device compound, and d is the dynamic friction that is always a torque opposite to the motion and we assume of equal intensity in both directions. With the measures of τ_s and θ_e, we can build a model to get the explicit formula of $g(\theta_e)$:

$$\tilde{\tau}_s(\theta_e) = -g(\theta_e) + d\,\mathrm{sign}(\dot{\theta}_e) = -mgb\cos(\theta_e - \mathit{off}) + d\,\mathrm{sign}(\dot{\theta}_e) \tag{5.2}$$

We can then estimate the value of model's parameters via a non-linear (because of the offset term *off*) regression algorithm, like Levenberg-Marquardt. We found that including the friction term in the torque reference for gravity compensation was useless, since the controller already compensate for it. Then, one can implement gravity compensation by computing the following torque reference for the low level controller:

$$\tau_{\mathit{ref}}(\theta_e) = -g(\theta_e) = -mgb\cos(\theta_e - \mathit{off}). \tag{5.3}$$

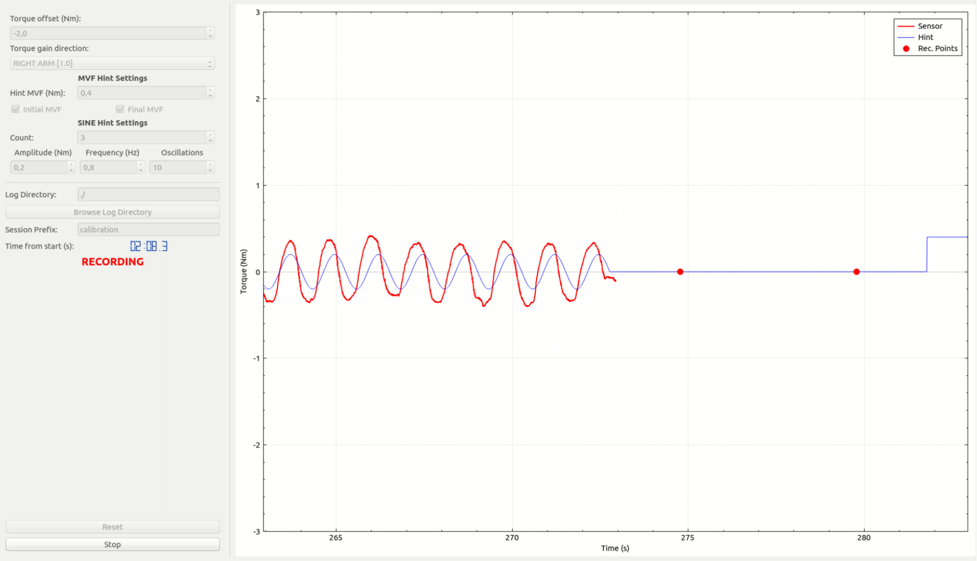
5.2.3 Phase 3: Decoders calibration

All the decoders are trained by fitting EMG signals to isometric torques. It is important to collect a good dataset to capture the actual underlying relationship. To measure the isometric torque, the device is fixed to a comfortable position, usually an intermediate point between full extension and full flexion, with a position controller and then the wearer is asked to contrast the controller trying to move up and down. By removing the weight from the torque measured by the sensor, what we obtain is the isometric force, opposite direction. After some trials, we found out that doing this movement with neither a visual feedback of the actual torque measured nor a guide, can result to a poor dataset and the quality of the regression can vary a lot. So we introduced a visual interface (figure 5.7) that helps collecting a good dataset by guiding the wearer to pursue a force profile with tunable sine waves, steps, and relax periods.

After recording the dataset, a series of MatLab™ scripts helps elaborating it to obtain the parameters of the decoders that are stored on a configuration file that is used when running the decoders.

5.2.4 Phase 4: Adapter's gain selection

At present, the only session dependent parameter that cannot be automatically tuned is the gain between decoder and controller. It is not a simple gain that has an unknown correct value and it is only a matter of finding it, that is the *adapter*, able to regulate not only the control performance, but also the level of assistance. Here



Torque offset (Nm): -2,0

Torque gain direction: RIGHT ARM [1.0]

MVF Hint Settings

Hint MVF (Nm): 0,4

☑ Initial MVF ☑ Final MVF

SINE Hint Settings

Count: 3

Amplitude (Nm)	Frequency (Hz)	Oscillations
0,2	0,8	10

Log Directory: ./

Session Prefix: calibration

Time from start (s): 02:08.3

RECORDING

Figure 5.7: Isometric torque dataset collection visual interface.

we propose a subjective heuristic to get one of the suitable gain values, tuning the interfaces at the same (subjective) performance level.

As a matter of fact, such gain acts as loop gain. Intuitively, as its value increases so the level of assistance: with an higher gain, lower myoelectric activity is needed. Unfortunately, a too high gain can cause system instability or make it uncontrollable, which can be intuitively motivated by the presence of a relatively high delay. So the objective is to increase the gain as much as possible while maintaining an acceptable level of controllability. Here we intend a naive concept of controllability, i.e. the ability for the subject to make the system moving as desired, without too much undesired oscillations.

Once decoders and controllers are tuned, we perform a first trial with gravity compensation without myoelectric control. Then we select a decoder/controller pair. We try the system with a low value for the *adapter* gain, typically 1.0, then we progressively increase its value until the effort required to maintain a fixed position is no more acceptable then we slightly reduce its value to make this task comfortable. The behavior we expect should be similar to the gravity compensation, but with much lower effort.

The selection of this *adapter* gain value is an unexplored research topic that is linked to the definition and the selection of the appropriate level of assistance. However, this gain affects also other factors, such as domain conversion (e.g. decoder tuned on torque to velocity controller reference) and device controllability. So we decided to study the overall impact of the gain value for different interfaces, see subsection 5.3.4.

5.3 Experiments description

This section explains how we collected the reaching task data and how the experiments have been conducted. After the preparatory phases described in section 5.2, depending on the specific question we want to answer with the experiment, the combination of task settings and myoelectric interface type will vary. A *performance* *evaluation interface* is used to guide the subject in a reaching task while collecting performance measures, see 5.3.1. Quantitative metrics are considered to compare different combination of myoelectric control architecture components on their ability to follow the subject motion intention.

Each experiment is described by:

- A scientific objective, which defines the question the experiment is going to answer to.
- The set of *myoelectric control architecture* components combinations to be compared.
- Specific task parameters: the reaching distance, the target width, and the number of repetitions.

In this work the experiments are limited to one subject and we do not know if the answer we found are generalizable. The analysis of first experiments' results gave us important hints on how to conduct the following ones, so there are improvements and differences during the experiments' progression.

5.3.1 Performance evaluation interface

The *performance evaluation interface* (PEI) provides the subjects with a visual feedback on the reaching task to be completed. At the same time, the PEI collects data and performance measures according to the Fitts' law [61]. The PEI is implemented as a ROS node – called `myo_interface` – which interacts with the device. The PEI is represented in figure 5.8. The control panel on the left allows to set task's parameters and to run the task. Current performance measures and status are also displayed. On the right there is a square plot with fixed aspect ratio showing the current position of the subject's arm represented as a rotating stick, where the center correspond to the elbow and the longer tip to the hand. The target position is represented as a square box on hand's trajectory. Since the device has only 1 DOF, the only relevant information is the stick angle and the angle at which it encounters the target. Note that this interface is the only mean by which we define the task, the subjects do not have any physical target to reach.

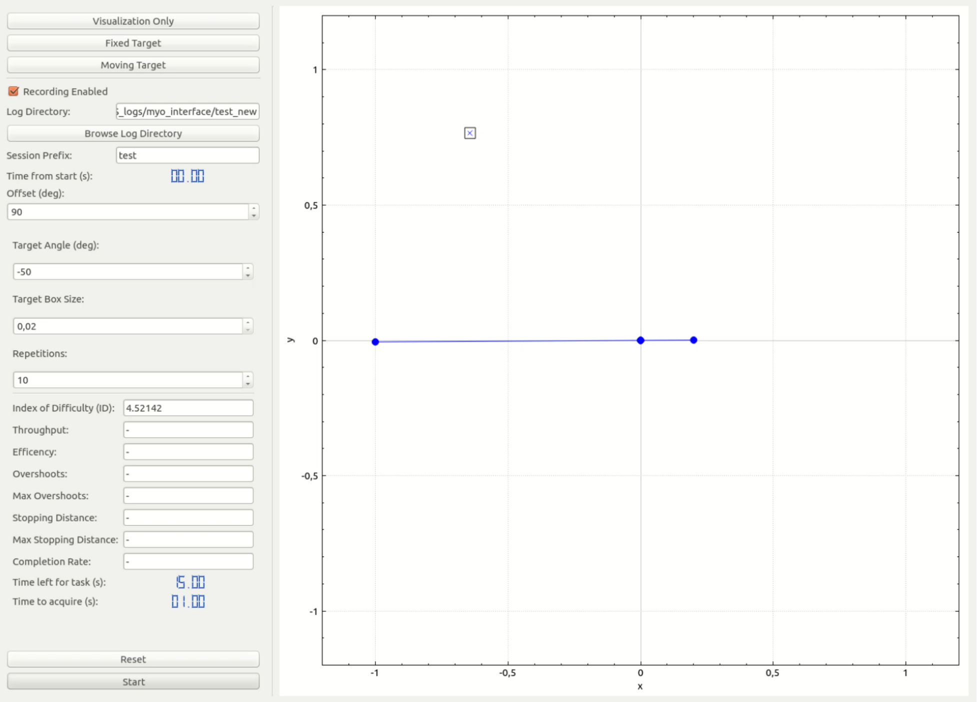
Task parameters The interface allows to set some parameters that affects both the task definition and visualization:

Offset parameter (degrees) compensates the elbow encoder offset. It has no meaning for the task but it is useful to align the virtual arm representation with the physical arm.

Target Angle (degrees) is the position of the first target as elbow angle. The point corresponding to the target angle on the virtual arm tip's trajectory is shown as a cross in the target box center.

Target Box Size (or width – plot axes unit) is the size of the target box. The size of the box edge is two times this value. The target is considered reached if the distance between the target center and the arm's tip is less then this value. The boundary is actually checked on the arc of the virtual arm tip's trajectory that has the target point at its center. So this parameter sets half the length of that arc, simplifying the check if the target area is reached to a simple interval test on current elbow angle.

Repetitions are the number of consecutive reaching tasks that will be proposed. They must be an even number since each forward reaching involves a backward reaching (that is evaluated) to return to the starting point.



☑ Recording Enabled

Log Directory: s_logs/myo_interface/test_new

Session Prefix: test

Time from start (s): 00.00

Offset (deg): 90

Target Angle (deg): -50

Target Box Size: 0,02

Repetitions: 10

Index of Difficulty (ID): 4.52142

Throughput: -

Efficency: -

Overshoots: -

Max Overshoots: -

Stopping Distance: -

Max Stopping Distance: -

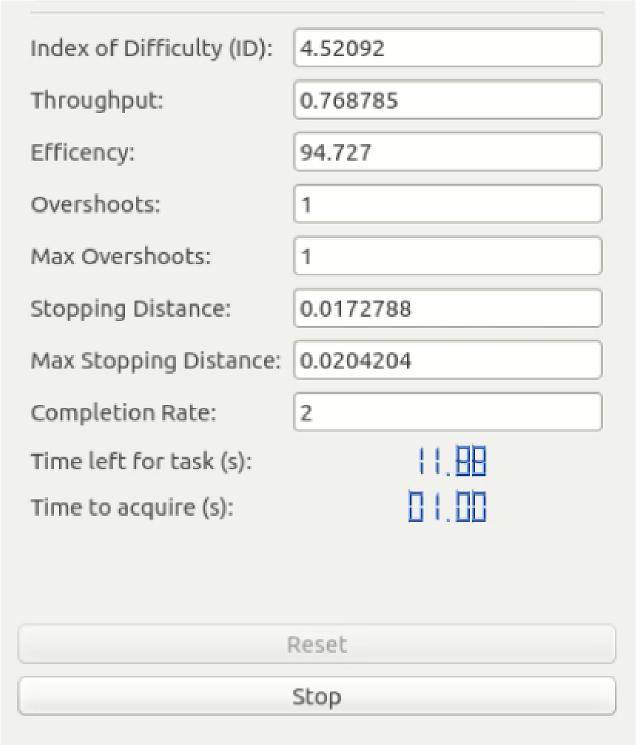
Completion Rate: -

Time left for task (s): 15.00

Time to acquire (s): 01.00

Figure 5.8: Performance evaluation interface. This is the appearance before starting the task: the subject is instructed to reach the target highlighted by a square then return to the starting point a certain number of times with the actual flexion of the elbow visualized as the blue stick orientation (considering the elbow and the hand represented respectively by the center and the tip of it).

Performance measures as described in 4.2. These include *throughput* – higher is better – considered as the velocity of task completion, *efficiency* – higher is better, *overshoot* and *target path* (or *scaled target path*) – lower is better for the last two – quantities related to the precision, but at different scales, and *IEmg* (or *mean IEmg*) – lower is better – that measure the (mean) muscular effort.



Index of Difficulty (ID): 4.52092

Throughput: 0.768785

Efficency: 94.727

Overshoots: 1

Max Overshoots: 1

Stopping Distance: 0.0172788

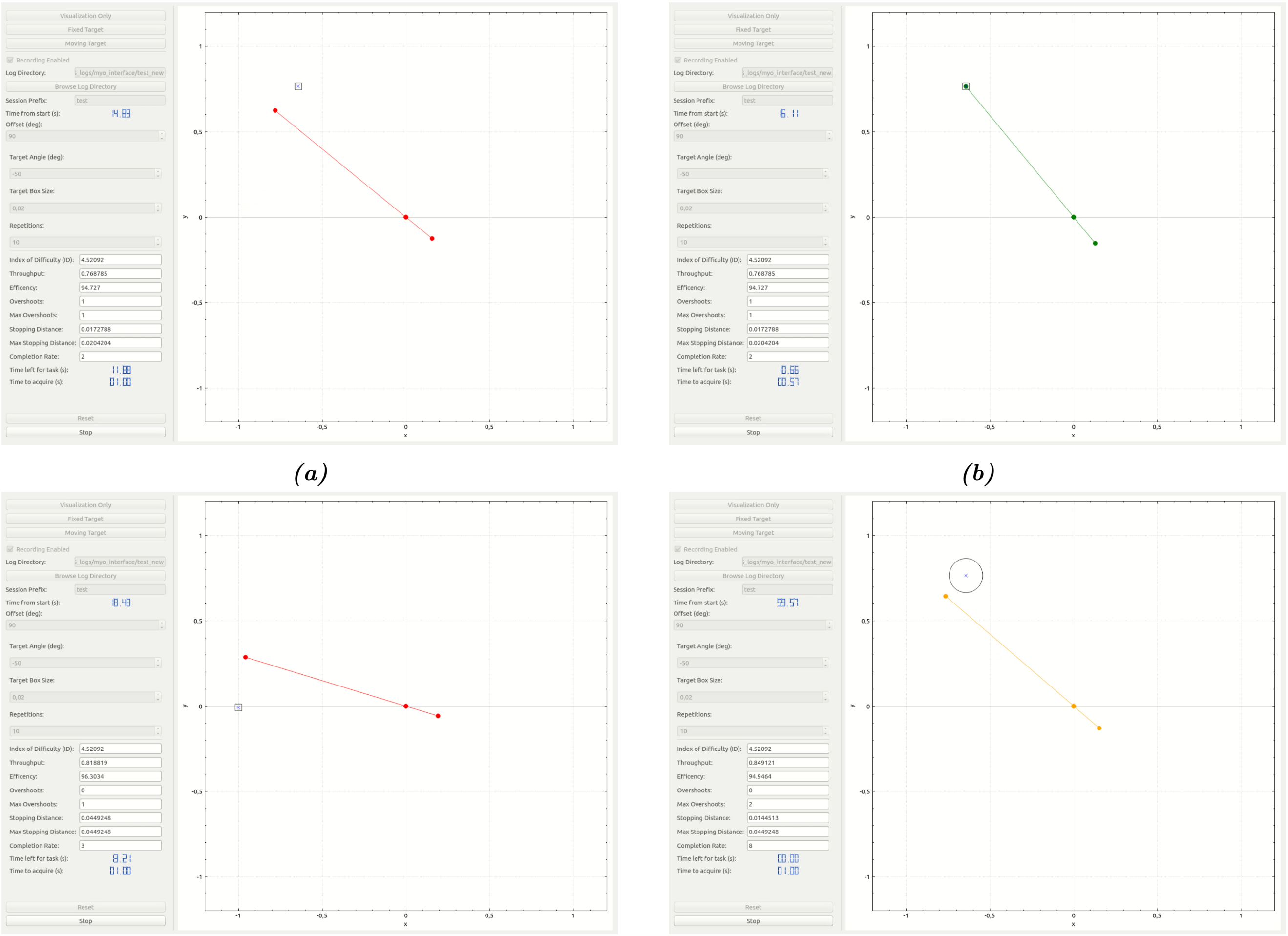
Max Stopping Distance: 0.0204204

Completion Rate: 2

Time left for task (s): 11.88

Time to acquire (s): 01.00

Figure 5.9: Performance measures detail.



☑ Recording Enabled
Log Directory: s_logs/myo_interface/test_new
Session Prefix: test
Time from start (s): 14.89
Offset (deg): 90
Target Angle (deg): -50
Target Box Size: 0,02
Repetitions: 10
Index of Difficulty (ID): 4.52092
Throughput: 0.768785
Efficency: 94.727
Overshoots: 1
Max Overshoots: 1
Stopping Distance: 0.0172788
Max Stopping Distance: 0.0204204
Completion Rate: 2
Time left for task (s): 11.88
Time to acquire (s): 01.00

(a)

☑ Recording Enabled
Log Directory: s_logs/myo_interface/test_new
Session Prefix: test
Time from start (s): 16.11
Offset (deg): 90
Target Angle (deg): -50
Target Box Size: 0,02
Repetitions: 10
Index of Difficulty (ID): 4.52092
Throughput: 0.768785
Efficency: 94.727
Overshoots: 1
Max Overshoots: 1
Stopping Distance: 0.0172788
Max Stopping Distance: 0.0204204
Completion Rate: 2
Time left for task (s): 10.66
Time to acquire (s): 00.57

(b)

☑ Recording Enabled
Log Directory: s_logs/myo_interface/test_new
Session Prefix: test
Time from start (s): 18.48
Offset (deg): 90
Target Angle (deg): -50
Target Box Size: 0,02
Repetitions: 10
Index of Difficulty (ID): 4.52092
Throughput: 0.818819
Efficency: 96.3034
Overshoots: 0
Max Overshoots: 1
Stopping Distance: 0.0449248
Max Stopping Distance: 0.0449248
Completion Rate: 3
Time left for task (s): 13.21
Time to acquire (s): 01.00

(c)

☑ Recording Enabled
Log Directory: s_logs/myo_interface/test_new
Session Prefix: test
Time from start (s): 59.57
Offset (deg): 90
Target Angle (deg): -50
Target Box Size: 0,02
Repetitions: 10
Index of Difficulty (ID): 4.52092
Throughput: 0.849121
Efficency: 94.9464
Overshoots: 0
Max Overshoots: 2
Stopping Distance: 0.0144513
Max Stopping Distance: 0.0449248
Completion Rate: 8
Time left for task (s): 00.00
Time to acquire (s): 01.00

(d)

Figure 5.10: Evolution of the performance evaluation interface as the reaching task advances. (a) Reaching target with subject moving forward (up, flexion). (b) Target reached, waiting the dwell time: to highlight that subject's arm in within the target area, the stick changes from red to green. (c) Reaching target with subject moving backward (down, extension). (d) Failed in reaching the target within the maximum time: the stick becomes yellow and the target a circle to be reached without time limits.

5.3.2 Experimental session 0

On 2019/02/07

The first experiment was primary designed to test the system and the evaluation interface. The related scientific questions are:

1. is the reaching task with the parameters as in table 5.1, able to show differences between the various combinations of myoelectric control architecture components?
2. if the answer is yes, is there a myoelectric interface showing – in this experiment – a significantly better behaviour w.r.t. the others or gravity compensation?
3. is there a component that affects more the performance w.r.t. the other between controller and decoder?

The set of myoelectric control architecture components combinations tested is given by the combination of the following decoders (1) `PLAIN_P` – plain proportional, (2) `MULTICH2` – multichannel with 2, and (3) `MULTICH8` – 8 EMG channels and controllers (1) `COMP` – gravity compensation, (2) `FORCE` – force, (3) `FORCE_INT` – force integral, (4) `IMP` – impedance, and (5) `VEL` – velocity. The *sequential* plan of the experimental session is summarized in table 5.2. For this experiment the PEI evaluates only the forward path, so for each repetition there is a forward path evaluated and a backward not evaluated.

Start Angle	Target Angle	Target Box Size	Repetitions
$-70°$	$-90°$	0.06	20

Table 5.1: Task parameters for experimental session 0.

5.3.3 Experimental session 1

On 2019/09/06

Given the lesson learnt from the previous experiment and the questions left unanswered, we were able to define an experimental plan with objectives higher than a proof-of-concept. The scientific questions are almost the same:

1. is the reaching task with the parameters as in table 5.3, able to show differences between the various combinations of myoelectric control architecture components?
2. is the behaviour of the interface based on gravity compensation alone consistent in time w.r.t. the considered performance measures? In other words, can we consider it as a *baseline* interface?
3. if the answer to 2 is yes, is there a combination of myoelectric control architecture components showing – in this experiment – a significantly better behaviour w.r.t. the others or gravity compensation?
4. considering that each architecture differs from the others because composed by a different controller and a different decoder module, is there a component affecting more the performance?

Controller	Decoder	Adapter's gain
COMP	NONE	–
FORCE	PLAIN_P	2.5
FORCE	MULTICH2	3.0
FORCE	MULTICH8	3.0
FORCE_INT	PLAIN_P	2.5
FORCE_INT	MULTICH2	4.0
FORCE_INT	MULTICH8	4.0
IMP	PLAIN_P	3.5
IMP	MULTICH2	4.0
IMP	MULTICH8	4.0
VEL	PLAIN_P	3.5
VEL	MULTICH2	4.0
VEL	MULTICH8	4.0

Table 5.2: Summary table of experimental session 0. We tested the combinations of 5 controllers, including gravity compensation (COMP), with 3 decoders grouped in 5 sequential blocks. The box size and the target angle were fixed, respectively, to 0.06 and 20° above starting angle (see table 5.1). Each row involved 20 repetition divided into 2 tasks of 10 forward reaching each.

5. are time and fatigue affecting the performance? Repeated gravity compensation tasks at different stages of the experiment allow to check the evolution over time of the performance.

The set of myoelectric control architecture components combinations tested is given by the combination of the following decoders (1) PLAIN_P – plain proportional, and (2) MULTICH8 – multichannel with 8 EMG channels and controllers (1) COMP – gravity compensation, (2) FORCE – force, (3) FORCE_INT – force integral, (4) IMP – impedance, (5) VEL – velocity, and (6) ADM – admittance. The *sequential* plan of the experimental session is summarized in table 5.4.

What we changed with respect to experimental session 0, is that we introduce the admittance controller which was not tested before and removed the multichannel decoder with 2 EMG channels since it was too similar to the plain proportional, allowing to perform more tests within the same experimental session. We point out that the admittance controller can be viewed as a velocity controller with the admittance model used as an *adapter*, instead of a gain.

Also, we decided to evaluate both the forward and the backward path, allowing to collect twice the data points in the same amount of time. Finally, as indicated in table 5.3, we left the box size unchanged w.r.t. experimental session 0, but tried two different target positions one about twice the distance of the other, so with different *index of difficulty*.

Configuration	Start Angle	Target Angle	Target Box Size	Repetitions
POS1	$-30°$	$-65°$	0.06	20
POS2	$-30°$	$-95°$	0.06	20

Table 5.3: Task parameters for experimental session 1.

Tag	Controller	Decoder	Adapter's parameter set		Task's configuration
COMPO	COMP	NONE	–		POS1
	COMP	NONE	–		POS2
	COMP	NONE	–		POS1
	COMP	NONE	–		POS2
FORCE			gain		
	COMP	NONE	–		POS1
	COMP	NONE	–		POS2
	FORCE	PLAIN_P	3.0		POS1
	FORCE	PLAIN_P	3.0		POS2
	FORCE	MULTICH8	3.0		POS1
	FORCE	MULTICH8	3.0		POS2
FORCE_INT			gain		
	COMP	NONE	–		POS1
	COMP	NONE	–		POS2
	FORCE_INT	PLAIN_P	4.0		POS1
	FORCE_INT	PLAIN_P	4.0		POS2
	FORCE_INT	MULTICH8	4.0		POS1
	FORCE_INT	MULTICH8	4.0		POS2
IMP			gain		
	COMP	NONE	–		POS1
	COMP	NONE	–		POS2
	IMP	PLAIN_P	4.0		POS1
	IMP	PLAIN_P	4.0		POS2
	IMP	MULTICH8	4.0		POS1
	IMP	MULTICH8	4.0		POS2
VEL			gain		
	COMP	NONE	–		POS1
	COMP	NONE	–		POS2
	VEL	PLAIN_P	4.0		POS1
	VEL	PLAIN_P	4.0		POS2
	VEL	MULTICH8	4.0		POS1
	VEL	MULTICH8	4.0		POS2
ADM			mass	damp	
	COMP	NONE	–	–	POS1
	COMP	NONE	–	–	POS2
	ADM	PLAIN_P	0.3	0.2	POS1
	ADM	PLAIN_P	0.3	0.2	POS2
	ADM	MULTICH8	0.3	0.2	POS1
	ADM	MULTICH8	0.3	0.2	POS2

Table 5.4: Summary table of experimental session 1. We tested the combinations of 6 controllers, including gravity compensation (COMP), with 2 decoders grouped in 6 sequential blocks. The box size was fixed to 0.06, but there were two kind of task with two different target angles (POS1 and POS2, see table 5.3). The reaching tasks start at the same angle, but the distance of the second target angle is twice that of the first. Each row involved 20 repetition, 10 forward and 10 backward.

5.3.4 Experimental sessions 2 to 6

On 2019/10/09-10-18 and 2020/06/26-30

The experimental session 2 and the subsequent ones are a more extensive campaign with multiple objectives. We wanted to answer to more detailed scientific questions, that were:

1. is there a combination of myoelectric control architecture components, or parameter sets, showing a significantly better behaviour w.r.t. the others or gravity compensation?
2. considering that each architecture differs from the others because composed by a different controller, a different decoder module, and different adapter's parameters, is there a component affecting more significantly the performance?
3. is the tuning procedure we proposed able to find a value for the decoders' parameters leading to performance not worse than a manual tuning?
4. how much the assistance level affects the performance? Does this effect depends on the type of controller or decoder?

The set of myoelectric control architecture components combinations tested is given by the following decoders types with different parameter sets: `PLAIN_P` with 3 different parameter sets (1) `PLAIN_P_FIT` – parameters fitted to predict isometric force, (2) `PLAIN_P_HALF_E` – as 1 but halved *extension* gain, (3) `PLAIN_P_HALF_F` – as 1 but halved *flexion* gain, and (4) `MULTICH8` – multichannel with 8 EMG channels. They were combined with controllers (1) `COMP` – gravity compensation, (2) `FORCE` – force, (3) `VEL` – velocity, (4) `IMP` – impedance, (5) `ADM` – admittance, and (6) `FORCE_INT` – force integral. In this experiment we also played with the *adapter*'s parameter, defining 3 different increasing assistance levels (`LEV1`, `LEV2`, and `LEV3`). The actual values of the parameters, different for each controller, are reported in table 5.7. This quantity is called assistance level because as it increases the decoder's signal amplification increases. The values are different between controllers because they have different meanings depending on the controller adopted. In order to allow a comparison between levels, they are tuned to have similar effect: the middle level – `LEV2` – is selected to be the subjective best for the user, while `LEV1` is about a half below it and `LE3` about a half above. We also changed also the task configurations, trying two task with different *index of difficulty* by setting the target box size to 0.04 and 0.02 (table 5.5), but same target position – about 30° above start angle (table 5.6).

The complete experimental session including all the combinations of decoders, controllers and assistance levels would have last too much, so we decided to break down the experiment into 5 sessions, each limited to test a single type of controller. We repeated the test on gravity compensation in each session, to verify that the subject conditions are similar across the different days. The *sequential* plan of one experimental session, chosen the controller type `COMP`, is summarized in table 5.8.

Configuration	Target Box Size	Repetitions
`BOX1`	0.04	20
`BOX2`	0.02	20

Table 5.5: Task parameters for experimental sessions 2 to 6 – configurations.

Controller	Start Angle	Target Angle
FORCE	$-70°$	$-100°$
VEL	$-75°$	$-105°$
IMP	$-90°$	$-119°$
ADM	$-75°$	$-120°$
FORCE_INT	$-75°$	$-120°$

Table 5.6: Task parameters for experimental sessions 2 to 6 – controllers.

	Controller					
Assistance Level	FORCE	VEL	IMP	ADM		FORCE_INT
	gain	gain	gain	mass	dump	gain
LEV1	1.5	3.0	2.0	0.25	1.25	2.0
LEV2	3.0	6.0	4.0	0.10	0.50	4.0
LEV3	4.0	8.0	5.0	0.05	0.25	6.0

Table 5.7: Task parameters for experimental session 1.

After the 5 experimental sessions, the collected data are rearranged to perform different "virtual experimental sessions" in order to answer to specific scientific questions.

To answer question 1, we considered only the intermediate gain value LEV2, in order to compare architectures in a fair scenario, since this value correspond to the subjective best configuration.

To answer question 3, we introduced the decoders PLAIN_P_HALF_E and PLAIN_P_HALF_F as possible results of manual tuning procedures, that to be truly different from PLAIN_P, not only a scaling, have respectively extension (E) and flexion (F) channel gains (about) half the value of automatically tuned decoder.

5.3.5 Statistical methodology

The scientific questions posed in the experiments were answered performing significance statistical tests based on ANOVA. Unfortunately, due to the pandemic outbreak, we had to limit our experiments to one disabled male subject, 29 years old, suffering from muscular dystrophy who in full possession of all his faculties accepted to go through the experimental procedures. Because of the limited amount of data at our disposal, some results could be partial or poorly significative.

The data selected for the statistical test were chosen to guarantee the same number of repetitions among groups. It is firstly tested for normality with Kolmogorov-Smirnov, Anderson-Darling and Shapiro-Wilk tests with significance threshold $\alpha = 0.05$: if all tests passed, we used parametric ANOVA, while if at least one normality test failed and we were not able to adequately transform the data to meet requirements, we employed also non-parametric Kruskal-Wallis test that operates on the rank of samples about the measure instead of plain data. The rank is computed globally giving higher rank to bigger values, so when groups' distributions are similar, a group with higher mean rank has bigger median. For discrete measures, only non-parametric analysis can be adopted. For both analyses, we used a significance threshold of $p < \alpha = 0.05$.

TAG	Controller	Decoder	Assistance Level	Task's configuration
COMP*	COMP	NONE	NONE	BOX1 BOX2
PLAIN_P_FIT	CTRL	PLAIN_P	LEV2	BOX1 BOX2
	CTRL	PLAIN_P	LEV1	BOX1 BOX2
	CTRL	PLAIN_P	LEV3	BOX1 BOX2
PLAIN_P_HALF_E	CTRL	PLAIN_P	LEV2	BOX1 BOX2
	CTRL	PLAIN_P	LEV1	BOX1 BOX2
	CTRL	PLAIN_P	LEV3	BOX1 BOX2
PLAIN_P_HALF_F	CTRL	PLAIN_P	LEV2	BOX1 BOX2
	CTRL	PLAIN_P	LEV1	BOX1 BOX2
	CTRL	PLAIN_P	LEV3	BOX1 BOX2
MULTICH8	CTRL	MULTICH8	LEV2	BOX1 BOX2
	CTRL	MULTICH8	LEV1	BOX1 BOX2
	CTRL	MULTICH8	LEV3	BOX1 BOX2

Table 5.8: Summary table of experimental sessions 2 to 6. In each session we tested a different controller (CTRL): FORCE in 2, VEL in 3, IMP in 4, ADM in 5, and FORCE_INT in 6. Gravity compensation (COMP) was tested in every session, with tag from COMP0 to COMP4. The 5 controllers were tested in combination with 4 decoders and 3 assistance levels. Each session was grouped in 5 blocks. As reported in tables 5.5 and 5.6, we tested tasks with two target box sizes, of 0.04 (BOX1) and 0.02 (BOX2), but with one target angle (POS1), fixed for the duration of each session. Each row involved 20 repetition, 10 forward and 10 backward.

CHAPTER 6

Results and discussion

This chapter presents the results and statistical analysis of the experiments described in section 5.3. We fixed the significance level for the statistical tests we performed to 95%, leading to an upper bound for the p-value of $p < \alpha = 0.05$.

6.1 On data visualization

Box-plots, such those in figure 6.5, report much information about data depicted. For each group there is a box such that:

- the horizontal line within each represents the group median;
- the top and bottom edges are traced in correspondence to the upper and lower quartiles, respectively. The distance between them is the inter-quartile range (IQR). Half of the values fall within this range from the median.
- the single points, displayed using an 'o' symbol, are *outliers*. These are values that are more than $1.5 \cdot IQR$ away from the top or bottom of the box,
- the dashed lines that extend above and below each box are called *whiskers*. They connect, respectively, the upper quartile to the non-outlier maximum (the maximum value that is not an outlier), and the lower quartile to the non-outlier minimum (the minimum value that is not an outlier).
- the tapered, shaded region around the median is called *notch*. Notches help to compare sample medians across multiple groups. Groups whose notches do not overlap have different medians at the 5% significance level, if the underlying distribution is normal. For other distributions, still the median comparison is reasonably robust.

When reporting statistical test results – e.g. figure 6.9 – the red points connected by continuous lines, superimposed to the boxes, are the groups' means with the corresponding 95% confidence interval (based on normal distribution assumption).

6.2 Experiment number 0

The first experiment was fundamental to refine some aspects of the experimental procedure, so its results are partial. The measures we considered are *throughput*, *efficiency*, *scaled target path*, *overshoots*, *IEmg* and *mean IEmg*.

6.2.1 Pre-analysis

One and two-way parametric ANOVA require strong hypotheses on the data, as described in the following. If these cannot verified, *non-parametric* analyses are considered, at the price of less statistical power. One of these is *Kruskal-Wallis* one-way ANOVA that operates on ranks instead of plain data. The first hypothesis is that data are continuous and normally distributed. This leads us to exclude *overshoots* measure, which is discrete. Normality has been tested using three methodologies: Kolmogorov-Smirnov, Anderson-Darling and Shapiro-Wilk tests. In all the cases, the *null* hypothesis is that data come from a normal distribution. If rejected there is a significative evidence that data come from a distribution that is *not* normal. A further check was done visually with quantile-quantile, CDF, and histogram plots.

The measures' summary statistics are reported in table 6.1, while table 6.2 shows normality tests' results and figures 6.1, 6.2, 6.3, and 6.4 the plots for visual inspection. We report test results and plots, both for the original data and the transformed ones, which we used to "normalize" certain measures.

A second hypothesis that should hold is homogeneity of variance among samples (or groups), also called homoscedasticity. ANOVA is generally considered robust to violations of this hypothesis, but it is a good practice to check it. We used Levene's test combined with visual inspection of a box plot of the data grouped by:

1. the combinations of controllers and decoders (see table 5.2), for a total of 13 categories or levels (*mix* or `CTRL+DEC`);
2. the controller (`CTRL`), 5 levels;
3. the decoder (`DEC`), 4 levels.

The *null* hypothesis is that the variance of data among groups are equal, so if rejected there is a significative evidence that at least one group has a variance different from the others. The test results are reported in table 6.3 and box-plots are showed in figures 6.5, 6.6, 6.7, and 6.8. Visual inspection is done by comparing the box heights – proportional to group variance – and H_0 is rejected when there is great difference between groups' box heights. One can notice that, for all grouping strategies, when H_0 is rejected – ANOVA assumptions are violated – gravity compensation (controller `COMP` and decoder `NONE`) has the shorter box. So we performed Levene's test again, this time excluding gravity compensation to see if some test decision changes (table 6.4).

Measure	mean	mean 95% CI		std
		lower	upper	
throughput	1.005	0.975	1.034	0.243
throughput_t	2.810	2.730	2.889	0.652
efficiency	56.284	54.143	58.425	17.532
scaled_target_path	1.314	1.213	1.415	0.826
scaled_target_path_t	0.355	0.294	0.417	0.503
iemg	61.459	59.040	63.879	19.815
iemg_t	4.071	4.033	4.108	0.308
mean_iemg	27.637	26.912	28.362	5.938

Table 6.1: Pre-analysis of data from experiment number 0, summary statistics.

Measure	Kolmogorov-Smirnov		Anderson-Darling		Shapiro-Wilk	
	H_0 rejected?	p	H_0 rejected?	p	H_0 rejected?	p
throughput	false	0.082	true	0.001	true	0.001
throughput_t	false	0.746	false	0.177	false	0.083
efficiency	false	0.472	false	0.071	false	0.146
scaled_target_path	true	0.003	true	0.001	true	<0.001
scaled_target_path_t	false	0.956	false	0.767	false	0.763
iemg	true	0.037	true	0.001	true	<0.001
iemg_t	false	0.809	false	0.165	false	0.177
mean_iemg	false	0.994	false	0.933	false	0.897

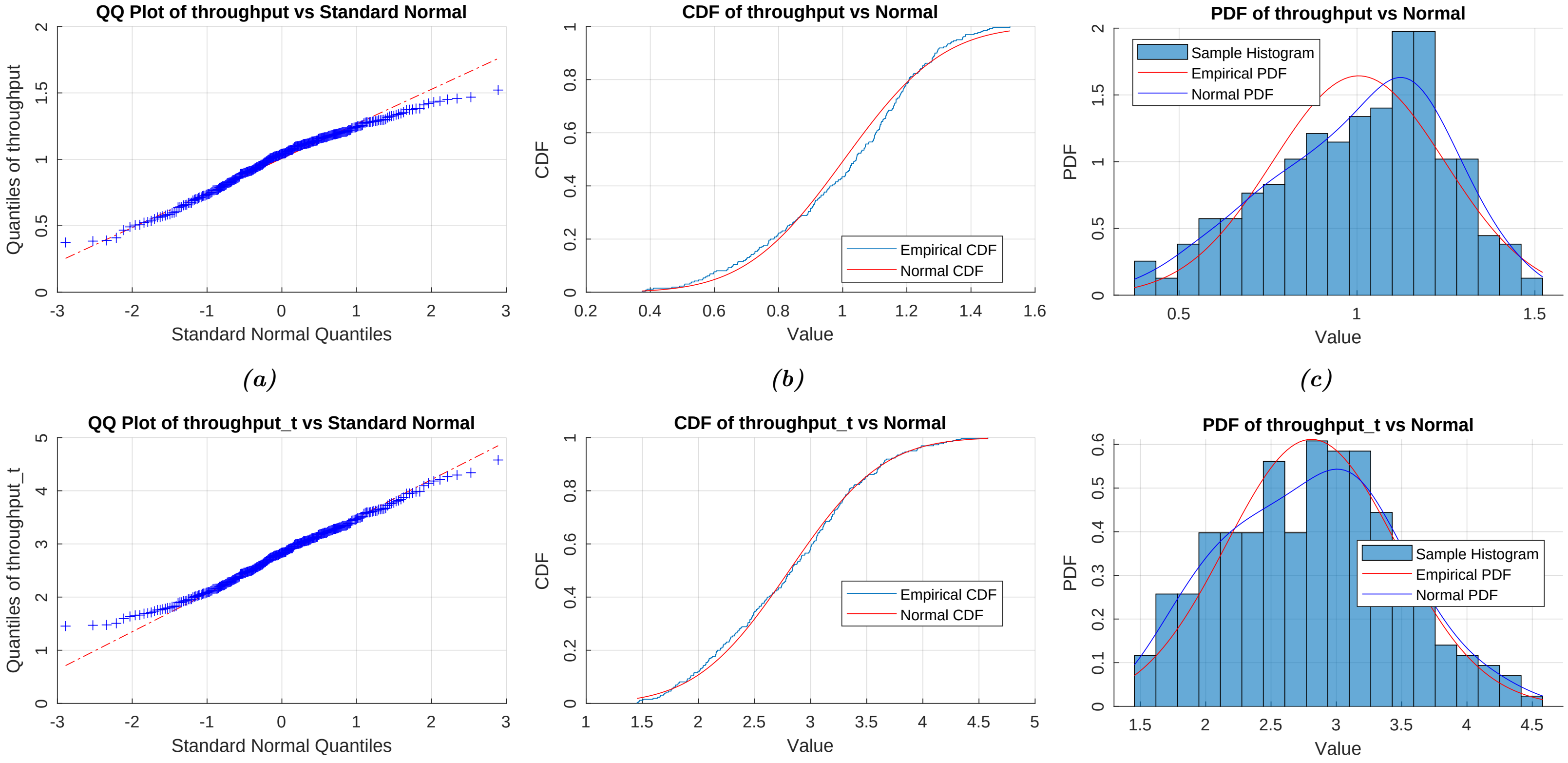
Table 6.2: Pre-analysis of data from experiment number 0, normality tests.

Measure	Levene's test (mix)		Levene's test (controller)		Levene's test (decoder)	
	H_0 rejected?	p	H_0 rejected?	p	H_0 rejected?	p
throughput	true	0.003	true	<0.001	false	0.403
throughput_t	true	<0.001	true	<0.001	false	0.251
efficiency	true	0.009	true	<0.001	true	<0.001
scaled_target_path	true	0.001	true	0.001	true	<0.001
scaled_target_path_t	true	0.001	true	0.002	true	<0.001
iemg	true	0.027	false	0.070	false	0.142
iemg_t	true	0.031	true	0.002	false	0.675
mean_iemg	false	0.291	false	0.478	false	0.770

Table 6.3: Pre-analysis of data from experiment number 0, Levene's test (with COMP).

Measure	Levene's test (mix)		Levene's test (controller)		Levene's test (decoder)	
	H_0 rejected?	p	H_0 rejected?	p	H_0 rejected?	p
throughput	true	0.003	true	<0.001	false	0.668
throughput_t	true	<0.001	true	<0.001	false	0.168
efficiency	false	0.199	true	0.035	false	0.122
scaled_target_path	true	0.038	false	0.080	true	0.009
scaled_target_path_t	true	0.017	true	0.049	true	0.006
iemg	true	0.025	false	0.052	false	0.113
iemg_t	true	0.023	true	0.001	false	0.635
mean_iemg	false	0.410	false	0.742	false	0.789

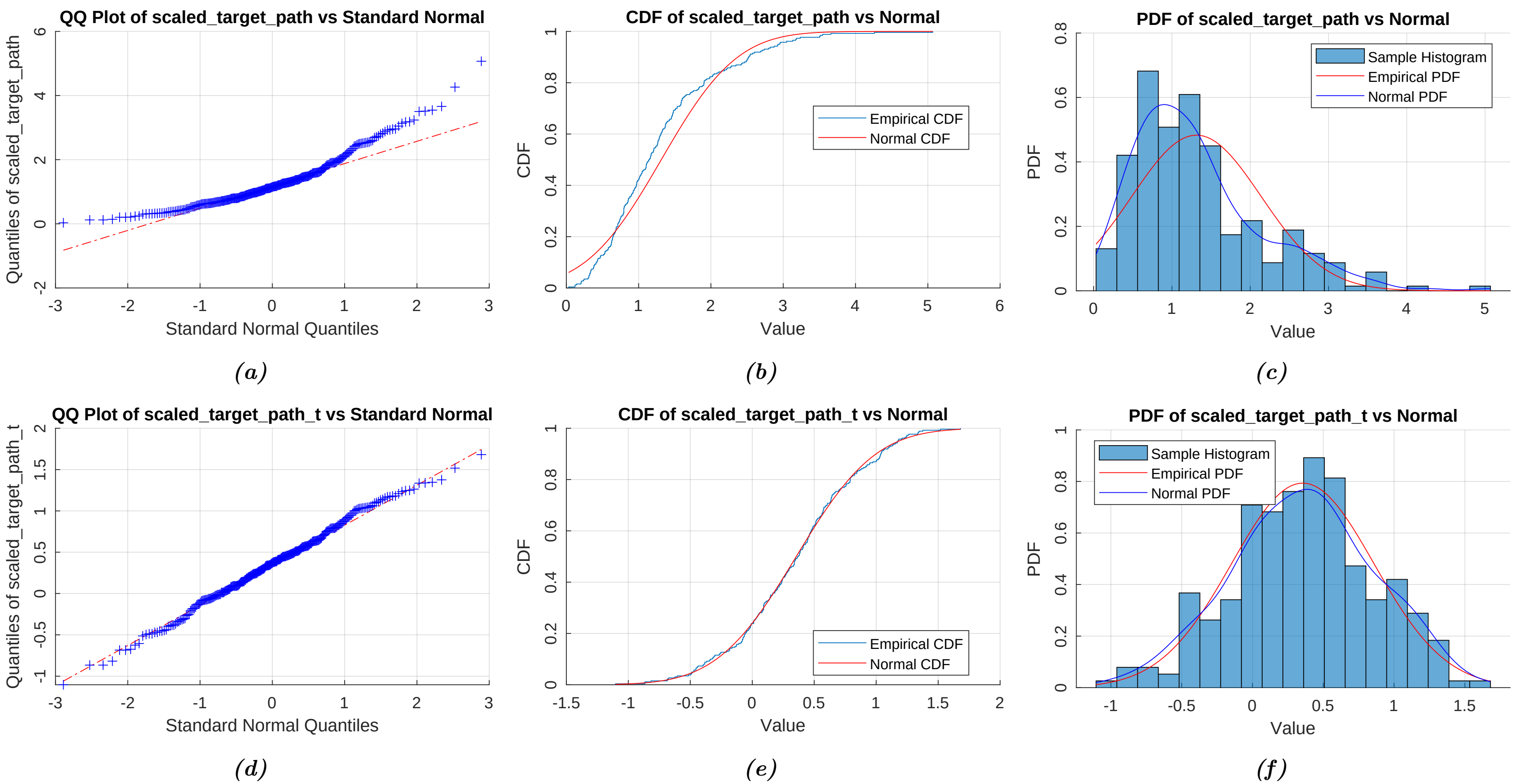
Table 6.4: Pre-analysis of data from experiment number 0, Levene's test (without COMP).



(a) *(b)* *(c)*

(d) *(e)* *(f)*

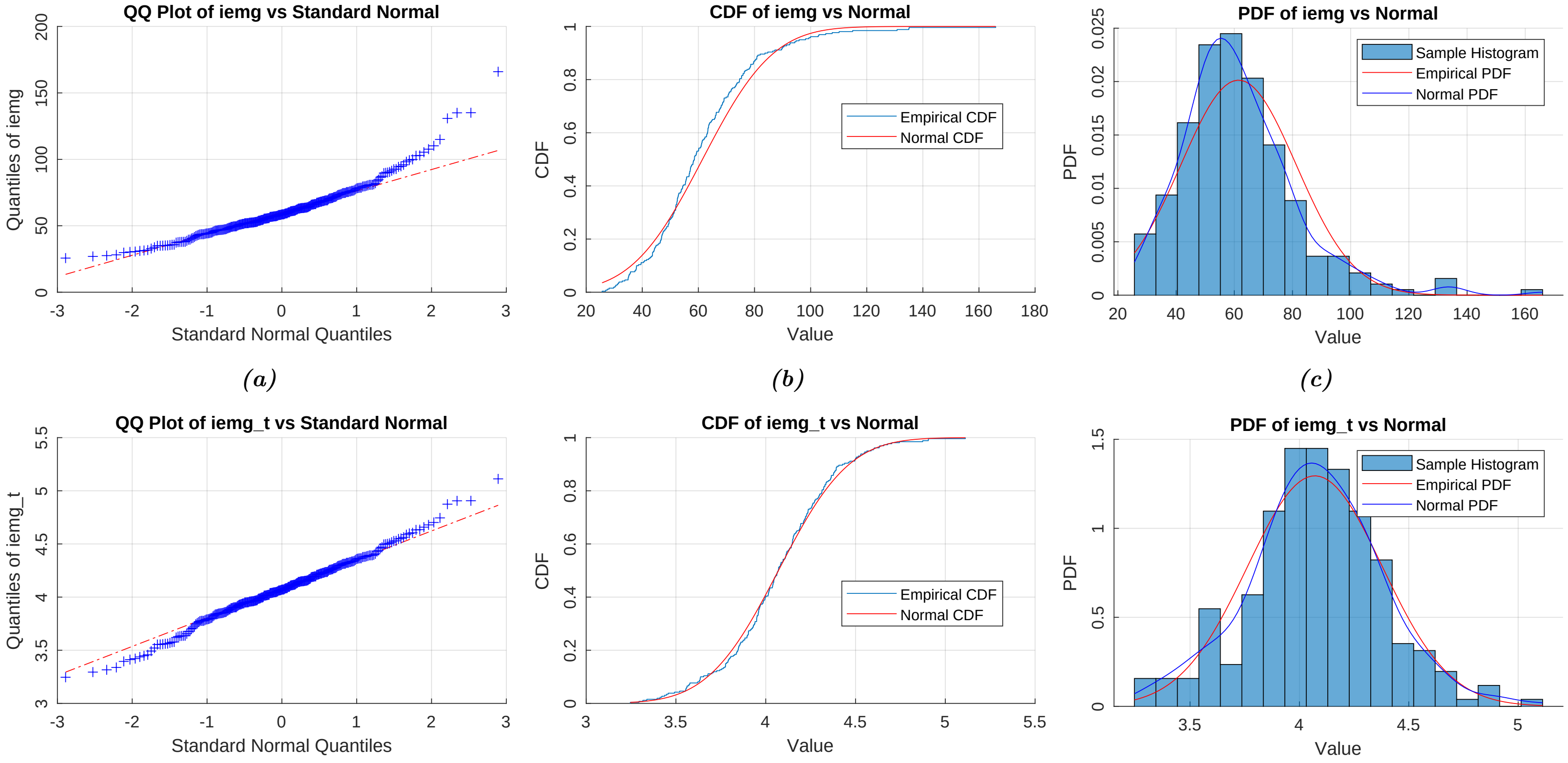
Figure 6.1: Pre-analysis of *throughput* vs transformed. Original data above and transformed data below: (a) and (d) QQ plots comparison; (b) and (e) CDF comparison; (c) and (f) histogram and PDF comparison.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

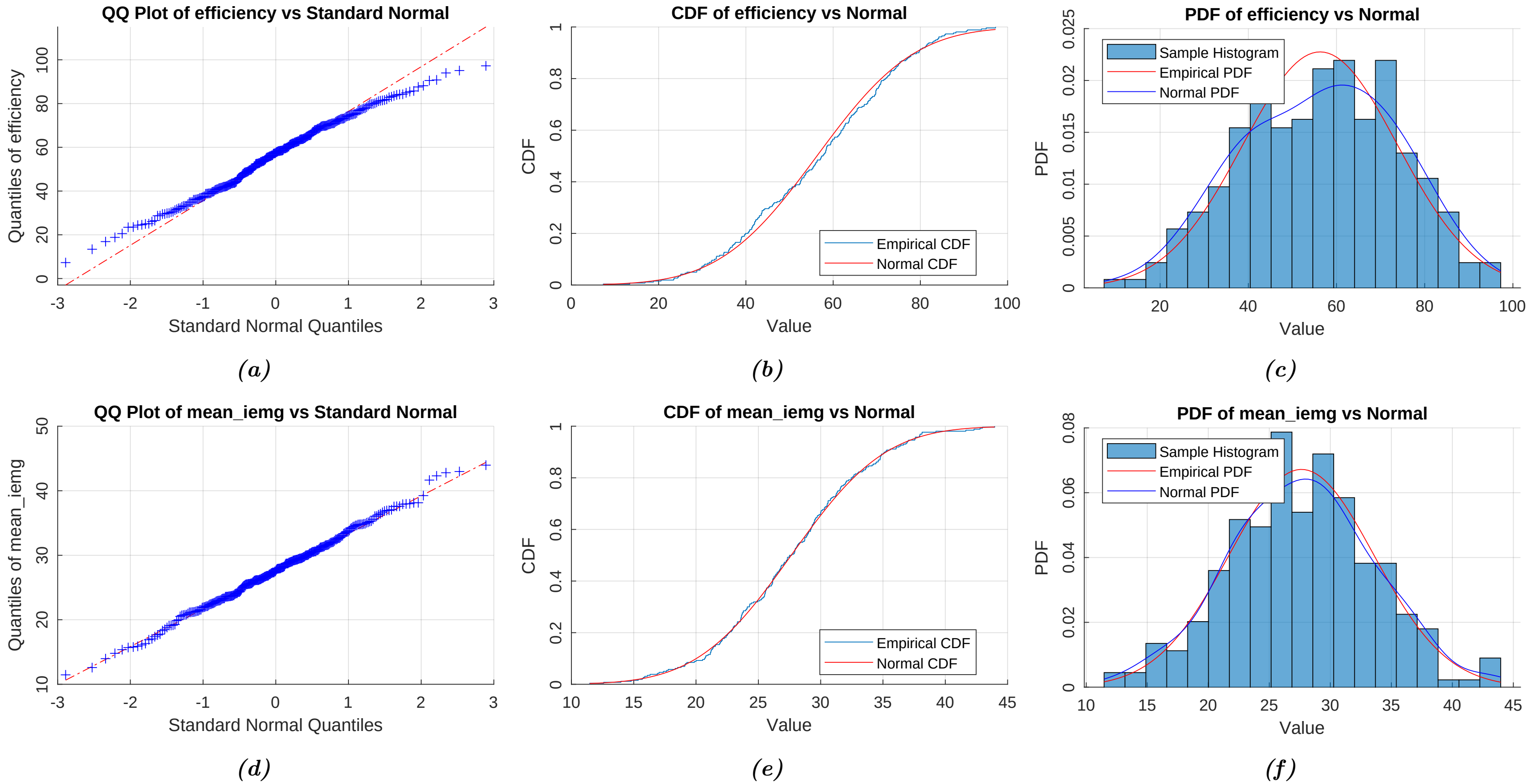
Figure 6.2: Pre-analysis of *scaled target path* vs transformed. Original data above and transformed data below: (a) and (d) QQ plots comparison; (b) and (e) CDF comparison; (c) and (f) histogram and PDF comparison.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

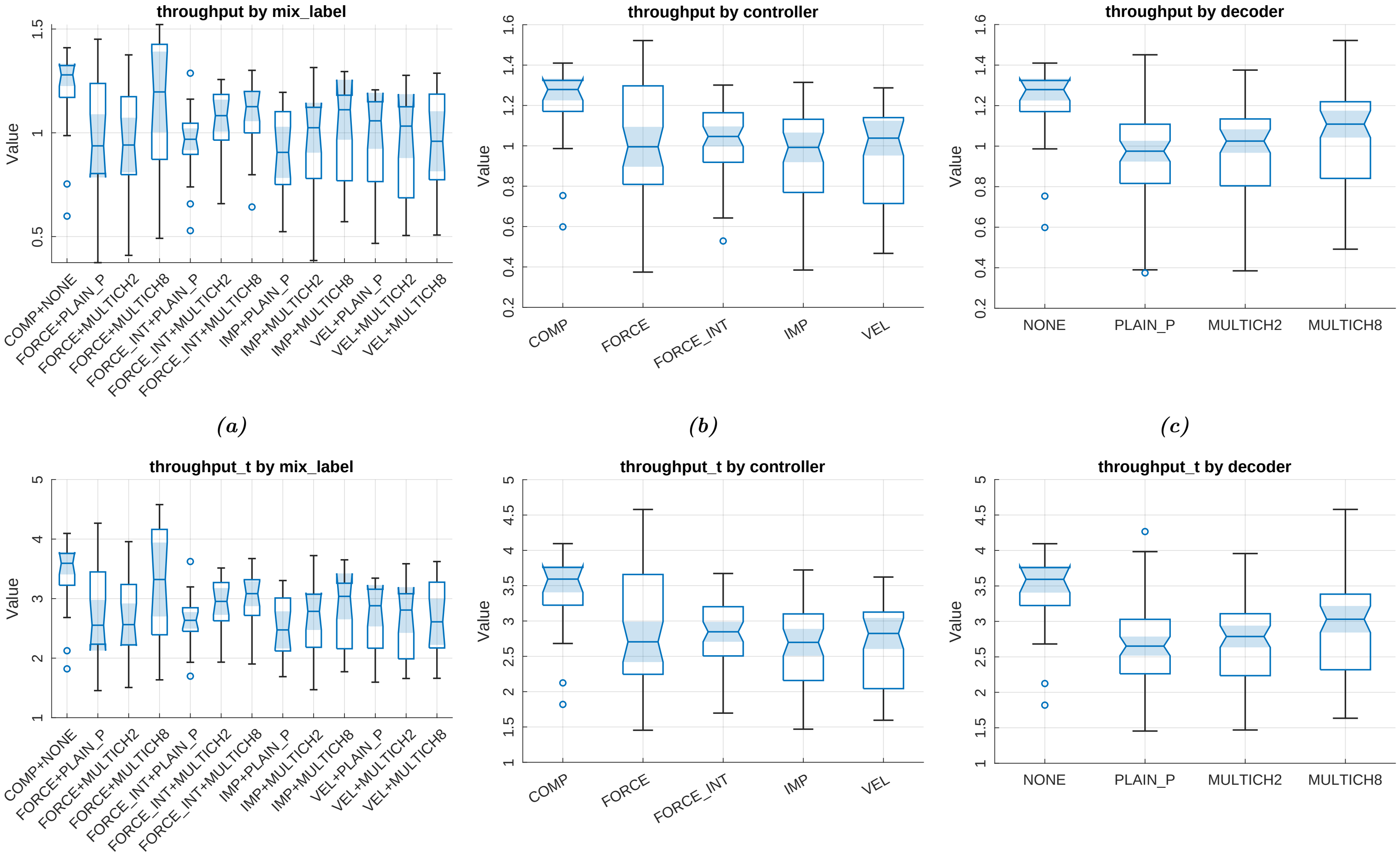
Figure 6.3: Pre-analysis of *IEmg* vs transformed. Original data above and transformed data below: (a) and (d) QQ plots comparison; (b) and (e) CDF comparison; (c) and (f) histogram and PDF comparison.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

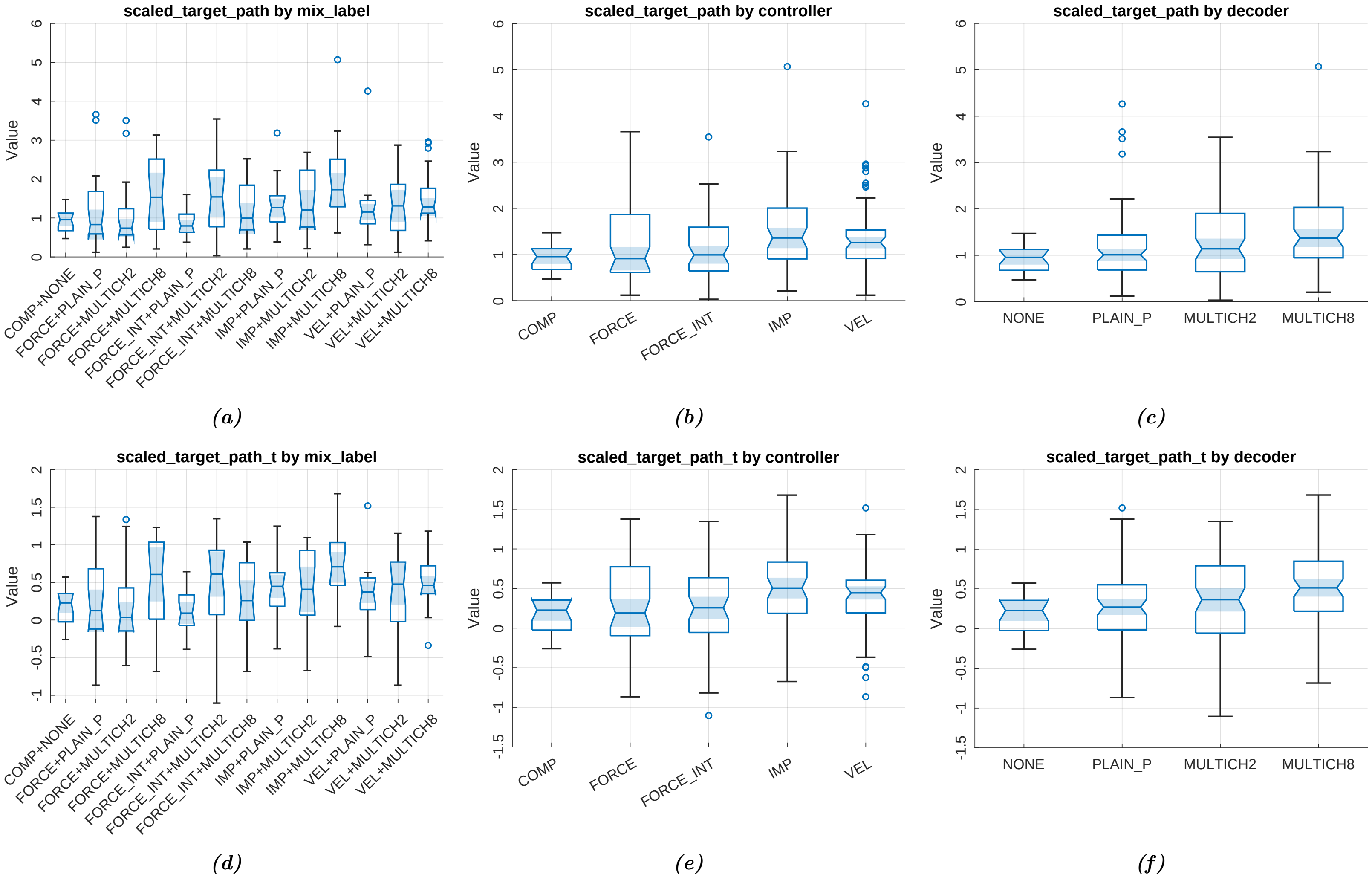
Figure 6.4: Pre-analysis of *efficiency* (above) and *mean IEmg* (below): (a) and (d) QQ plots; (b) and (e) CDF; (c) and (f) histogram and PDF.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 6.5: Pre-analysis of *throughput* vs transformed group variance. Original data above and transformed data below: (a) and (d) box grouped by `CTRL+DEC`; (b) and (e) box grouped by `CTRL`; (c) and (f) box grouped by `DEC`.



scaled_target_path by mix_label

scaled_target_path by controller

scaled_target_path by decoder

(a)

(b)

(c)

scaled_target_path_t by mix_label

scaled_target_path_t by controller

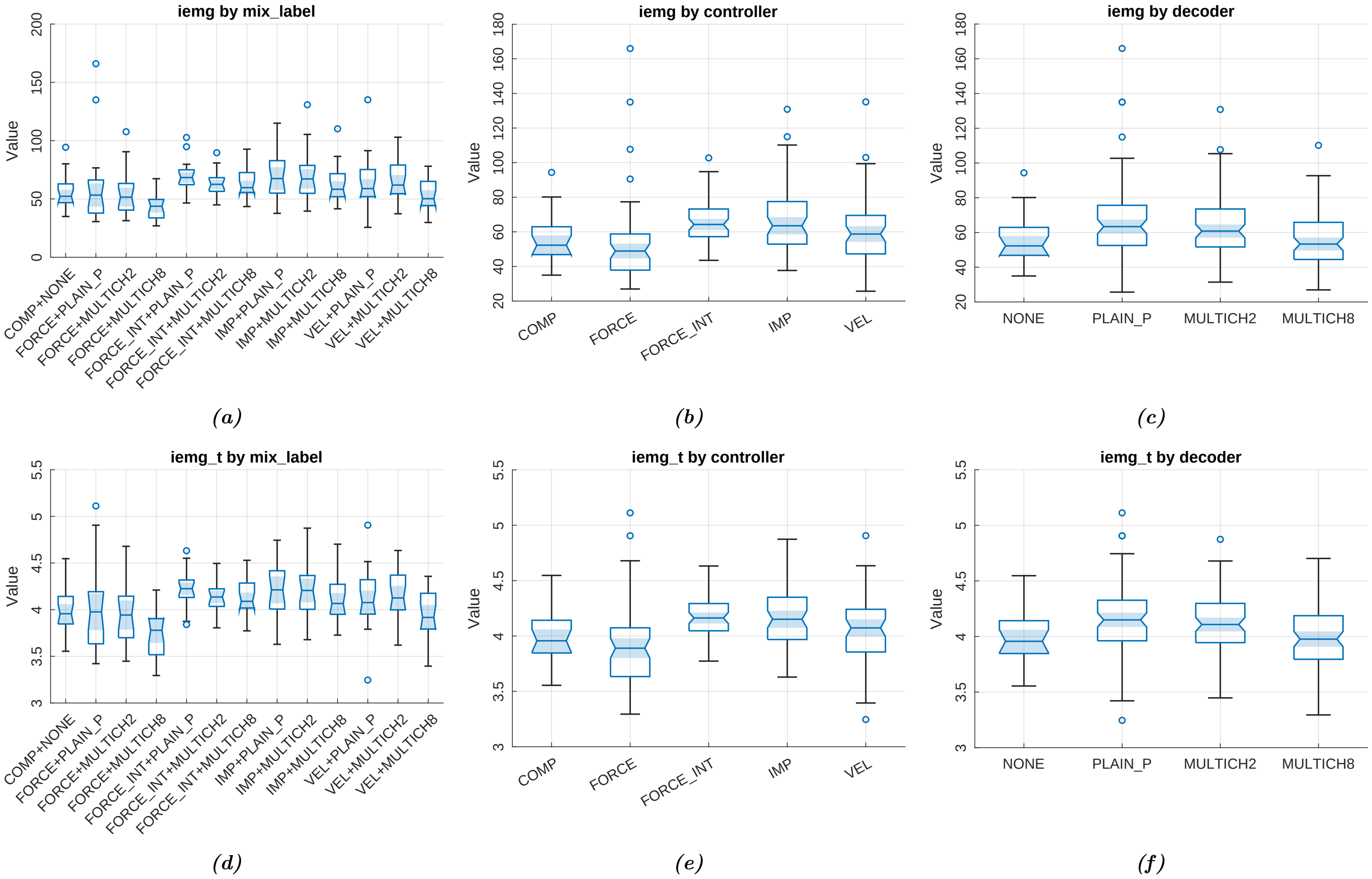
scaled_target_path_t by decoder

(d)

(e)

(f)

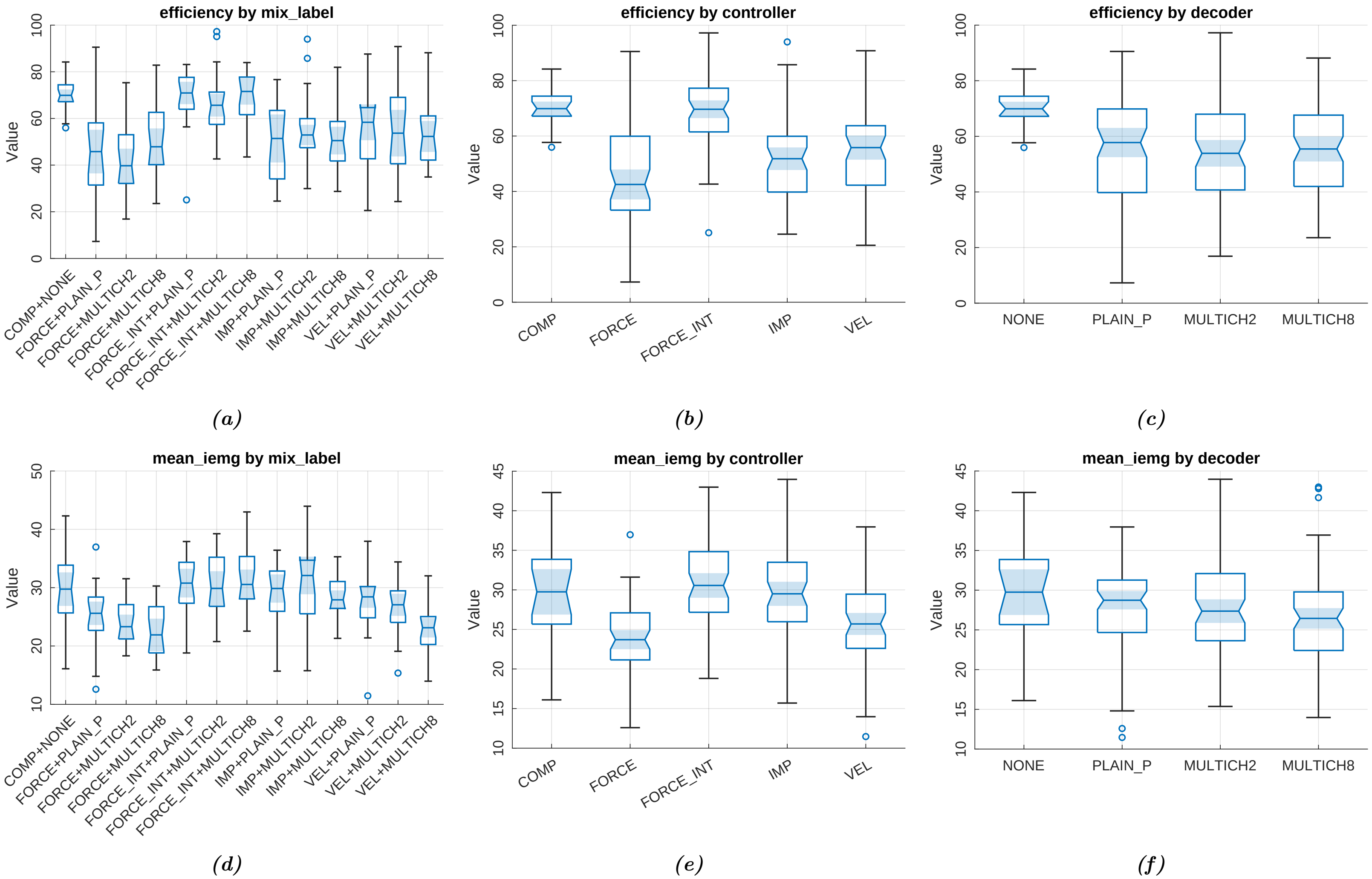
Figure 6.6: Pre-analysis of *scaled target path* vs transformed group variance. Original data above and transformed data below: (a) and (d) box grouped by `CTRL+DEC`; (b) and (e) box grouped by `CTRL`; (c) and (f) box grouped by `DEC`.



(a) *(b)* *(c)*

(d) *(e)* *(f)*

Figure 6.7: Pre-analysis of *IEmg* vs transformed group variance. Original data above and transformed data below: (a) and (d) box grouped by `CTRL+DEC`; (b) and (e) box grouped by `CTRL`; (c) and (f) box grouped by `DEC`.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 6.8: Pre-analysis of *efficiency* (above) and *mean IEmg* (below) group variance: (a) and (d) box grouped by `CTRL+DEC`; (b) and (e) box grouped by `CTRL`; (c) and (f) box grouped by `DEC`.

6.2.2 Pre-analysis results

Our pre-analysis let us to determine when the hypotheses for using powerful *parametric* ANOVA tools hold or, alternatively, if we are forced to use less powerful but more robust *non-parametric* ANOVA analysis.

Firstly, we checked for the normality hypothesis. We found out that there is no evidence that measures *efficiency* and *mean IEmg* are not normally distributed. The other measures analysed, instead, show significative difference from a normal distribution. So we decided to perform data transformation on these measures to obtain normally distributed values.

The *throughput* distribution showed negative skewness, so we used an exponential transform. The distribution of *scaled target path* and *IEmg* measures have positive skewness, so we employed logarithmic transformations. The direct and inverse transformations are reported in table 6.5. Parameters are found by linear search to maximize normality tests' p-value. The transformed measures resulted in distributions that passed the three tests we performed.

The homogeneity of variance among groups was respected for all the grouping strategies only by *mean IEmg*. The *throughput* and *IEmg* showed no evidence of inhomogeneous variance only when grouped by decoder, both the transformed and original version. The *null* hypothesis was not rejected also by the original *IEmg* when grouped by controller. In all the other cases H_0 was rejected. The option of excluding gravity compensation from the groups did not improved the situation so much. There is a change in *efficiency* test results, for which H_0 is only rejected with when grouped by controller, and in the original *scaled target path*, which now also do not reject H_0 when grouped by controller.

In conclusion, we looked for correlation between measures. Sample correlation coefficients normalized $r_{x,y}$ along with corresponding *p-values* for *null* hypothesis $r_{x,y} = 0$ are reported in tables 6.6 and 6.7. We found out a significant inversely proportional relation ($r < -0.6$) between *throughput* and *IEmg*, probably due to the common dependency from test duration: the longer the test lasts, the lower is the throughput, and the higher is the *total* effort. As a consequence, we are substantially measuring the same characteristics of the task. On the other hand, the mediated version of *IEmg* has little correlation with throughput ($|r| < 0.3$), because the mean makes the effort measure independent from test duration. Therefore, we exclude *IEmg* from further analysis and consider only *mean IEmg*. This last gives a more significative measure of the effort, as an intrinsic characteristics of the task and not as a direct consequence of task duration.

There is also evidence of a significant inversely proportional relation ($r < -0.6$) between *overshoots* and *throughput* and between *overshoots* and *efficiency*, but there is a weak directly proportional relation between *throughput* and *efficiency* ($0.4 < r < 0.5$). These relations are not quite surprising, since all these measures are in some way related to the concept of "precision". The lower is the precision, the more are the overshoots, so the lower the efficiency and the higher the completion time and, as a consequence, the lower is the throughput. Efficiency and throughput are not so strongly related, since efficiency accounts only for a part of completion time, given that it is a measure of travelled distance: also other factors affect completion time, e.g. mean velocity. However, even if overshoots are highly related to both throughput and efficiency, they measure different aspects of precision and cannot be "decorrelated" retaining a physical meaning for the measure, so it is important to maintain all of them in our analysis.

Measure	Measure → Transformed	Transformed → Measure
throughput	e^x	$\ln y$
scaled_target_path	$\ln(x + 0.3)$	$e^y - 0.3$
iemg	$\ln x$	e^y

Table 6.5: Data transformations for experiment number 0.

Measure	throughput	efficiency	scaled_target_path	overshoots	mean_iemg	iemg
throughput	1	0.447	0.095	−0.635	0.290	−0.689
efficiency	0.447	1	−0.319	−0.680	−0.067	−0.505
scaled_target_path	0.095	−0.319	1	−0.002	0.144	0.023
overshoots	−0.635	−0.680	−0.002	1	−0.098	0.635
mean_iemg	0.290	−0.067	0.144	−0.098	1	0.405
iemg	−0.689	−0.505	0.023	0.635	0.405	1

Table 6.6: Correlation between measures for experiment number 0.

Measure	throughput	efficiency	scaled_target_path	overshoots	mean_iemg	iemg
throughput	1	<0.001	0.126	<0.001	<0.001	<0.001
efficiency	<0.001	1	<0.001	<0.001	0.282	<0.001
scaled_target_path	0.126	<0.001	1	0.980	0.020	0.714
overshoots	<0.001	<0.001	0.980	1	0.115	<0.001
mean_iemg	<0.001	0.282	0.020	0.115	1	<0.001
iemg	<0.001	<0.001	0.714	<0.001	<0.001	1

Table 6.7: Correlation between measures for experiment number 0 – p-values.

6.2.3 Statistical tests and results

The objective of this first experiment was to answer the questions enumerated in subsection 5.3.2. We wanted to answer them with the support of statistical tests, to show differences among groups (determined by the type of controller, decoder, and the pair of them – mix) for the selected measures.

The first question involves the measures recorded and the system interface as the result of the combination between controller and decoder. So, we had to group the data by the *factor* `CTRL+DEC`, that enumerates all the tested configurations, and model how it affects the measure's value. This can be done by checking if there are differences in groups' means with one-way fixed effects parametric ANOVA analysis, or, when the assumptions of parametric ANOVA are not met, in groups' mean ranks with Kruskal-Wallis test. Higher rank is given to bigger values, so comparison between mean ranks is equivalent to comparison between medians when groups' distributions are similar. Unfortunately Kruskal-Wallis test has less statistical power of parametric ANOVA.

The second question follows from the first. Only when the previous tests show evidence of differences between groups, we can perform *post-hoc* multiple tests for all possible pairwise comparisons (with Tukey correction) to find out which factors are different.

The third question can be answered only by a two-way analysis in which we consider the controller and the decoder as two different factors. This allows us to check the interaction between them and the amount of variability each factor is able to intercept in the model we build. This analysis can be conducted using parametric two-way ANOVA.

In the pre-analysis we checked parametric ANOVA assumptions on data, which are normal distribution and homogeneity of variance among groups. The results allowed us to perform parametric ANOVA on *mean IEmg* without restrictions, since it is normally distributed and has homogeneous variance for all the grouping strategies. We decided to perform parametric ANOVA on *efficiency*, transformed *throughput* and *scaled target path* even if variance among groups is not homogeneous. Parametric ANOVA is usually considered robust to violations of this assumption. However, for these measures we performed also Kruskal-Wallis test as second check to see if conclusions are different. Measure *overshoots* being discrete cannot be tested with parametric ANOVA, so Kruskal-Wallis was used instead.

Interfaces comparison: one-way fixed effect `CTRL+DEC` In the following, we tested the results of an impaired subject performing computer-guided reaching tasks with an assistive elbow device. The orthosis can be controlled by mean of 13 different interfaces – *levels* – as pairs of controller and decoder, each tuned accordingly to the procedure described in chapter 5. The purpose was to examine the effect of control interface on some performance measures. Each task was repeated 20 times for each interface pair.

Results of parametric ANOVA *F-tests* showed that the type of interface used lead to statistically significant differences on the mean value of all the measures tested:

- *throughput* – after transformation – $F = 3.3616 > F_{crit} = F_{12,247;1-\alpha} = 1.7915$, $p = 1.4741 \times 10^{-4} < \alpha = 0.05$;
- *efficiency* – $F = 7.2004 > F_{crit}$, $p = 2.8938 \times 10^{-11} < \alpha$;
- *scaled target path* – after transformation – $F = 2.2570 > F_{crit}$, $p = 1.0014 \times 10^{-2} < \alpha$;

- *mean IEmg* – $F = 6.5279 > F_{crit}$, $p = 4.1844 \times 10^{-10} < \alpha$.

Results of non-parametric *Kruskal-Wallis* tests showed that the type of interface used lead to statistically significant differences on the mean rank about all the measure tested:

- *throughput* – $\chi^2 = 33.0572 > \chi^2_{crit} = \chi^2_{12;1-\alpha} = 21.0261$, $p = 9.4833 \times 10^{-4} < \alpha = 0.05$;
- *efficiency* – $\chi^2 = 72.1376 > \chi^2_{crit}$, $p = 1.2727 \times 10^{-10} < \alpha$;
- *scaled target path* – $\chi^2 = 31.3162 > \chi^2_{crit}$, $p = 1.7626 \times 10^{-3} < \alpha$;
- *overshoots* – $\chi^2 = 72.1376 > \chi^2_{crit}$, $p = 1.2727 \times 10^{-10} < \alpha$.

For variables on which we performed both tests, these results are aligned with parametric ANOVA tests.

Details on the tests performed are reported in tables 6.8, 6.10, 6.12, and 6.15 for parametric ANOVA *F-tests*, in tables 6.9, 6.11, 6.13, and 6.14 for non-parametric *Kruskal-Wallis* tests.

Since all the quantities we measured showed significative differences between the interfaces, we performed pairwise comparison *post-hoc* tests to find out which interfaces were different from the common mean. Based on which pairs resulted significantly different, we assigned the same letter to the *levels* (i.e. interface types) for which we had *no evidence* of statistical difference between them: in other term, the interfaces for which we did not found evidence they had a distinguishable behaviour about that performance measure. Each level can have one or more consecutive letter assigned, resulting in a label. If this labelling is done minimizing the total number of different letters used, it allows to define a partial order on the interfaces, based on what we can actually claim given the tests' results. These labels are reported in the following box-plots, capital letters for parametric ANOVA pairwise tests, lower case letters for Kruskal-Wallis ones.

Box-plots in figures 6.9, 6.10, 6.11, 6.12 and 6.13, show data grouped by `CTRL+DEC`. A detailed description of this kind of plots can be found in section 6.1.

Labels resulting from the pairwise tests are sorted in lexicographical order, always ascending because letters are renamed in such a way that the first letter is assigned to the best level. Levels with the same label are further sorted by mean – parametric tests – or by median value – non-parametric – in descending order if greater values are better (*throughput* and *efficiency*), ascending if lower values are better (*scaled target path*, *overshoots*, and *mean IEmg*).

Levels, values, confidence intervals, and labels are reported – sorted this way – in tables 6.16, 6.17, 6.18, 6.19, 6.20, 6.21, 6.23, and 6.24. We report also levels, values and confidence intervals for measures not tested for significative differences in tables 6.22 and 6.25.

Significative pairwise comparisons that lead to labels assigment are reported in tables 6.26, 6.27, 6.28, 6.29, 6.30, 6.31, 6.32, and 6.33. Each row of the tables contain the name of the two levels compared, namely level 1 and 2, the estimated difference $\mu_d = \mu_1 - \mu_2$, 95% confidence interval for μ_d, and the test's p-value. The quantity tested can either be mean variable value or mean group rank. From all the different pairs compared, only statistically significative ones – i.e. those for which the test gives a (corrected) $p < \alpha$ – are reported.

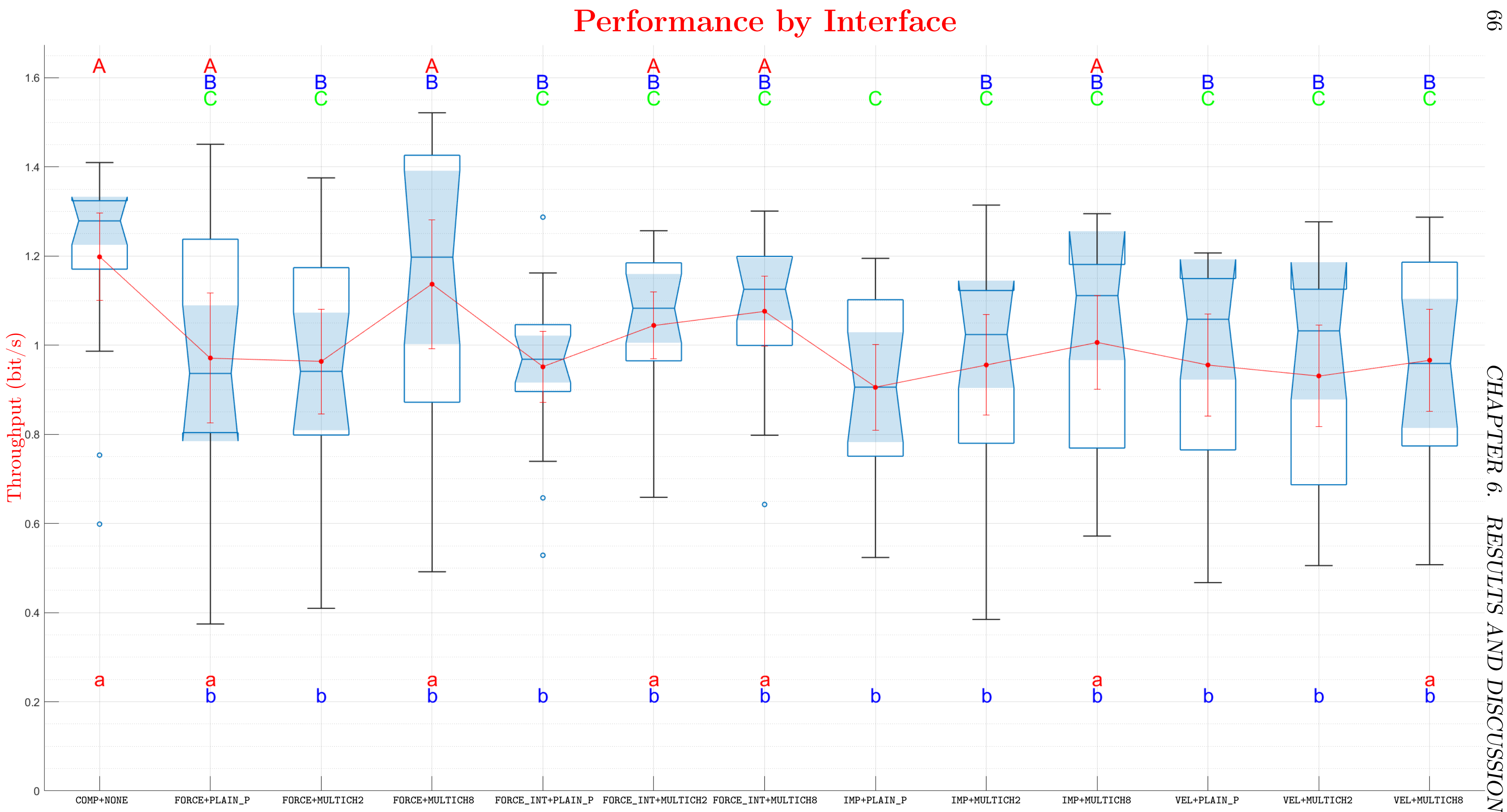


Figure 6.9: Results of pairwise comparisons between `CTRL+DEC` factor levels on *throughput*. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

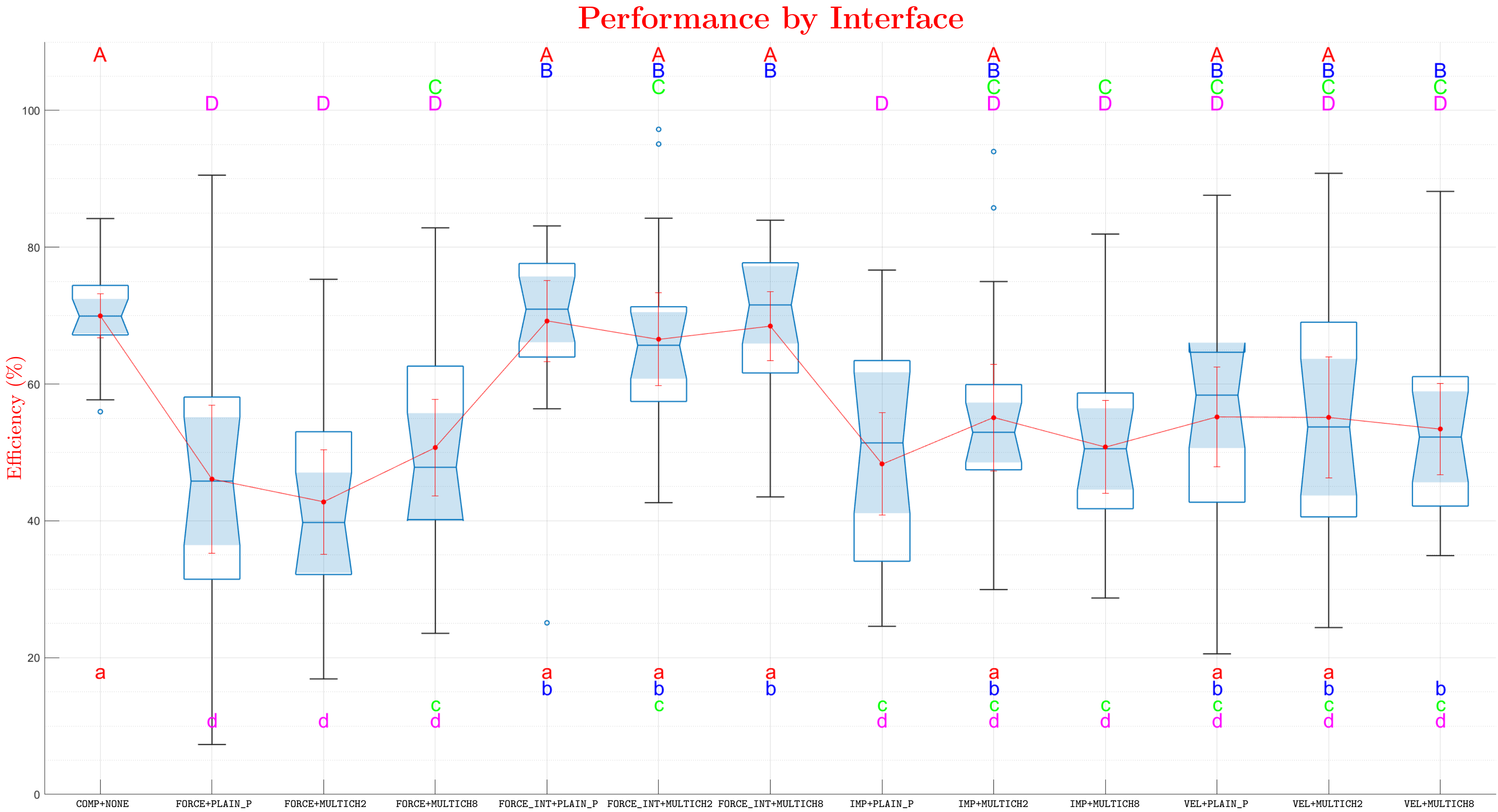


Figure 6.10: Results of pairwise comparisons between `CTRL+DEC` factor levels on *efficiency*. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

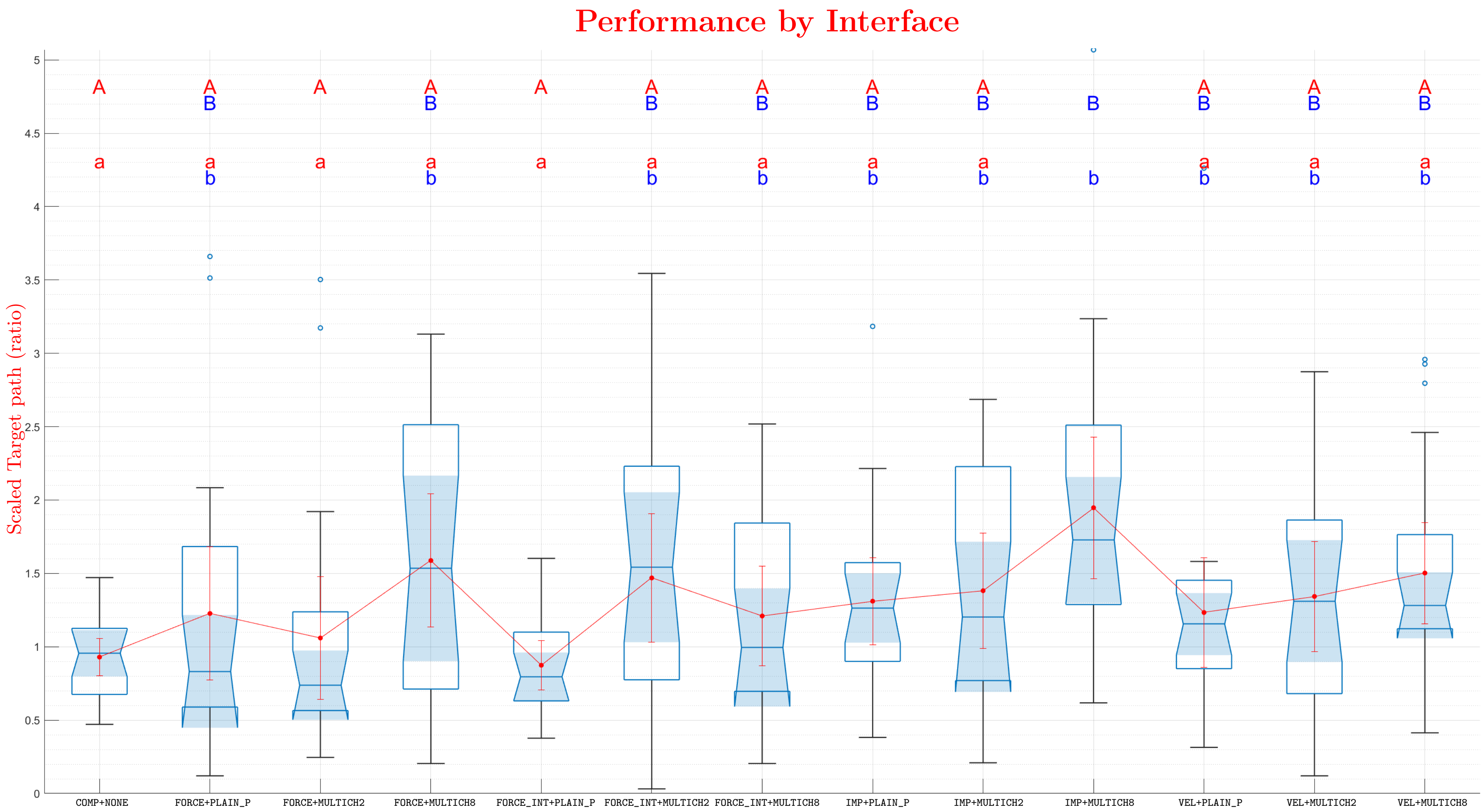


Figure 6.11: Results of pairwise comparisons between CTRL+DEC factor levels on *scaled target path*. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

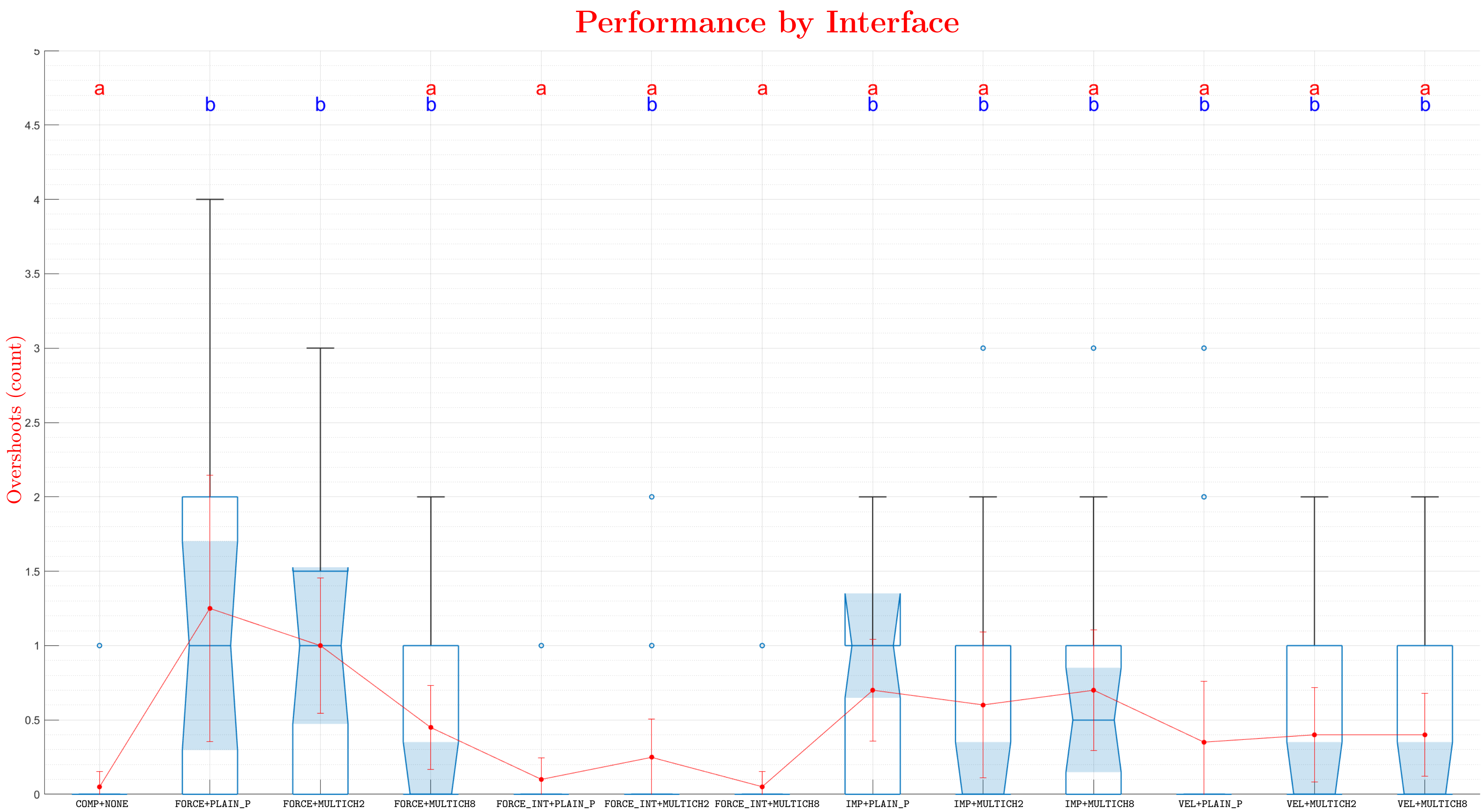


Figure 6.12: Results of pairwise comparisons between `CTRL+DEC` factor levels on *overshoots*. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

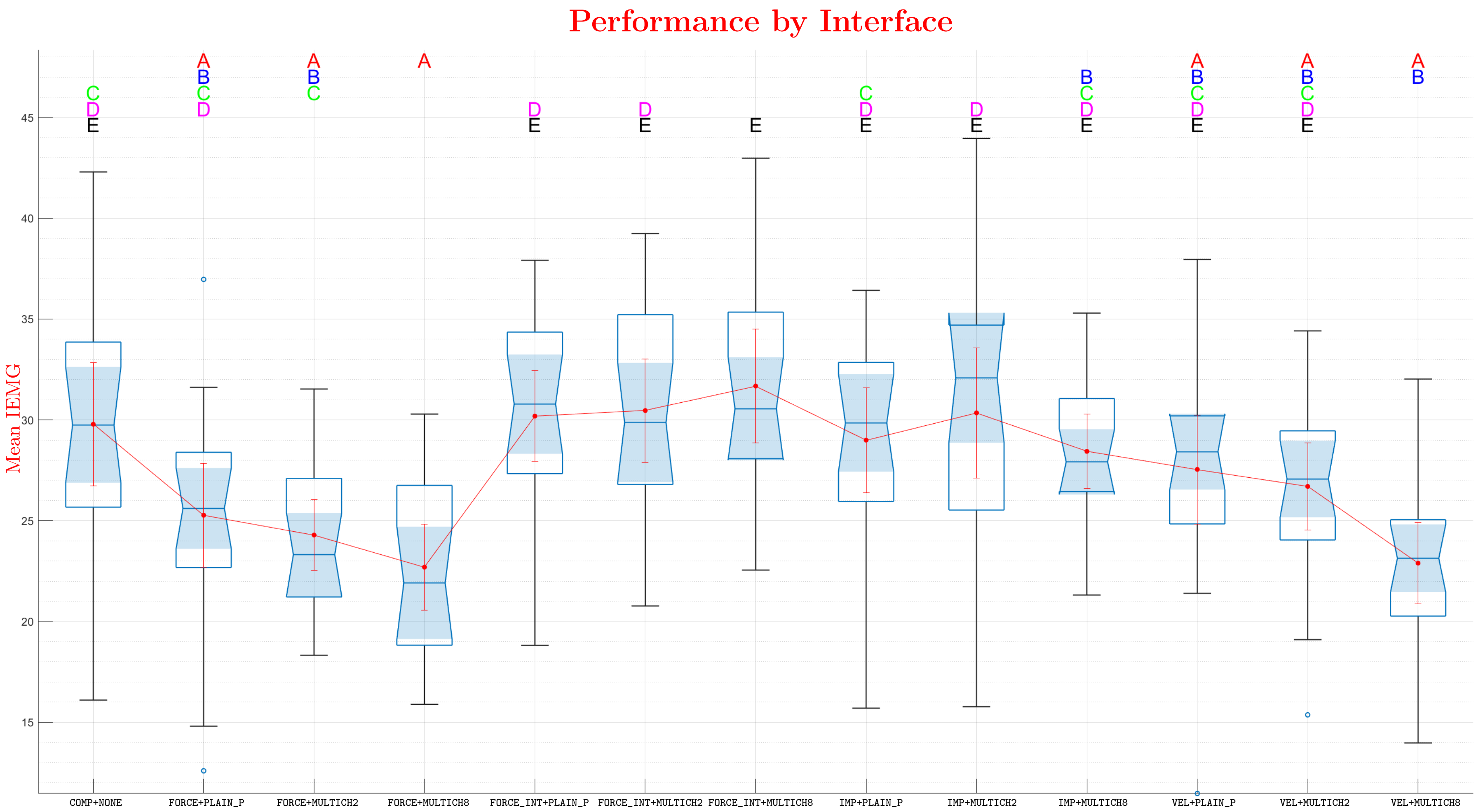


Figure 6.13: Results of pairwise comparisons between `CTRL+DEC` factor levels on *mean IEmg*. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

Source	SS	DOF	MS	F-value	p-value
CTRL+DEC	15.467	12	1.289	3.362	<0.001
Error	94.705	247	0.383		
Total	110.172	259			

Table 6.8: One-way ANOVA of *throughput* (transformed) with CTRL+DEC as factor.

Source	SS	DOF	MS	χ^2-value	p-value
Groups	$1.869 \cdot 10^5$	12	$1.558 \cdot 10^4$	33.057	0.001
Error	$1.278 \cdot 10^6$	247	$5.173 \cdot 10^3$		
Total	$1.465 \cdot 10^6$	259			

Table 6.9: Kruskal-Wallis test of *throughput* grouped by CTRL+DEC.

Source	SS	DOF	MS	F-value	p-value
CTRL+DEC	20 630.758	12	1 719.230	7.200	<0.001
Error	58 975.452	247	238.767		
Total	79 606.210	259			

Table 6.10: One-way ANOVA of *efficiency* with CTRL+DEC as factor.

Source	SS	DOF	MS	χ^2-value	p-value
Groups	$4.079 \cdot 10^5$	12	$3.399 \cdot 10^4$	72.138	<0.001
Error	$1.057 \cdot 10^6$	247	$4.278 \cdot 10^3$		
Total	$1.465 \cdot 10^6$	259			

Table 6.11: Kruskal-Wallis test of *efficiency* grouped by CTRL+DEC.

Source	SS	DOF	MS	F-value	p-value
CTRL+DEC	6.468	12	0.539	2.257	0.010
Error	58.987	247	0.239		
Total	65.455	259			

Table 6.12: One-way ANOVA of *scaled target path* (transformed) with CTRL+DEC as factor.

Source	SS	DOF	MS	χ^2-value	p-value
Groups	$1.771 \cdot 10^5$	12	$1.476 \cdot 10^4$	31.316	0.002
Error	$1.288 \cdot 10^6$	247	$5.213 \cdot 10^3$		
Total	$1.465 \cdot 10^6$	259			

Table 6.13: Kruskal-Wallis test of *scaled target path* grouped by CTRL+DEC.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$1.712 \cdot 10^5$	12	$1.427 \cdot 10^4$	44.860	<0.001
Error	$8.174 \cdot 10^5$	247	$3.309 \cdot 10^3$		
Total	$9.887 \cdot 10^5$	259			

Table 6.14: Kruskal-Wallis test of *overshoots* grouped by `CTRL+DEC`.

Source	SS	DoF	MS	F-value	p-value
CTRL+DEC	2 198.711	12	183.226	6.528	<0.001
Error	6 932.874	247	28.068		
Total	9 131.584	259			

Table 6.15: One-way ANOVA of *mean IEmg* with `CTRL+DEC` as factor.

Factor level	Mean throughput	Mean 95% CI		Labels		
		lower	upper			
COMP+NONE	1.198	1.100	1.296	A		
FORCE+MULTICH8	1.137	0.992	1.282	A	B	
FORCE_INT+MULTICH8	1.076	0.998	1.155	A	B	C
FORCE_INT+MULTICH2	1.044	0.970	1.119	A	B	C
IMP+MULTICH8	1.006	0.901	1.111	A	B	C
FORCE+PLAIN_P	0.971	0.826	1.117	A	B	C
VEL+MULTICH8	0.966	0.852	1.080		B	C
FORCE+MULTICH2	0.964	0.846	1.081		B	C
IMP+MULTICH2	0.956	0.843	1.068		B	C
VEL+PLAIN_P	0.956	0.842	1.070		B	C
FORCE_INT+PLAIN_P	0.951	0.872	1.031		B	C
VEL+MULTICH2	0.931	0.817	1.045		B	C
IMP+PLAIN_P	0.905	0.810	1.001			C

Table 6.16: Mean *throughput* by `CTRL+DEC` factor levels. Levels are sorted by not significantly different labels and decreasing mean *throughput*.

Factor level	Median throughput	Median 95% CI		Labels	
		lower	upper		
COMP+NONE	1.279	1.177	1.323	a	
FORCE+MULTICH8	1.197	0.881	1.424	a	b
FORCE_INT+MULTICH8	1.126	1.007	1.196	a	b
IMP+MULTICH8	1.111	0.769	1.176	a	b
FORCE_INT+MULTICH2	1.083	0.988	1.183	a	b
VEL+MULTICH8	0.959	0.799	1.176	a	b
FORCE+PLAIN_P	0.937	0.805	1.237	a	b
VEL+PLAIN_P	1.058	0.832	1.145		b
VEL+MULTICH2	1.032	0.694	1.121		b
IMP+MULTICH2	1.024	0.787	1.121		b
FORCE_INT+PLAIN_P	0.969	0.898	1.045		b
FORCE+MULTICH2	0.941	0.805	1.154		b
IMP+PLAIN_P	0.906	0.770	1.100		b

Table 6.17: Median *throughput* by `CTRL+DEC` factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and decreasing median *throughput*.

Factor level	Mean `efficiency`	Mean 95% CI		Labels			
		lower	upper				
COMP+NONE	69.965	66.735	73.195	A			
FORCE_INT+PLAIN_P	69.205	63.269	75.141	A	B		
FORCE_INT+MULTICH8	68.474	63.427	73.521	A	B		
FORCE_INT+MULTICH2	66.549	59.758	73.340	A	B	C	
VEL+PLAIN_P	55.179	47.874	62.485	A	B	C	D
VEL+MULTICH2	55.126	46.300	63.951	A	B	C	D
IMP+MULTICH2	55.080	47.298	62.862	A	B	C	D
VEL+MULTICH8	53.431	46.777	60.086		B	C	D
IMP+MULTICH8	50.797	43.985	57.608			C	D
FORCE+MULTICH8	50.712	43.659	57.765			C	D
IMP+PLAIN_P	48.322	40.864	55.781				D
FORCE+PLAIN_P	46.101	35.295	56.906				D
FORCE+MULTICH2	42.755	35.141	50.370				D

Table 6.18: Mean *efficiency* by CTRL+DEC factor levels. Levels are sorted by not significantly different labels and decreasing mean *efficiency*.

Factor level	Median `efficiency`	Median 95% CI		Labels			
		lower	upper				
COMP+NONE	69.900	67.470	74.399	a			
FORCE_INT+MULTICH8	71.569	61.728	77.397	a	b		
FORCE_INT+PLAIN_P	70.915	64.510	77.529	a	b		
FORCE_INT+MULTICH2	65.643	58.523	70.966	a	b	c	
VEL+PLAIN_P	58.348	42.928	64.028	a	b	c	d
VEL+MULTICH2	53.691	41.354	67.175	a	b	c	d
IMP+MULTICH2	52.909	47.628	59.699	a	b	c	d
VEL+MULTICH8	52.278	42.495	59.754		b	c	d
IMP+PLAIN_P	51.418	35.930	63.034			c	d
IMP+MULTICH8	50.510	41.871	58.345			c	d
FORCE+MULTICH8	47.856	40.437	62.543			c	d
FORCE+PLAIN_P	45.796	31.567	55.152				d
FORCE+MULTICH2	39.733	32.166	50.741				d

Table 6.19: Median *efficiency* by CTRL+DEC factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and decreasing median *efficiency*.

Factor level	Mean `scaled_target_path`	Mean 95% CI		Labels	
		lower	upper		
FORCE_INT+PLAIN_P	0.874	0.705	1.043	A	
COMP+NONE	0.930	0.802	1.057	A	
FORCE+MULTICH2	1.060	0.640	1.479	A	
FORCE_INT+MULTICH8	1.210	0.871	1.548	A	B
FORCE+PLAIN_P	1.227	0.773	1.681	A	B
VEL+PLAIN_P	1.234	0.860	1.607	A	B
IMP+PLAIN_P	1.311	1.014	1.608	A	B
VEL+MULTICH2	1.343	0.968	1.718	A	B
IMP+MULTICH2	1.382	0.989	1.775	A	B
FORCE_INT+MULTICH2	1.469	1.033	1.906	A	B
VEL+MULTICH8	1.503	1.158	1.847	A	B
FORCE+MULTICH8	1.588	1.134	2.041	A	B
IMP+MULTICH8	1.948	1.465	2.431		B

Table 6.20: Mean *scaled target path* by `CTRL+DEC` factor levels. Levels are sorted by not significantly different labels and increasing mean *scaled target path*.

Factor level	Median `scaled_target_path`	Median 95% CI		Labels	
		lower	upper		
FORCE+MULTICH2	0.738	0.582	1.171	a	
FORCE_INT+PLAIN_P	0.796	0.632	1.067	a	
COMP+NONE	0.956	0.684	1.120	a	
FORCE+PLAIN_P	0.833	0.610	1.605	a	b
FORCE_INT+MULTICH8	0.995	0.718	1.824	a	b
VEL+PLAIN_P	1.155	0.877	1.452	a	b
IMP+MULTICH2	1.204	0.814	2.166	a	b
IMP+PLAIN_P	1.265	0.923	1.553	a	b
VEL+MULTICH8	1.283	1.133	1.671	a	b
VEL+MULTICH2	1.312	0.697	1.845	a	b
FORCE+MULTICH8	1.534	0.712	2.500	a	b
FORCE_INT+MULTICH2	1.542	0.786	2.209	a	b
IMP+MULTICH8	1.728	1.291	2.391		b

Table 6.21: Median *scaled target path* by `CTRL+DEC` factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and increasing median *scaled target path*.

Level	Mean `overshoots`	Mean 95% CI	
		lower	upper
`COMP+NONE`	0.050	−0.055	0.155
`FORCE_INT+MULTICH8`	0.050	−0.055	0.155
`FORCE_INT+PLAIN_P`	0.100	−0.044	0.244
`FORCE_INT+MULTICH2`	0.250	−0.007	0.507
`VEL+PLAIN_P`	0.350	−0.060	0.760
`VEL+MULTICH2`	0.400	0.081	0.719
`VEL+MULTICH8`	0.400	0.120	0.680
`FORCE+MULTICH8`	0.450	0.167	0.733
`IMP+MULTICH2`	0.600	0.110	1.090
`IMP+PLAIN_P`	0.700	0.357	1.043
`IMP+MULTICH8`	0.700	0.295	1.105
`FORCE+MULTICH2`	1	0.544	1.456
`FORCE+PLAIN_P`	1.250	0.353	2.147

Table 6.22: Mean *overshoots* by `CTRL+DEC` factor levels. Levels are sorted by increasing mean *overshoots*. Differences in levels' means have not been tested.

Factor level	Median `overshoots`	Median 95% CI		Labels	
		lower	upper		
`COMP+NONE`	0	0	0	a	
`FORCE_INT+MULTICH8`	0	0	0	a	
`FORCE_INT+PLAIN_P`	0	0	0	a	
`FORCE+MULTICH8`	0	0	1	a	b
`FORCE_INT+MULTICH2`	0	0	0	a	b
`VEL+MULTICH8`	0	0	1	a	b
`VEL+MULTICH2`	0	0	1	a	b
`IMP+MULTICH2`	0	0	1	a	b
`VEL+PLAIN_P`	0	0	0	a	b
`IMP+MULTICH8`	0.500	0	1	a	b
`IMP+PLAIN_P`	1	0	1	a	b
`FORCE+PLAIN_P`	1	0	2		b
`FORCE+MULTICH2`	1	0	1.237		b

Table 6.23: Median *overshoots* by `CTRL+DEC` factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and increasing median *overshoots*.

Factor level	Mean `mean_iemg`	Mean 95% CI		Labels				
		lower	upper					
FORCE+MULTICH8	22.694	20.560	24.829	A				
VEL+MULTICH8	22.892	20.878	24.907	A	B			
FORCE+MULTICH2	24.288	22.530	26.046	A	B	C		
FORCE+PLAIN_P	25.270	22.685	27.855	A	B	C	D	
VEL+MULTICH2	26.700	24.544	28.857	A	B	C	D	E
VEL+PLAIN_P	27.542	24.835	30.250	A	B	C	D	E
IMP+MULTICH8	28.439	26.601	30.277		B	C	D	E
IMP+PLAIN_P	28.995	26.391	31.600			C	D	E
COMP+NONE	29.783	26.726	32.839			C	D	E
FORCE_INT+PLAIN_P	30.189	27.941	32.438				D	E
IMP+MULTICH2	30.349	27.121	33.577				D	E
FORCE_INT+MULTICH2	30.464	27.899	33.028				D	E
FORCE_INT+MULTICH8	31.679	28.851	34.508					E

Table 6.24: Mean *mean IEmg* by CTRL+DEC factor levels. Levels are sorted by not significantly different labels and increasing mean *mean IEmg*.

Level	Median `mean_iemg`	Median 95% CI	
		lower	upper
FORCE+MULTICH8	21.910	18.977	26.712
VEL+MULTICH8	23.134	20.403	24.821
FORCE+MULTICH2	23.315	21.224	26.793
FORCE+PLAIN_P	25.610	22.764	28.209
VEL+MULTICH2	27.069	24.172	29.391
IMP+MULTICH8	27.919	26.471	30.522
VEL+PLAIN_P	28.424	25.222	30.106
COMP+NONE	29.750	25.865	33.717
IMP+PLAIN_P	29.853	27.199	32.753
FORCE_INT+MULTICH2	29.871	26.804	35.031
FORCE_INT+MULTICH8	30.556	28.155	34.968
FORCE_INT+PLAIN_P	30.783	27.333	34.291
IMP+MULTICH2	32.090	25.536	34.656

Table 6.25: Median *mean IEmg* by CTRL+DEC factor levels. Levels are sorted by increasing median *mean IEmg*. Differences in levels' medians have not been tested.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	FORCE+MULTICH2	0.224	0.010	0.399	0.031
COMP+NONE	FORCE_INT+PLAIN_P	0.252	0.039	0.428	0.008
COMP+NONE	IMP+PLAIN_P	0.292	0.079	0.468	0.001
COMP+NONE	IMP+MULTICH2	0.235	0.022	0.411	0.019
COMP+NONE	VEL+PLAIN_P	0.235	0.022	0.411	0.019
COMP+NONE	VEL+MULTICH2	0.258	0.045	0.434	0.006
COMP+NONE	VEL+MULTICH8	0.223	0.010	0.399	0.032
FORCE+MULTICH8	IMP+PLAIN_P	0.255	0.033	0.437	0.012

Table 6.26: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *throughput* (de-transformed). Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	FORCE+MULTICH2	80.200	1.422	158.978	0.041
COMP+NONE	FORCE_INT+PLAIN_P	91.050	12.272	169.828	0.008
COMP+NONE	IMP+PLAIN_P	101.450	22.672	180.228	0.001
COMP+NONE	IMP+MULTICH2	81.900	3.122	160.678	0.033
COMP+NONE	VEL+PLAIN_P	79.800	1.022	158.578	0.044
COMP+NONE	VEL+MULTICH2	89.550	10.772	168.328	0.011

Table 6.27: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *throughput*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	FORCE+PLAIN_P	23.865	7.677	40.052	<0.001
COMP+NONE	FORCE+MULTICH2	27.210	11.023	43.397	<0.001
COMP+NONE	FORCE+MULTICH8	19.254	3.066	35.441	0.005
COMP+NONE	IMP+PLAIN_P	21.643	5.456	37.830	0.001
COMP+NONE	IMP+MULTICH8	19.169	2.981	35.356	0.006
COMP+NONE	VEL+MULTICH8	16.534	0.347	32.721	0.040
FORCE+PLAIN_P	FORCE_INT+PLAIN_P	−23.105	−39.292	−6.918	<0.001
FORCE+PLAIN_P	FORCE_INT+MULTICH2	−20.448	−36.636	−4.261	0.002
FORCE+PLAIN_P	FORCE_INT+MULTICH8	−22.374	−38.561	−6.186	<0.001
FORCE+MULTICH2	FORCE_INT+PLAIN_P	−26.450	−42.637	−10.263	<0.001
FORCE+MULTICH2	FORCE_INT+MULTICH2	−23.794	−39.981	−7.606	<0.001
FORCE+MULTICH2	FORCE_INT+MULTICH8	−25.719	−41.906	−9.532	<0.001
FORCE+MULTICH8	FORCE_INT+PLAIN_P	−18.494	−34.681	−2.306	0.010
FORCE+MULTICH8	FORCE_INT+MULTICH8	−17.762	−33.950	−1.575	0.017
FORCE_INT+PLAIN_P	IMP+PLAIN_P	20.883	4.696	37.070	0.001
FORCE_INT+PLAIN_P	IMP+MULTICH8	18.409	2.222	34.596	0.011
FORCE_INT+MULTICH2	IMP+PLAIN_P	18.227	2.039	34.414	0.012
FORCE_INT+MULTICH8	IMP+PLAIN_P	20.152	3.965	36.339	0.003
FORCE_INT+MULTICH8	IMP+MULTICH8	17.678	1.490	33.865	0.018

Table 6.28: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *efficiency*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	FORCE+PLAIN_P	103	24.222	181.778	0.001
COMP+NONE	FORCE+MULTICH2	120.700	41.922	199.478	<0.001
COMP+NONE	FORCE+MULTICH8	90.400	11.622	169.178	0.009
COMP+NONE	IMP+PLAIN_P	98.500	19.722	177.278	0.002
COMP+NONE	IMP+MULTICH8	89.950	11.172	168.728	0.010
COMP+NONE	VEL+MULTICH8	80.200	1.422	158.978	0.041
FORCE+PLAIN_P	FORCE_INT+PLAIN_P	−100.550	−179.328	−21.772	0.002
FORCE+PLAIN_P	FORCE_INT+MULTICH2	−79.250	−158.028	−0.472	0.047
FORCE+PLAIN_P	FORCE_INT+MULTICH8	−94.950	−173.728	−16.172	0.004
FORCE+MULTICH2	FORCE_INT+PLAIN_P	−118.250	−197.028	−39.472	<0.001
FORCE+MULTICH2	FORCE_INT+MULTICH2	−96.950	−175.728	−18.172	0.003
FORCE+MULTICH2	FORCE_INT+MULTICH8	−112.650	−191.428	−33.872	<0.001
FORCE+MULTICH8	FORCE_INT+PLAIN_P	−87.950	−166.728	−9.172	0.014
FORCE+MULTICH8	FORCE_INT+MULTICH8	−82.350	−161.128	−3.572	0.031
FORCE_INT+PLAIN_P	IMP+PLAIN_P	96.050	17.272	174.828	0.004
FORCE_INT+PLAIN_P	IMP+MULTICH8	87.500	8.722	166.278	0.015
FORCE_INT+MULTICH8	IMP+PLAIN_P	90.450	11.672	169.228	0.009
FORCE_INT+MULTICH8	IMP+MULTICH8	81.900	3.122	160.678	0.033

Table 6.29: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *efficiency*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	IMP+MULTICH8	−0.860	−1.341	−0.057	0.028
FORCE+MULTICH2	IMP+MULTICH8	−0.895	−1.362	−0.117	0.014
FORCE_INT+PLAIN_P	IMP+MULTICH8	−0.936	−1.387	−0.185	0.006

Table 6.30: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *scaled target path* (de-transformed). Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	IMP+MULTICH8	−87.250	−166.028	−8.472	0.015
FORCE+MULTICH2	IMP+MULTICH8	−89.750	−168.528	−10.972	0.010
FORCE_INT+PLAIN_P	IMP+MULTICH8	−95.800	−174.578	−17.022	0.004

Table 6.31: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *scaled target path.* Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	FORCE+PLAIN_P	−70.750	−135.473	−6.027	0.018
COMP+NONE	FORCE+MULTICH2	−80.025	−144.748	−15.302	0.003
FORCE+PLAIN_P	FORCE_INT+PLAIN_P	64.975	0.252	129.698	0.048
FORCE+PLAIN_P	FORCE_INT+MULTICH8	70.750	6.027	135.473	0.018
FORCE+MULTICH2	FORCE_INT+PLAIN_P	74.250	9.527	138.973	0.009
FORCE+MULTICH2	FORCE_INT+MULTICH8	80.025	15.302	144.748	0.003

Table 6.32: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *overshoots.* Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
COMP+NONE	FORCE+MULTICH8	7.088	1.538	12.638	0.002
COMP+NONE	VEL+MULTICH8	6.890	1.340	12.440	0.003
FORCE+PLAIN_P	FORCE_INT+MULTICH8	−6.409	−11.959	−0.859	0.008
FORCE+MULTICH2	FORCE_INT+PLAIN_P	−5.901	−11.451	−0.351	0.025
FORCE+MULTICH2	FORCE_INT+MULTICH2	−6.176	−11.726	−0.625	0.014
FORCE+MULTICH2	FORCE_INT+MULTICH8	−7.391	−12.941	−1.841	0.001
FORCE+MULTICH2	IMP+MULTICH2	−6.061	−11.611	−0.511	0.018
FORCE+MULTICH8	FORCE_INT+PLAIN_P	−7.495	−13.045	−1.945	0.001
FORCE+MULTICH8	FORCE_INT+MULTICH2	−7.769	−13.319	−2.219	<0.001
FORCE+MULTICH8	FORCE_INT+MULTICH8	−8.985	−14.535	−3.435	<0.001
FORCE+MULTICH8	IMP+PLAIN_P	−6.301	−11.851	−0.751	0.011
FORCE+MULTICH8	IMP+MULTICH2	−7.655	−13.205	−2.105	<0.001
FORCE+MULTICH8	IMP+MULTICH8	−5.745	−11.295	−0.195	0.035
FORCE_INT+PLAIN_P	VEL+MULTICH8	7.297	1.747	12.847	0.001
FORCE_INT+MULTICH2	VEL+MULTICH8	7.571	2.021	13.121	<0.001
FORCE_INT+MULTICH8	VEL+MULTICH8	8.787	3.237	14.337	<0.001
IMP+PLAIN_P	VEL+MULTICH8	6.103	0.553	11.653	0.017
IMP+MULTICH2	VEL+MULTICH8	7.457	1.907	13.007	0.001

Table 6.33: Significative ($p < \alpha$) pairwise comparisons between CTRL+DEC factor levels on *mean IEmg*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

Components comparison: two-way fixed effects `CTRL` and `DEC` In the following, we tested the same tasks of the previous paragraph but according to a two-way model. There are two factors that determine the myoelectric interface: the controller – `CTRL` – with 4 levels and the decoder – `DEC` – with 3 levels. We excluded the gravity compensation controller since it has no decoder: it would lead to an unbalanced design, difficult to threat statistically, and it has already been included in previous analysis. The purpose was to examine the effect of myoelectric interface components on some performance measures. Each task was repeated 20 times for each controller and decoder pair, a complete experiment design.

Results of parametric ANOVA *F-tests* are reported by measure.

- *throughput* – after transformation:
  - saturated model allowed to reject H_0 on interaction term, so factors can be analysed independently:
    - `CTRL` × `DEC` interaction factor has *not significative* effect on throughput: $F_{\texttt{CTRL}\times\texttt{DEC}} = 0.969\,47 < F_{\texttt{CTRL}\times\texttt{DEC},crit} = F_{6,228;1-\alpha} = 2.1385$, $p = 4.4666 \times 10^{-1} > \alpha = 0.05$;
    - `CTRL` factor has *not significative* effect on throughput: $F_{\texttt{CTRL}} = 2.3126 < F_{\texttt{CTRL},crit} = F_{3,228;1-\alpha} = 2.6442$, $p = 7.6865 \times 10^{-2} > \alpha = 0.05$;
    - `DEC` factor has *significative* effect on throughput: $F_{\texttt{DEC}} = 4.6688 > F_{\texttt{DEC},crit} = F_{2,228;1-\alpha} = 3.0354$, $p = 1.0298 \times 10^{-2} < \alpha = 0.05$.
  - reduced one-factor model shows that `DEC` factor has *significative* effect on throughput: $F''_{\texttt{DEC}} = 4.5960 > F''_{\texttt{DEC},crit} = F_{2,237;1-\alpha} = 3.0339$, $p = 1.1008 \times 10^{-2} < \alpha = 0.05$.

- *efficiency*:
  - saturated model allowed to reject H_0 on interaction term, so factors can be analysed independently:
    - `CTRL` × `DEC` interaction factor has *not significative* effect on efficiency: $F_{\texttt{CTRL}\times\texttt{DEC}} = 0.758\,57 < F_{\texttt{CTRL}\times\texttt{DEC},crit}$, $p = 6.0322 \times 10^{-1} > \alpha = 0.05$;
    - `CTRL` factor has *significative* effect on efficiency: $F_{\texttt{CTRL}} = 20.096 > F_{\texttt{CTRL},crit}$, $p = 1.3619 \times 10^{-11} < \alpha = 0.05$;
    - `DEC` factor has *not significative* effect on efficiency: $F_{\texttt{DEC}} = 0.120\,91 < F_{\texttt{DEC},crit}$, $p = 8.8617 \times 10^{-1} > \alpha = 0.05$.
  - reduced one-factor model shows that `CTRL` factor has *significative* effect on efficiency: $F''_{\texttt{CTRL}} = 20.372 > F''_{\texttt{CTRL},crit} = F_{3,236;1-\alpha} = 2.6429$, $p = 8.9429 \times 10^{-12} < \alpha = 0.05$.

- *scaled target path* – after transformation:
  - saturated model allowed to reject H_0 on interaction term, so factors can be analysed independently:
    - `CTRL` × `DEC` interaction factor has *not significative* effect on scaled target path: $F_{\texttt{CTRL}\times\texttt{DEC}} = 0.990\,66 < F_{\texttt{CTRL}\times\texttt{DEC},crit}$, $p = 4.3233 \times 10^{-1} > \alpha = 0.05$;
    - `CTRL` factor has *significative* effect on scaled target path: $F_{\texttt{CTRL}} = 2.7090 > F_{\texttt{CTRL},crit}$, $p = 4.5953 \times 10^{-2} < \alpha = 0.05$;
    - `DEC` factor has *significative* effect on scaled target path: $F_{\texttt{DEC}} = 4.4098 > F_{\texttt{DEC},crit}$, $p = 1.3212 \times 10^{-2} < \alpha = 0.05$.
  - independence model showed that both `CTRL` and `DEC` factors has *significative* effect on scaled target path:

- $*$ $F'_{\texttt{CTRL}} = 2.7096 > F'_{\texttt{CTRL},crit} = F_{3,234;1-\alpha} = 2.6432$, $p = 4.5850 \times 10^{-2} < \alpha = 0.05$;
- $*$ $F'_{\texttt{DEC}} = 4.4108 > F'_{\texttt{DEC},crit} = F_{2,234;1-\alpha} = 3.0344$, $p = 1.3171 \times 10^{-2} < \alpha = 0.05$.

- *mean IEmg*:
  - saturated model allowed to reject H_0 on interaction term, so factors can be analysed independently:
    - $*$ CTRL × DEC interaction factor has *not significative* effect on efficiency: $F_{\texttt{CTRL}\times\texttt{DEC}} = 1.5450 < F_{\texttt{CTRL}\times\texttt{DEC},crit}$, $p = 1.6446 \times 10^{-1} > \alpha = 0.05$;
    - $*$ CTRL factor has *significative* effect on efficiency: $F_{\texttt{CTRL}} = 21.377 > F_{\texttt{CTRL},crit}$, $p = 3.0777 \times 10^{-12} < \alpha = 0.05$;
    - $*$ DEC factor has *not significative* effect on efficiency: $F_{\texttt{DEC}} = 2.3832 < F_{\texttt{DEC},crit}$, $p = 9.4546 \times 10^{-2} > \alpha = 0.05$.
  - reduced one-factor model shows that CTRL factor has *significative* effect on efficiency: $F''_{\texttt{CTRL}} = 20.8441 > F''_{\texttt{CTRL},crit}$, $p = 5.1487 \times 10^{-12} < \alpha = 0.05$.

Results of non-parametric *Kruskal-Wallis* tests grouping on reduced models are reported by measure.

- *throughput*:
  - grouping by CTRL factor do *not* lead to *significative* differences on the mean rank about throughput: $\chi^2_{\texttt{CTRL}} = 3.7317 < \chi^2_{\texttt{CTRL},crit} = \chi^2_{3;1-\alpha} = 7.8147$, $p = 2.9193 \times 10^{-1} > \alpha = 0.05$;
  - grouping by DEC factor lead to *significative* differences on the mean rank about throughput: $\chi^2_{\texttt{DEC}} = 7.8250 > \chi^2_{\texttt{DEC},crit} = \chi^2_{2;1-\alpha} = 5.9915$, $p = 1.9991 \times 10^{-2} < \alpha = 0.05$.

- *efficiency*:
  - grouping by CTRL factor lead to *significative* differences on the mean rank about efficiency: $\chi^2_{\texttt{CTRL}} = 52.451 > \chi^2_{\texttt{CTRL},crit}$, $p = 2.4001 \times 10^{-11} < \alpha = 0.05$;
  - grouping by DEC factor do *not* lead to *significative* differences on the mean rank about efficiency: $\chi^2_{\texttt{DEC}} = 0.285\,28 < \chi^2_{\texttt{DEC},crit}$, $p = 8.6706 \times 10^{-1} > \alpha = 0.05$.

- *scaled target path*:
  - grouping by CTRL factor lead to *significative* differences on the mean rank about scaled target path: $\chi^2_{\texttt{CTRL}} = 8.0694 > \chi^2_{\texttt{CTRL},crit}$, $p = 4.4599 \times 10^{-2} < \alpha = 0.05$;
  - grouping by DEC factor lead to *significative* differences on the mean rank about scaled target path: $\chi^2_{\texttt{DEC}} = 9.6982 > \chi^2_{\texttt{DEC},crit}$, $p = 7.8354 \times 10^{-3} < \alpha = 0.05$.

- *overshoots*:
  - grouping by CTRL factor lead to *significative* differences on the mean rank about overshoots: $\chi^2_{\texttt{CTRL}} = 28.449 > \chi^2_{\texttt{CTRL},crit}$, $p = 2.9231 \times 10^{-6} < \alpha = 0.05$;
  - grouping by DEC factor do *not* lead to *significative* differences on the mean rank about overshoots: $\chi^2_{\texttt{DEC}} = 0.720\,26 < \chi^2_{\texttt{DEC},crit}$, $p = 6.9759 \times 10^{-1} > \alpha = 0.05$.

For measures on which we performed both tests, these results report as significative factors the same of parametric ANOVA tests. However, this kind of test allows to claim nothing on factors interaction or relative effect size.

Details on the tests performed are reported in tables 6.34, 6.35, 6.38, 6.39, 6.42, 6.43, 6.48, and 6.49 for parametric ANOVA *F-tests*, in tables 6.36, 6.37, 6.40, 6.41, 6.44, 6.45, 6.46, and 6.47 for non-parametric *Kruskal-Wallis* tests.

None of the parametric ANOVA results report a significative effect of the interaction factor on the measures of interest, but at least one between CTRL or DEC factors resulted as significative. Non-parametric tests do not allow to test interaction factor, but gave the same results about the two factors tested alone. In case a factor resulted as significative for a performance measure, we performed pairwise comparison *post-hoc* tests to find out which levels were different from the common mean. As in the previous analysis, we assigned the same letter to the *levels* (i.e. controllers or decoders) for which we had *no evidence* of statistical difference between them, so giving to each level a label formed by one or more letters. This allows to define a partial order on the levels. These labels are reported in the following box-plots, capital letters for parametric ANOVA pairwise tests, lower case letters for Kruskal-Wallis ones.

Box-plots in figures 6.14, 6.15, 6.16, 6.17 and 6.18, show data grouped by CTRL (above) and DEC (below). We report the box-plots also when the factor for which data are grouped was not significative: in this case no labels are shown.

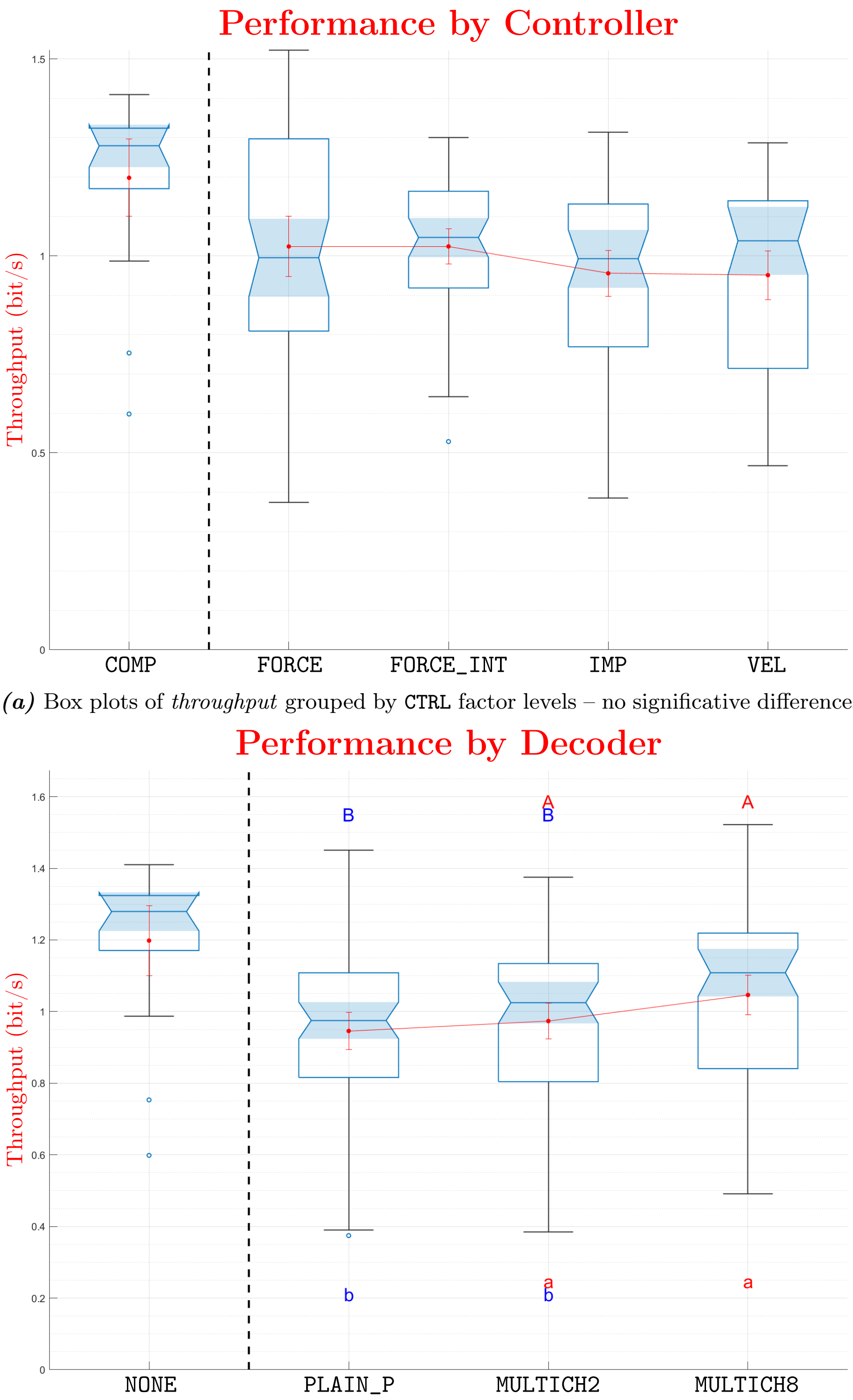
Levels, values, confidence intervals, and labels – sorted from the best to the worst – are reported in tables 6.52, 6.53, 6.54, 6.55, 6.58, 6.59, 6.60, 6.61, 6.63, and 6.66. We report also levels, values and confidence intervals for measures that do not show significative differences in tables 6.50, 6.51, 6.56, 6.57, 6.65, and 6.68, or that are not tested for significative differences in tables 6.62, 6.64, 6.67, and 6.69.

We remember that for *throughput* and *efficiency* greater values are better, while for *scaled target path*, *overshoots*, and *mean IEmg* lower values are better.

Significative pairwise comparisons that lead to labels assigment are reported in tables 6.70, 6.71, 6.72, 6.73, 6.74, 6.75, 6.76, and 6.77. Each row of the tables contain the name of the two levels compared, namely level 1 and 2, the estimated difference $\mu_d = \mu_1 - \mu_2$, 95% confidence interval for μ_d, and the test's p-value. The quantity tested can either be mean variable value or mean group rank. Only pairs for which the null hypothesis is rejected are reported.

Source	SS	DoF	MS	F-value	p-value
CTRL	2.674	3	0.891	2.313	0.077
DEC	3.598	2	1.799	4.669	0.010
CTRL*DEC	2.242	6	0.374	0.969	0.447
Error	87.863	228	0.385		
Total	96.377	239			

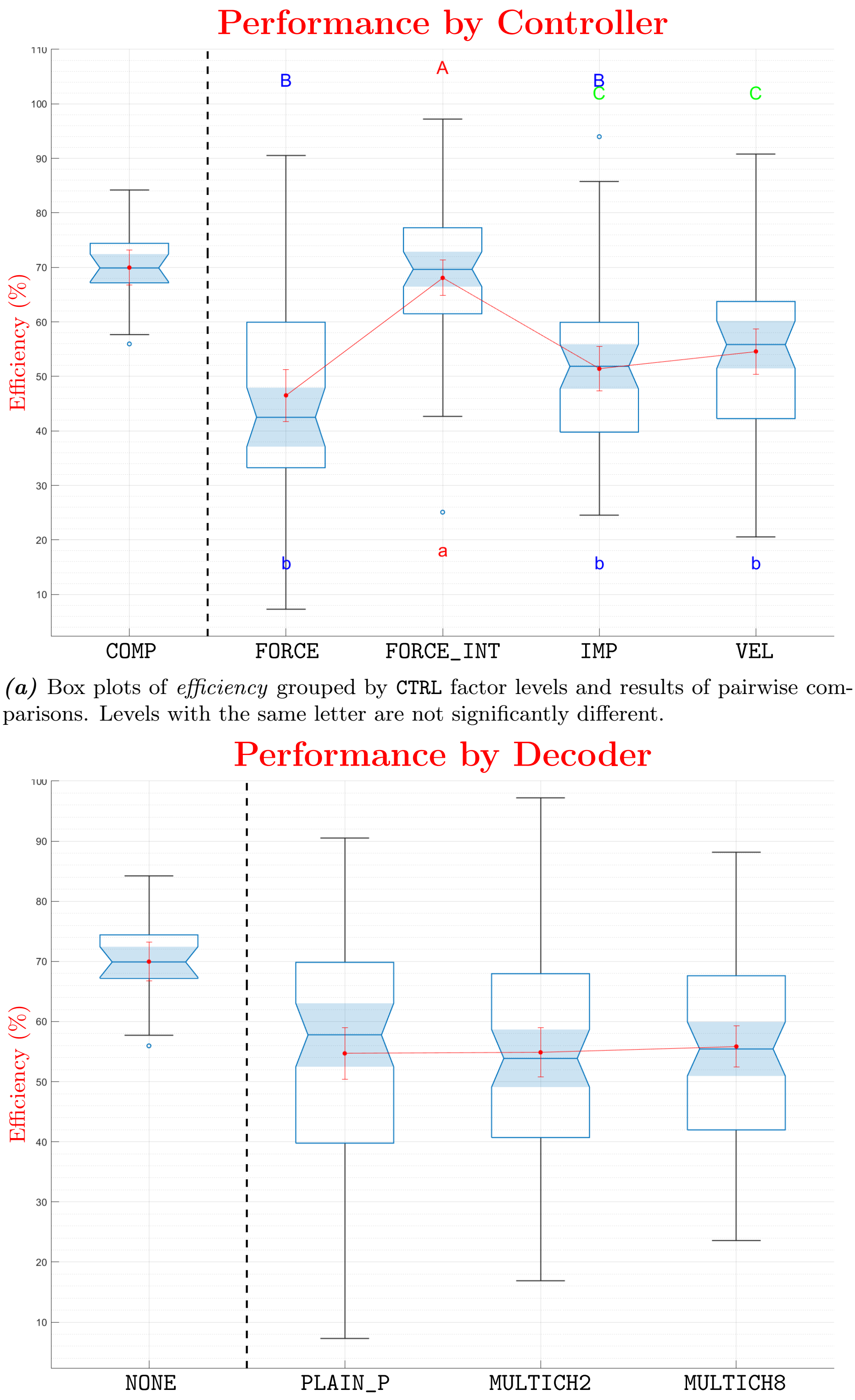
Table 6.34: Two-way ANOVA saturated model of *throughput* (transformed) with CTRL, DEC and their interaction as factors.



(a) Box plots of *throughput* grouped by `CTRL` factor levels – no significative differences.

(b) Box plots of *throughput* grouped by `DEC` factor levels and results of pairwise comparisons. Levels with the same letter are not significantly different.

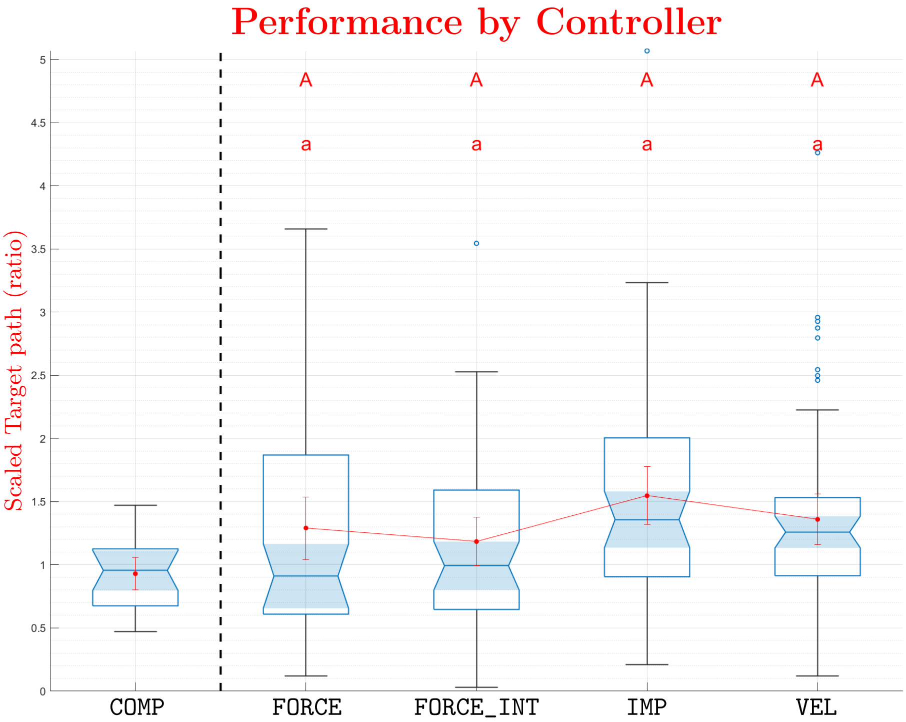
Figure 6.14: Box plots of *throughput* grouped by `CTRL` (a) and by `DEC` factor levels (b). Letters – when present – represent pairwise comparisons results. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.



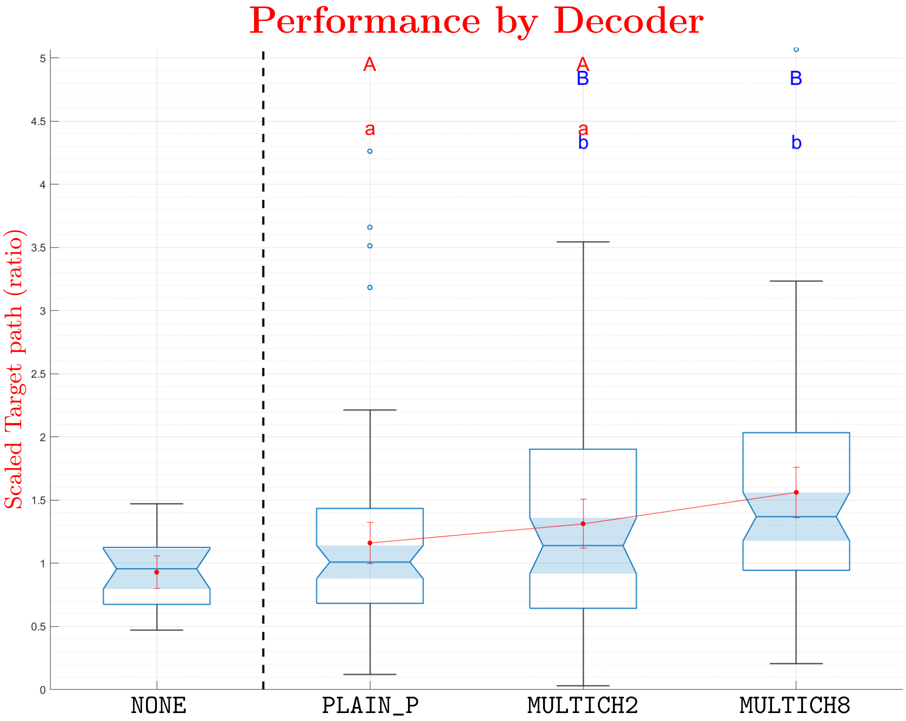
(a) Box plots of *efficiency* grouped by `CTRL` factor levels and results of pairwise comparisons. Levels with the same letter are not significantly different.

(b) Box plots of *efficiency* grouped by `DEC` factor levels – no significative differences.

Figure 6.15: Box plots of *efficiency* grouped by `CTRL` (a) and by `DEC` factor levels (b). Letters – when present – represent pairwise comparisons results. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

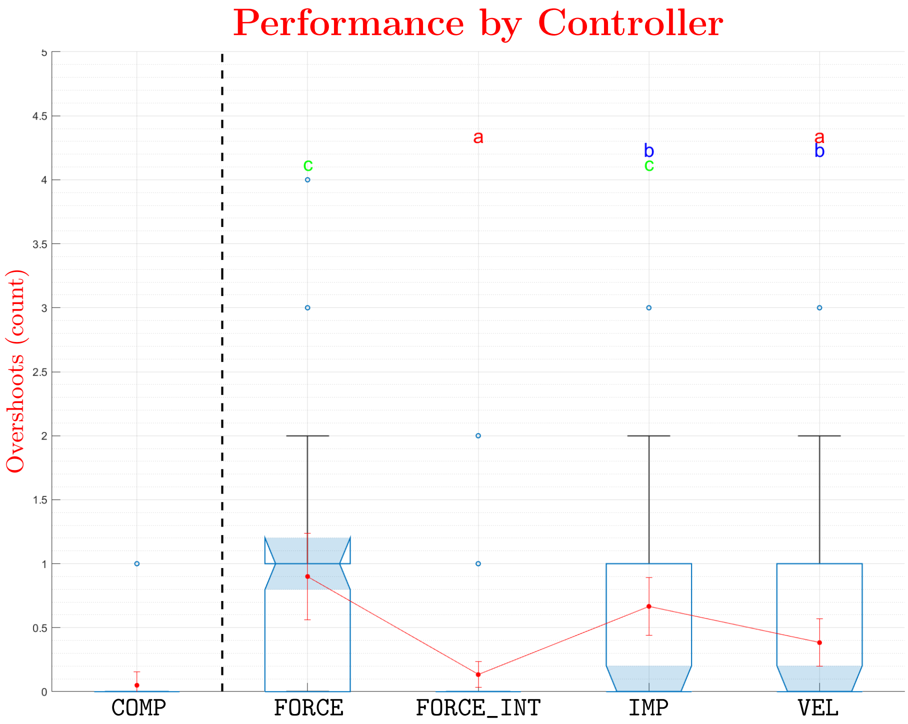


(a) Box plots of *scaled target path* grouped by CTRL factor levels and results of pairwise comparisons. Levels with the same letter are not significantly different.

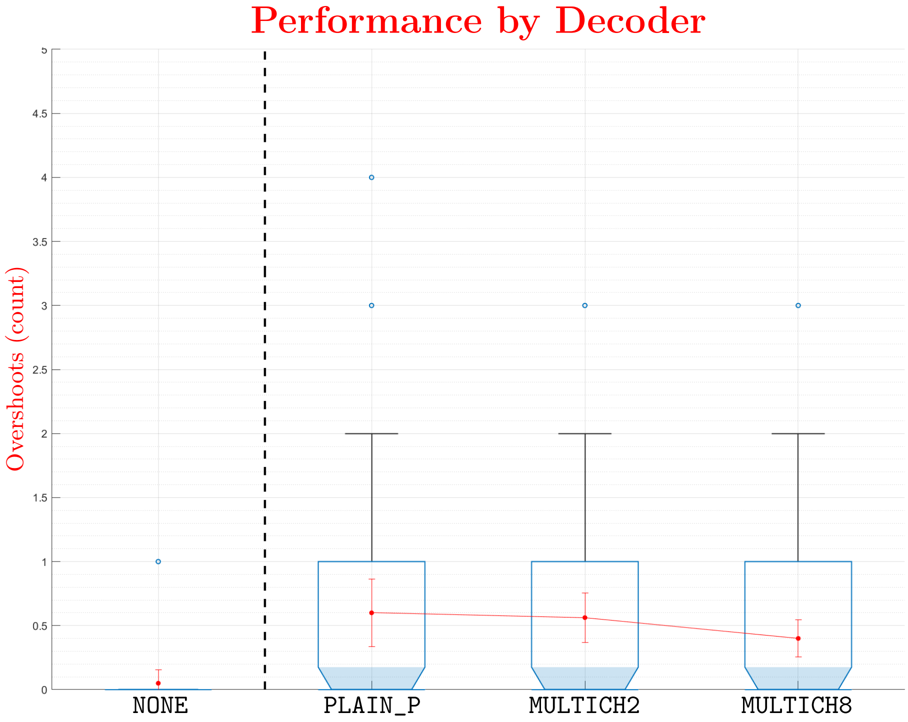


(b) Box plots of *scaled target path* grouped by DEC factor levels and results of pairwise comparisons. Levels with the same letter are not significantly different.

Figure 6.16: Box plots of *scaled target path* grouped by CTRL (a) and by DEC factor levels (b). Letters – when present – represent pairwise comparisons results. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

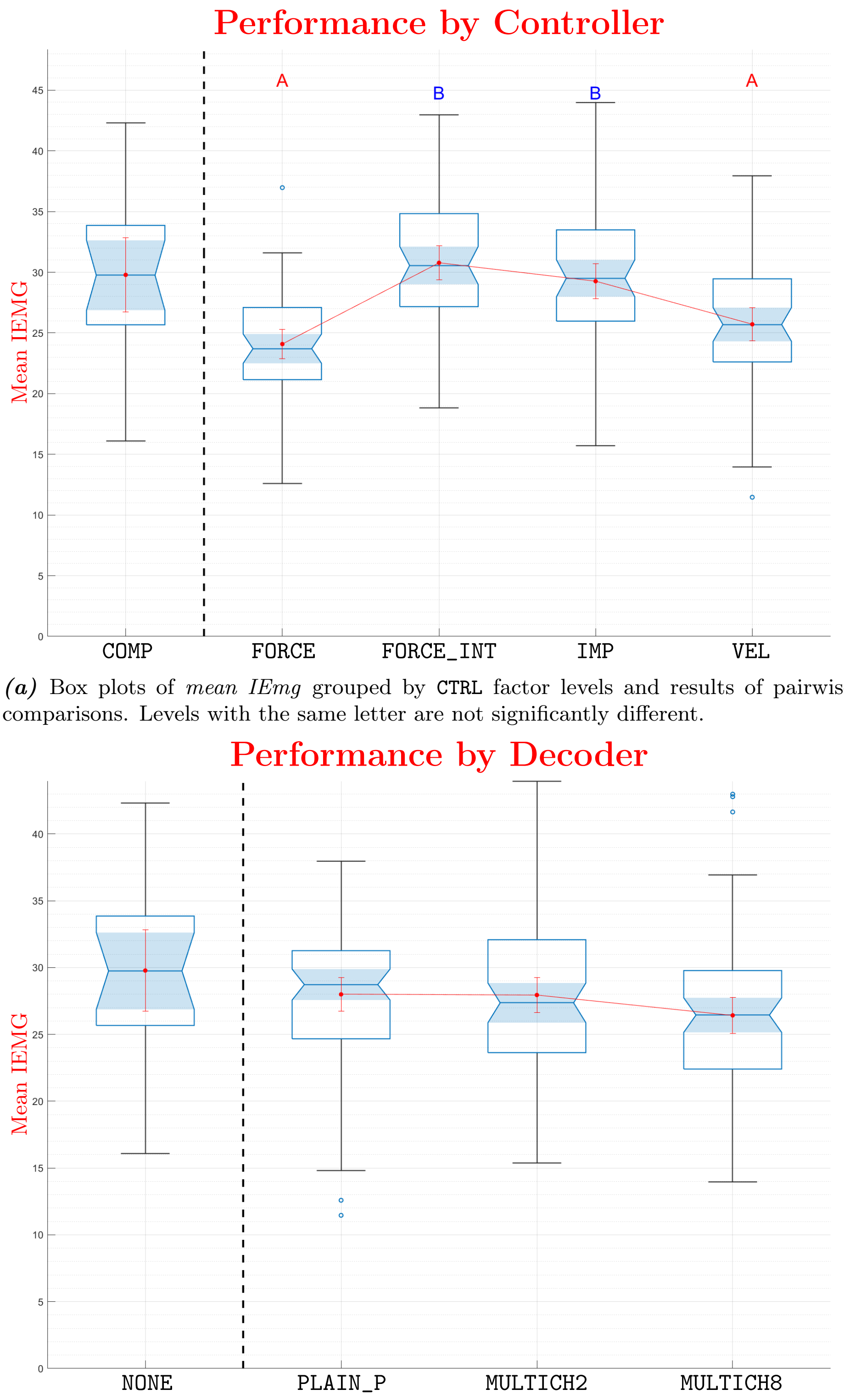


(a) Box plots of *overshoots* grouped by `CTRL` factor levels and results of pairwise comparisons. Levels with the same letter are not significantly different.



(b) Box plots of *overshoots* grouped by `DEC` factor levels – no significative differences.

Figure 6.17: Box plots of *overshoots* grouped by `CTRL` (a) and by `DEC` factor levels (b). Letters – when present – represent pairwise comparisons results. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.



(a) Box plots of *mean IEmg* grouped by `CTRL` factor levels and results of pairwise comparisons. Levels with the same letter are not significantly different.

(b) Box plots of *mean IEmg* grouped by `DEC` factor levels – no significative differences.

Figure 6.18: Box plots of *mean IEmg* grouped by `CTRL` (a) and by `DEC` factor levels (b). Letters – when present – represent pairwise comparisons results. Levels with the same letter are not significantly different. Upper-case letters are used for the results of parametric ANOVA post-hoc, while lower-case ones for Kruskal-Wallis post-hoc.

Source	SS	DoF	MS	F-value	p-value
DEC	3.598	2	1.799	4.596	0.011
Error	92.778	237	0.391		
Total	96.377	239			

Table 6.35: One-way ANOVA reduced model of *throughput* (transformed) with DEC as factor.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$1.799 \cdot 10^4$	3	$5.996 \cdot 10^3$	3.732	0.292
Error	$1.134 \cdot 10^6$	236	$4.805 \cdot 10^3$		
Total	$1.152 \cdot 10^6$	239			

Table 6.36: Kruskal-Wallis test of *throughput* grouped by CTRL – reduced model.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$3.772 \cdot 10^4$	2	$1.886 \cdot 10^4$	7.825	0.020
Error	$1.114 \cdot 10^6$	237	$4.702 \cdot 10^3$		
Total	$1.152 \cdot 10^6$	239			

Table 6.37: Kruskal-Wallis test of *throughput* grouped by DEC – reduced model.

Source	SS	DoF	MS	F-value	p-value
CTRL	15 354.624	3	5 118.208	20.095	<0.001
DEC	61.589	2	30.794	0.121	0.886
CTRL*DEC	1 159.221	6	193.204	0.759	0.603
Error	58 070.426	228	254.695		
Total	74 645.859	239			

Table 6.38: Two-way ANOVA saturated model of *efficiency* with CTRL, DEC and their interaction as factors.

Source	SS	DoF	MS	F-value	p-value
CTRL	15 354.624	3	5 118.208	20.372	<0.001
Error	59 291.236	236	251.234		
Total	74 645.859	239			

Table 6.39: One-way ANOVA reduced model of *efficiency* with CTRL as factor.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$2.528 \cdot 10^5$	3	$8.427 \cdot 10^4$	52.451	<0.001
Error	$8.992 \cdot 10^5$	236	$3.810 \cdot 10^3$		
Total	$1.152 \cdot 10^6$	239			

Table 6.40: Kruskal-Wallis test of *efficiency* grouped by CTRL – reduced model.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$1.375 \cdot 10^3$	2	$6.875 \cdot 10^2$	0.285	0.867
Error	$1.151 \cdot 10^6$	237	$4.855 \cdot 10^3$		
Total	$1.152 \cdot 10^6$	239			

Table 6.41: Kruskal-Wallis test of *efficiency* grouped by DEC – reduced model.

Source	SS	DoF	MS	F-value	p-value
CTRL	2.067	3	0.689	2.709	0.046
DEC	2.244	2	1.122	4.410	0.013
CTRL*DEC	1.512	6	0.252	0.991	0.432
Error	57.999	228	0.254		
Total	63.822	239			

Table 6.42: Two-way ANOVA saturated model of *scaled target path* (transformed) with CTRL, DEC and their interaction as factors.

Source	SS	DoF	MS	F-value	p-value
CTRL	2.067	3	0.689	2.710	0.046
DEC	2.244	2	1.122	4.411	0.013
Error	59.511	234	0.254		
Total	63.822	239			

Table 6.43: Two-way ANOVA independence model of *scaled target path* (transformed) with CTRL and DEC – no interaction – as factors.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$3.889 \cdot 10^4$	3	$1.296 \cdot 10^4$	8.069	0.045
Error	$1.113 \cdot 10^6$	236	$4.716 \cdot 10^3$		
Total	$1.152 \cdot 10^6$	239			

Table 6.44: Kruskal-Wallis test of *scaled target path* grouped by CTRL – reduced model.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$4.675 \cdot 10^4$	2	$2.337 \cdot 10^4$	9.698	0.008
Error	$1.105 \cdot 10^6$	237	$4.663 \cdot 10^3$		
Total	$1.152 \cdot 10^6$	239			

Table 6.45: Kruskal-Wallis test of *scaled target path* grouped by DEC – reduced model.

Source	SS	DoF	MS	χ^2-value	p-value
Groups	$9.643 \cdot 10^4$	3	$3.214 \cdot 10^4$	28.449	<0.001
Error	$7.137 \cdot 10^5$	236	$3.024 \cdot 10^3$		
Total	$8.101 \cdot 10^5$	239			

Table 6.46: Kruskal-Wallis test of *overshoots* grouped by CTRL – reduced model.

Source	SS	DOF	MS	χ^2-value	p-value
Groups	$2.441 \cdot 10^3$	2	$1.221 \cdot 10^3$	0.720	0.698
Error	$8.077 \cdot 10^5$	237	$3.408 \cdot 10^3$		
Total	$8.101 \cdot 10^5$	239			

Table 6.47: Kruskal-Wallis test of *overshoots* grouped by DEC – reduced model.

Source	SS	DOF	MS	F-value	p-value
CTRL	1 722.085	3	574.028	21.377	<0.001
DEC	127.991	2	63.995	2.383	0.095
CTRL*DEC	248.926	6	41.488	1.545	0.164
Error	6 122.315	228	26.852		
Total	8 221.317	239			

Table 6.48: Two-way ANOVA saturated model of *mean IEmg* with CTRL, DEC and their interaction as factors.

Source	SS	DOF	MS	F-value	p-value
CTRL	1 722.085	3	574.028	20.844	<0.001
Error	6 499.232	236	27.539		
Total	8 221.317	239			

Table 6.49: One-way ANOVA reduced model of *mean IEmg* with CTRL as factor.

Level	Mean throughput	Mean 95% CI	
		lower	upper
FORCE_INT	1.024	0.980	1.068
FORCE	1.024	0.947	1.101
IMP	0.956	0.898	1.014
VEL	0.951	0.889	1.013

Table 6.50: Mean *throughput* by CTRL factor levels. Levels are sorted by decreasing mean *throughput*. There are no significative differences between levels.

Level	Median throughput	Median 95% CI	
		lower	upper
FORCE_INT	1.046	1.009	1.094
VEL	1.038	0.913	1.101
FORCE	0.995	0.904	1.142
IMP	0.992	0.899	1.103

Table 6.51: Median *throughput* by CTRL factor levels. Levels are sorted by decreasing median *throughput*. There are no significative differences between levels.

Factor level	Mean `throughput`	Mean 95% CI		Labels	
		lower	upper		
`MULTICH8`	1.046	0.992	1.101	`A`	
`MULTICH2`	0.974	0.923	1.024	`A`	`B`
`PLAIN_P`	0.946	0.894	0.998		`B`

Table 6.52: Mean *throughput* by `DEC` factor levels. Levels are sorted by not significantly different labels and decreasing mean *throughput*.

Factor level	Median `throughput`	Median 95% CI		Labels	
		lower	upper		
`MULTICH8`	1.109	1.006	1.160	`a`	
`MULTICH2`	1.025	0.927	1.096	`a`	`b`
`PLAIN_P`	0.975	0.911	1.035		`b`

Table 6.53: Median *throughput* by `DEC` factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and decreasing median *throughput*.

Factor level	Mean `efficiency`	Mean 95% CI		Labels		
		lower	upper			
`FORCE_INT`	68.076	64.826	71.327	`A`		
`VEL`	54.579	50.423	58.734		`B`	
`IMP`	51.400	47.340	55.459		`B`	`C`
`FORCE`	46.523	41.753	51.292			`C`

Table 6.54: Mean *efficiency* by `CTRL` factor levels. Levels are sorted by not significantly different labels and decreasing mean *efficiency*.

Factor level	Median `efficiency`	Median 95% CI		Labels	
		lower	upper		
`FORCE_INT`	69.672	63.619	71.958	`a`	
`VEL`	55.823	46.417	58.362		`b`
`IMP`	51.815	46.781	54.377		`b`
`FORCE`	42.525	39.607	48.994		`b`

Table 6.55: Median *efficiency* by `CTRL` factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and decreasing median *efficiency*.

Level	Mean `efficiency`	Mean 95% CI	
		lower	upper
`MULTICH8`	55.853	52.426	59.280
`MULTICH2`	54.877	50.786	58.969
`PLAIN_P`	54.702	50.427	58.976

Table 6.56: Mean *efficiency* by `DEC` factor levels. Levels are sorted by decreasing mean *efficiency*. There are no significative differences between levels.

Level	Median efficiency	Median 95% CI	
		lower	upper
PLAIN_P	57.763	49.710	62.930
MULTICH8	55.450	51.266	60.404
MULTICH2	53.886	48.462	58.618

Table 6.57: Median *efficiency* by DEC factor levels. Levels are sorted by decreasing median *efficiency*. There are no significative differences between levels.

Factor level	Mean scaled_target_path	Mean 95% CI		Labels
		lower	upper	
FORCE_INT	1.184	0.993	1.376	A
FORCE	1.292	1.045	1.538	A
VEL	1.360	1.160	1.560	A
IMP	1.547	1.319	1.775	A

Table 6.58: Mean *scaled target path* by CTRL factor levels. Levels are sorted by not significantly different labels and increasing mean *scaled target path*.

Factor level	Median scaled_target_path	Median 95% CI		Labels
		lower	upper	
FORCE	0.911	0.709	1.373	a
FORCE_INT	0.992	0.796	1.280	a
VEL	1.259	1.096	1.372	a
IMP	1.359	1.195	1.630	a

Table 6.59: Median *scaled target path* by CTRL factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and increasing median *scaled target path*.

Factor level	Mean scaled_target_path	Mean 95% CI		Labels	
		lower	upper		
PLAIN_P	1.161	0.998	1.324	A	
MULTICH2	1.314	1.121	1.506	A	B
MULTICH8	1.562	1.362	1.762		B

Table 6.60: Mean *scaled target path* by DEC factor levels. Levels are sorted by not significantly different labels and increasing mean *scaled target path*.

Factor level	Median scaled_target_path	Median 95% CI		Labels	
		lower	upper		
PLAIN_P	1.011	0.853	1.210	a	
MULTICH2	1.139	0.858	1.477	a	b
MULTICH8	1.369	1.232	1.608		b

Table 6.61: Median *scaled target path* by DEC factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and increasing median *scaled target path*.

Level	Mean overshoots	Mean 95% CI	
		lower	upper
FORCE_INT	0.133	0.033	0.234
VEL	0.383	0.199	0.568
IMP	0.667	0.440	0.893
FORCE	0.900	0.561	1.239

Table 6.62: Mean *overshoots* by CTRL factor levels. Levels are sorted by increasing mean *overshoots*. Differences in levels' means have not been tested.

Factor level	Median overshoots	Median 95% CI		Labels
		lower	upper	
FORCE_INT	0	0	0	a
VEL	0	0	0	a b
IMP	0	0	1	b c
FORCE	1	0	1	c

Table 6.63: Median *overshoots* by CTRL factor levels. Levels are sorted by not significantly different labels (Kruskal-Wallis *post-hoc* pairwise comparison) and increasing median *overshoots*.

Level	Mean overshoots	Mean 95% CI	
		lower	upper
MULTICH8	0.400	0.256	0.544
MULTICH2	0.563	0.369	0.756
PLAIN_P	0.600	0.336	0.864

Table 6.64: Mean *overshoots* by DEC factor levels. Levels are sorted by increasing mean *overshoots*. Differences in levels' means have not been tested.

Level	Median overshoots	Median 95% CI	
		lower	upper
PLAIN_P	0	0	0
MULTICH2	0	0	0
MULTICH8	0	0	0

Table 6.65: Median *overshoots* by DEC factor levels. Levels are sorted by increasing median *overshoots*. There are no significative differences between levels.

Factor level	Mean mean_iemg	Mean 95% CI		Labels
		lower	upper	
FORCE	24.084	22.867	25.302	A
VEL	25.712	24.351	27.072	A
IMP	29.261	27.825	30.697	B
FORCE_INT	30.777	29.379	32.175	B

Table 6.66: Mean *mean IEmg* by CTRL factor levels. Levels are sorted by not significantly different labels and increasing mean *mean IEmg*.

Level	Median `mean_iemg`	Median 95% CI	
		lower	upper
`FORCE`	23.703	22.498	25.909
`VEL`	25.695	24.040	27.336
`IMP`	29.502	27.948	31.963
`FORCE_INT`	30.556	28.834	31.706

Table 6.67: Median *mean IEmg* by `CTRL` factor levels. Levels are sorted by increasing median *mean IEmg*. Differences in levels' medians have not been tested.

Level	Mean `mean_iemg`	Mean 95% CI	
		lower	upper
`MULTICH8`	26.426	25.079	27.773
`MULTICH2`	27.950	26.652	29.248
`PLAIN_P`	27.999	26.744	29.254

Table 6.68: Mean *mean IEmg* by `DEC` factor levels. Levels are sorted by increasing mean *mean IEmg*. There are no significative differences between levels.

Level	Median `mean_iemg`	Median 95% CI	
		lower	upper
`MULTICH8`	26.447	24.434	27.847
`MULTICH2`	27.363	26.251	29.351
`PLAIN_P`	28.733	27.334	29.769

Table 6.69: Median *mean IEmg* by `DEC` factor levels. Levels are sorted by increasing median *mean IEmg*. Differences in levels' medians have not been tested.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
`PLAIN_P`	`MULTICH8`	−0.103	−0.195	−0.019	0.010

Table 6.70: Significative ($p < \alpha$) pairwise comparisons between `DEC` factor levels on *throughput* (de-transformed). Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
`PLAIN_P`	`MULTICH8`	−29.925	−55.652	−4.198	0.018

Table 6.71: Significative ($p < \alpha$) pairwise comparisons between `DEC` factor levels on *throughput*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
FORCE	FORCE_INT	−21.554	−28.988	−14.119	<0.001
FORCE	VEL	−8.056	−15.491	−0.622	0.028
FORCE_INT	IMP	16.677	9.242	24.111	<0.001
FORCE_INT	VEL	13.497	6.063	20.932	<0.001

Table 6.72: Significative ($p < \alpha$) pairwise comparisons between CTRL factor levels on *efficiency*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
FORCE	FORCE_INT	−86.350	−118.914	−53.786	<0.001
FORCE_INT	IMP	69.683	37.120	102.247	<0.001
FORCE_INT	VEL	56.767	24.203	89.330	<0.001

Table 6.73: Significative ($p < \alpha$) pairwise comparisons between CTRL factor levels on *efficiency*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
PLAIN_P	MULTICH8	−0.331	−0.557	−0.059	0.014

Table 6.74: Significative ($p < \alpha$) pairwise comparisons between DEC factor levels on *scaled target path* (de-transformed). Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
PLAIN_P	MULTICH8	−33.563	−59.290	−7.835	0.006

Table 6.75: Significative ($p < \alpha$) pairwise comparisons between DEC factor levels on *scaled target path*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
FORCE	FORCE_INT	51.775	24.467	79.083	<0.001
FORCE	VEL	32.642	5.334	59.949	0.011
FORCE_INT	IMP	−41.392	−68.699	−14.084	0.001

Table 6.76: Significative ($p < \alpha$) pairwise comparisons between CTRL factor levels on *overshoots*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' mean ranks. These are the result of Kruskal-Wallis post-hoc tests.

Level 1	Level 2	Est. difference	Difference 95% CI		p-value
			lower	upper	
FORCE	FORCE_INT	−6.693	−9.155	−4.232	<0.001
FORCE	IMP	−5.177	−7.638	−2.715	<0.001
FORCE_INT	VEL	5.066	2.604	7.527	<0.001
IMP	VEL	3.549	1.088	6.011	0.001

Table 6.77: Significative ($p < \alpha$) pairwise comparisons between CTRL factor levels on *mean IEmg*. Each comparison tests the *null* hypothesis that $H_0 : \mu_1 - \mu_2 = 0$ and reports the estimated difference in levels' means. These are the result of parametric ANOVA post-hoc tests.

6.2.4 Results discussion

To find the answers to the scientific questions posed in subsection 5.3.2, we performed some statistical tests on the performance measures we collected. The results allow to conclude:

1. the task performed is able to show differences between the different myoelectric interfaces w.r.t. all the measures collected, since both parametric and non-parametric ANOVA tests found evidence of an effect of the selection of controller and decoder pair on measures outcome (tables 6.8, 6.9, 6.10, 6.11, 6.12, 6.13, 6.14, and 6.15).

2. *post-hoc* tests on the model with the `CTRL+DEC` factor as fixed effect, revealed that the interface based gravity compensation is always between the best score levels for *throughput* (tables 6.16 and 6.17), *efficiency* (tables 6.18 and 6.19), *scaled target path* (tables 6.20 and 6.21), and *overshoots* (table 6.23), but one of the worse levels w.r.t. *mean IEmg* (table 6.24): it is the best interface at which other myoelectric interfaces aim from the point of view of the task performance, but at the price of high effort, so the need of less-effort myoelectric interfaces.

3. the two-way ANOVA analysis gave an insight on how the components of the myoelectric control architecture work and how they affect the overall performance. None of the measure tested with parametric two-way ANOVA showed a significative interaction between controller and decoder, allowing to threat them separately. Only the decoder has a significative effect on *throughput* (tables 6.34, 6.35, and 6.37), while only the controller has a significative effect on *efficiency* (tables 6.38, 6.39, and 6.40), *overshoots* (table 6.46), and *mean IEmg* (tables 6.48 and 6.49). The case of *scaled target path* is different: both controller and decoder have a significative effect (tables 6.42, 6.43, 6.44 and 6.45), but the effect size is such small that *post-hoc* pairwise test on the controller levels was not able to show any difference between them (tables 6.58 and 6.59).

Analysing in detail the results of *post-hoc* tests on the model with the controller and decoder pair (`CTRL+DEC`) factor as fixed effect, we made some observations looking for the best pairs, aiming at those not significantly different from gravity compensation.

- *Throughput*, from figure 6.9, tables 6.16 and 6.17 sorted by significative labels and, respectively, mean or median group value, tables 6.26 and 6.27 for significative pairwise comparisons. There are no big differences between the controllers, a part from `VEL` that independently from the decoder performs always significantly worse than gravity compensation. On the other hand, `MULTICH8` decoder 3 times out of 4 is not significantly different from gravity compensation, while both `MULTICH2` and `PLAIN_P` 3 times out of 4 we found evidence of a worst behaviour w.r.t. gravity compensation. This is an indication that *throughput* is deeply affected by the quality of the decoder's prediction.

- *Efficiency*, from figure 6.10, tables 6.18 and 6.19 sorted by significative labels and, respectively, mean or median group value, tables 6.28 and 6.29 for significative pairwise comparisons. Looking at the box-plots, it appears evident a clear dependency on the controller component: in pair with 3 decoders out of 3 `FORCE_INT` controller is not significantly different from gravity compensation, `VEL` controller in pair with 2 decoders, `IMP` controller only in pair with `MULTICH2` decoder, while all of the pairs having `FORCE` as controller are significantly worse than gravity compensation. On the other hand, the choice of the decoder appears not to have the same influence on the *efficiency* outcome. This is an indication that *efficiency* is more dependent on the interpretation of decoder's output made by the controller rather than on the decoder's precision.

- *Scaled target path*, from figure 6.11, tables 6.20 and 6.21 sorted by significative labels and, respectively, mean or median group value, tables 6.30 and 6.31 for significative pairwise comparisons. About this measure, there are no evident differences or trends between the interfaces, a part from the pair controller and decoder `IMP+MULTICH8` having a significative higher value w.r.t. `COMP`, `FORCE+MULTICH2`, and `FORCE_INT+PLAIN_P`. Although this measure shows some minimal differences between levels, in these experiments seems not to be particularly discriminative among levels.

- *Overshoots*, from figure 6.12, table 6.23 sorted by significative labels and median group value, table 6.32 for significative pairwise comparisons. Also in this case the measure does not seem particularly discriminative, but two groups can be identified: those interfaces with almost no overshoots (controllers `COMP` and `FORCE_INT`, and pair `VEL+PLAIN_P`), and the others.

- *Mean IEmg*, from figure 6.13, table 6.24 sorted by significative labels and mean group value, table 6.33 for significative pairwise comparisons. This measure appears very discriminative – we are able to assign 5 letters, representing value levels – and differences among controllers are evident. Looking at the pairs having the same controller component, we can see a trend of improvement as we increase the precision of the decoder, evident for `FORCE` and `VEL` controllers. It's worth noticing this is the only measure for which at least an interface shows a significantly better behaviour than gravity compensation: these are the `FORCE+MULTICH8` and the `VEL+MULTICH8` interface pairs.

Here we report the detailed analysis of *post-hoc* test results on the two-way model with the controller and the decoder (`CTRL` and `DEC`) factors as different fixed effects. These observations are coherent with the ones about the one-way model, but allow to distinguish between the contributions of the two components. However, the comparison with gravity compensation cannot directly done, because the inclusion of gravity compensation tests would have made the experiment design unbalanced.

- *Throughput*, from figure 6.14 and table 6.34, we can see that only the decoder factor has a significant effect on the measure. Therefore, tables 6.52 and 6.53 show the `DEC` levels sorted by significative labels and, respectively, mean or median group value, while tables 6.70 and 6.71 report the significative pairwise comparisons between decoder's levels. The best decoder is `MULTICH8`, with an estimated increase of mean *throughput* respect to `PLAIN_P` of $0.103\,\mathrm{bit\,s^{-1}}$ with 95% $CI = [0.019, 0.195]\ \mathrm{bit\,s^{-1}}$. This is another indication that *throughput* is more affected by the quality of the decoder's prediction rather than the type of controller.

- *Efficiency*, from figure 6.15 and table 6.38, we can see that only the controller factor has a significant effect on the measure. Therefore, tables 6.54 and 6.55 show the `CTRL` levels sorted by significative labels and, respectively, mean or median group value, tables 6.72 and 6.73 report the significative pairwise comparisons between controller's levels. The best controller is `FORCE_INT`, with an estimated increase of mean *efficiency* respect to `FORCE` of 21.6 with 95% $CI = [14.1, 29.0]$, to `IMP` of 16.7 with 95% $CI = [9.24, 24.1]$, and to `VEL` of 13.5

with 95% $CI = [6.06, 20.9]$. Again, this is an indication that *efficiency* depends more on the controller rather than the decoder.

- *Scaled target path*, from figure 6.16 and tables 6.42 and 6.43, we can see that both the controller and the decoder factors have a significant effect on the measure. Therefore, tables 6.58, 6.59, 6.60 and 6.61 show, respectively, the `CTRL` and the `DEC` levels sorted by significative labels and mean or median group value, tables 6.74 and 6.75 report the significative pairwise comparisons between decoder's levels. The `CTRL` factor, though having a significative effect on this measure, this effect is too small to identify significative differences between levels at the pairwise *post-hoc* tests. Regarding instead the `DEC` factor, the best decoder appears to be `PLAIN_P`, with an estimated decrease of mean *scaled target path* respect to `MULTICH8` of 0.331 with 95% $CI = [0.059, 0.557]$. These results further support the hypothesis that this measure is not particularly discriminative of interface quality.

- *Overshoots*, from figure 6.17 and tables 6.46 and 6.47, we can see that only the controller factor has a significant effect on the measure. We cannot, however, exclude the presence of interaction between factors, since non-parametric analysis we performed cannot test for this hypothesis. Table 6.63 shows the `CTRL` levels sorted by significative labels and median group value, then table 6.76 report the significative pairwise comparisons between controller's levels. The best controller is `FORCE_INT`, with an estimated decrease of mean rank about *overshoots* respect to `FORCE` of 51.8 with 95% $CI = [24.5, 79.1]$ and to `IMP` of 41.4 with 95% $CI = [14.1, 68.7]$ and estimated *overshoots* mean value for `FORCE_INT` of 0.133 with 95% $CI = [0.033, 0.234]$, for `VEL` of 0.383 with 95% $CI = [0.199, 0.568]$, for `IMP` of 0.667 with 95% $CI = [0.440, 0.893]$, and for `FORCE` of 0.900 with 95% $CI = [0.561, 1.24]$. As we can see from the mean group values, the difference about *overshoots* between controllers is very little and the differences between decoders are not significant. This confirms again the scarce ability of the measure to discriminate between interfaces.

- *Mean IEmg*, from figure 6.18 and table 6.48, we can see that only the controller factor has a significant effect on the measure. Table 6.66 shows the `CTRL` levels sorted by significative labels and mean group value, while table 6.77 reports the significative pairwise comparisons between controller's levels. This measure is able to discriminate between two separated groups, statistically different from each other. The first group is composed by `FORCE` and `VEL` controllers, with mean values lower than gravity compensation, and the second by `FORCE_INT` and `IMP`, with mean values comparable to that of gravity compensation. In the first group we have:

  - controller `FORCE`, with an estimated *mean IEmg* mean value of 24.1 a u with 95% $CI = [22.9, 25.3]$ a u and an estimated decrease of mean value respect to `FORCE_INT` of 6.69 a u with 95% $CI = [4.23, 9.16]$ a u and to `IMP` of 5.18 a u with 95% $CI = [2.72, 7.64]$ a u ;
  - controller `VEL`, with an estimated *mean IEmg* mean value of 25.7 a u with 95% $CI = [24.4, 27.1]$ a u and an estimated decrease of mean value respect to `FORCE_INT` of 5.07 a u with 95% $CI = [2.60, 7.53]$ a u and to `IMP` of 3.55 a u with 95% $CI = [1.09, 6.01]$ a u .

CHAPTER 7

Conclusion and future work

Controlling assistive orthoses requires to understand the human volition. Existing assistive devices suffer from the lack of proper methods of intention recognition, especially if they are intended for motor impaired people with limited capacities of exerting forces to *physically* interact with the device. This work focuses on the case of people suffering from muscular weakness, with the objective of providing a *myoelectric control architecture* onto which natural and intuitive interfaces for assistance in ADLs can be developed. In particular, people who can adopt such interfaces are subjects for which the control on peripheral nervous system (PNS) is intact, e.g. muscular dystrophy, myopathies, ageing, etc. In these conditions, the electrical activity of motoneurons and muscles encodes information on human volition and it can be measured non-invasively from the skin surface using electrodes, a technique known as surface electromyography (sEMG). Unfortunately, the signals recorded with the sEMG are difficult to be used as control commands: clinical applications revealed that sEMG is unreliable, badly conditioned, and dependent on time, fatigue, and sweat [9].

Here, the specific *myoelectric interface* is defined considering a generic architecture composed by a cascading combination of signal processing algorithms, respectively, a *decoder*, an *adapter* or post-processing module, and a low-level *controller*, together with their tuning.

The *decoder* is responsible of inferring the movement volition starting from the sEMG recordings. Many *decoders* have been proposed and they can be divided in two classes based on their outputs:

discrete that output only a label representing the intended action in a predefined set (qualitative control);

proportional – usually to *velocity* or to *torque* – that output a continuous velocity or torque control either in task or joint space.

In this work only *proportional* decoders are considered, since they give more control on the device.Even if there exists a huge literature on such decoders, most of them are intended for prostheses control, with several products already on the market. Unlike robotic prostheses, exoskeletons need coherence with physiological activations, as they act in parallel with the human body. Decoders to control exoskeletons are still associated to technological barriers and to basic scientific questions and therefore confined to laboratory prototypes usually trained and tested only on healthy subjects – except for few cases [11, 39]. Moreover, most of the existing decoders have been applied to rehabilitation devices or to lower limbs control, where the cognitive and volitional contribution is usually minor. In fact, assisting a person during well-defined rehabilitation tasks or in repetitive walking pattern can be considered significantly easier than understanding the wearer intention and synergically providing assistance during ADLs, which typically require a more complex cognitive plan.

A wealth of models and methodologies has been used to solve the proportional decoding problem, ranging from linear proportional to non-linear regressions and neural networks, from state-space to biomechanical models and muscle synergies, also using innovative sensing techniques as high-density EMG (HD-EMG). In this work:

- a *myoelectric control architecture* is proposed to describe, classify and compare existing myoelectric interfaces that has been applied to the case of study of linear regression decoders and 1-Dof elbow controllers.

- methodologies to automatically tune gravity compensation and to collect training data for subjects suffering from severe disabilities are proposed.

- task-based performance measures are proposed to evaluate and compare different myoelectric interfaces and the methodology is applied to the case of study.

We designed an experimental test bed to implement our case of study interfaces, including an elbow exoskeleton and a software framework able to make the sEMG signals available to the *myoelectric control architecture*. The research goal of this thesis is to investigate how different *myoelectric interfaces* can affect the final performance of EMG-driven exoskeletons, limited to this scenario. The results presented in this thesis allow to conclude that EMG-based control is a viable technology to assist muscular weakness patients, and that all the modules composing the *myoelectric control architecture*, decoder, adapter, controller, and tuning, significantly affect the final performance, as we measure them.

Future work includes:

- a methodology to automatically tune all the parameters of the interfaces proposed, not the decoders' ones only, both with the dataset of isometric torques and with the proposed (but not used) *computer-guided* methodology.

- the analysis of the adapter's quantitative effect on the regulation of the assistance level and of the tradeoff between speed and accuracy.

Specifically on this last point, model-driven as well as data-driven approaches can employed. However, from the data collected on out test bed it appears that in a reduced complexity situation settings such as elbow control, the system behavior can be described as a second order linear dynamics.

APPENDIX A

Mathematical Tools

A.1 Ordinary Least-Squares (OLS) regression

OLS [53] is a linear regression technique that given a set of observations $(y_i, \boldsymbol{x}_i)$ with $i = 1, \ldots, N$ where N is the number of observation, models the scalar response y_i as a linear combination of the m predictors or regressors x_{ij} in the column vector $\boldsymbol{x}_i$:

$$y_i = \sum_{j=1}^{m} w_j x_{ij} + \varepsilon_i \tag{A.1}$$

If the N predictor observations are stored as rows of a $N \times m$ matrix $\boldsymbol{X}$ and the scalar responses in a N-dimensional vector $\boldsymbol{y}$, OLS can be viewed as the problem of solving the overdetermined system $\boldsymbol{X}\boldsymbol{w} = \boldsymbol{y}$ in the least-square sense, e.g. minimizing the objective function $S(\boldsymbol{w}) = \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|^2$. If the model also contains an offset w_0, it is sufficient to add to the regessors set the constant scalar value 1, thus solving the system:

$$\widetilde{\boldsymbol{X}}\widetilde{\boldsymbol{w}} = \boldsymbol{y} \tag{A.2}$$

where $\widetilde{\boldsymbol{X}} = [\boldsymbol{1}_{N\times 1}\boldsymbol{X}]$ and $\widetilde{\boldsymbol{w}}^T = [w_0\boldsymbol{w}^T]$. The solution to this problem is given by:

$$\begin{bmatrix} \hat{w}_0 \\ \hat{\boldsymbol{w}} \end{bmatrix} = (\widetilde{\boldsymbol{X}}^T\widetilde{\boldsymbol{X}})^{-1}\widetilde{\boldsymbol{X}}^T\boldsymbol{y} \tag{A.3}$$

The values $\hat{w}_0$ and $\hat{\boldsymbol{w}}$ are called the *OLS estimators* of w_0 and $\boldsymbol{w}$, i.e the values that minimize the error function S, that is also the *maximum likelihood estimator* (MLE) under the normality assumption for the error terms.

A.2 Principal Component Analysis (PCA)

Principal Component Analysis [31] is a common tool that allows to compress m-dimensional data vectors along few directions or dimensions that account for most of the variability. Firstly, we must compute the sample covariance or scatter matrix $\boldsymbol{S}$:

$$\boldsymbol{S} = \frac{1}{k-1} \sum_{i=1}^{k} (\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^T \tag{A.4}$$

where k is the number of samples, $\overline{\boldsymbol{x}}$ is the sample mean and $\boldsymbol{x}_i$ is the i^{th} sample. The normalization factor $\frac{1}{k-1}$ is usually omitted since it is only a scaling factor. Since the scatter matrix S is symmetric (i.e. $S^T = S$), it is orthogonally diagonalizable with all real eigenvalues: so there exist real values $\lambda_1, \ldots, \lambda_m$ and orthogonal non-zero real eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ such that for each $i = 1, \ldots, m$:

$$\boldsymbol{S}\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i \tag{A.5}$$

As the scatter matrix can be viewed as the product of a matrix by its transpose $S = BB^T$ with B an $m \times k$ matrix having as columns the vectors $\boldsymbol{x}_i - \overline{\boldsymbol{x}}$, its eigenvalues are all non-negative. If $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m$ are the eigenvalues of S, the corresponding orthonormal eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ are called the *principal components* of the data set. Since the scatter matrix has on its diagonal the variance of each dimension, the sum of these values, the *trace* T of S, represents the total variance of the data set. As the trace of a orthogonally diagonalizable matrix is also the sum of its eigenvalues $T = \lambda_1 + \ldots + \lambda_m$, we can say that a fraction $\frac{\lambda_i}{T}$ of the total variance T lives along the direction spanned by $\boldsymbol{u}_i$. Since it is often the case that few eigenvalues of S are largely greater than all the others, it is possible that the eigenvectors corresponding to the first $n < m$ largest eigenvalues could explain most of the variability of the data set.Under this assumption, projecting the samples onto the n-dimensional subspace spanned by $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$, only a minor part of the original information is lost. Considering an $m \times n$ matrix $W = [\boldsymbol{u}_1 | \ldots | \boldsymbol{u}_n]$, it is possible to compute a reduced feature vector $\boldsymbol{f}_{red}(t)$:

$$\boldsymbol{f}_{red}(t) = \boldsymbol{W}^T(\boldsymbol{f}(t) - \overline{\boldsymbol{x}}) \tag{A.6}$$

where the size of $\boldsymbol{f}_{red}(t)$ is $n < m$ and $\boldsymbol{f}(t)$ the full-size m-dimensional feature vector.

A.3 Non-Negative Matrix Factorization (NNMF)

Non-Negative Matrix Factorization [35] is a technique that given a set of multivariate m-dimensional non-negative data vectors, these are placed in the columns of an $m \times k$ matrix $\boldsymbol{V}$ where k is the number of samples in the data set. The non-negative matrix $\boldsymbol{V}$ is then approximately factorized into two non-negative matrix factors $\boldsymbol{W}$ $m \times n$ and $\boldsymbol{H}$ $n \times k$ such that:

$$\boldsymbol{V} \approx \boldsymbol{W}\boldsymbol{H} \tag{A.7}$$

minimizing a distance cost function. A common choice is to minimize the square of the Euclidean distance $\|\boldsymbol{V} - \boldsymbol{W}\boldsymbol{H}\|^2$ between non-negative matrices of same size $\boldsymbol{V}$ and $\boldsymbol{W}\boldsymbol{H}$, where the square of the Euclidean distance between non-negative matrices of same size $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as:

$$\|\boldsymbol{A} - \boldsymbol{B}\|^2 = \sum_{ij} (a_{ij} - b_{ij})^2 \tag{A.8}$$

Usually n is chosen to be smaller than m or k, thus the result is a compressed version of the original data matrix. If we rewrite the product column by column as $\boldsymbol{v} \approx \boldsymbol{W}\boldsymbol{h}$, we can see that each data vector $\boldsymbol{v}$ is approximated by a linear combination of the columns of $\boldsymbol{W}$, weighted by the components of $\boldsymbol{h}$. Thus the resulting columns of $\boldsymbol{W}$ can be considered as the generators of a feature subspace of dimension $n < m$ [73]. The reduced feature vector $\boldsymbol{f}_{red}(t)$ of size n than can be computed starting from the full size m-dimensional feature vector $\boldsymbol{f}(t)$ as:

$$\boldsymbol{f}_{red}(t) = (\boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{f}(t) \tag{A.9}$$

where the Euclidean distance:

$$S = \|\boldsymbol{f}(t) - \boldsymbol{W}\boldsymbol{f}_{red}(t)\|^2 \tag{A.10}$$

is minimized, so a least squares problem is solved. It is not a simple subspace projection: instead, it is equivalent to solving the NNMF problem $\boldsymbol{f}(t) \approx \boldsymbol{W}\boldsymbol{f}_{red}(t)$ with $\boldsymbol{W}$ fixed and using as cost function the square of the Euclidean distance S.

A.4 Fixed effects ANOVAs

The notions of statistical analysis of experiments are took from the popular handbook [62].

One-way fixed effects ANOVA can be employed when population can be grouped by *one* factor of interest. Let A be the factor with a levels, the model analysed can be written as:

$$y_{i,j} = \mu + \alpha_i + \varepsilon_{i,j} \tag{A.11}$$

where:

- $y_{i,j}$ is the observed value, or *response*, to the i-th level of factor A in the j-th replicate, for a total of n observations per group, if all groups are of equal size, n_i otherwise.

- μ is the overall population mean.

- α_i is the deviation of the mean of group i (considered *fixed*, from this the model name) from the global mean μ. It is usually called the fixed effect of factor A at level i. The *null* hypothesis states that all α_i equal zero; the alternate hypothesis is that some or all α_i are non-zero, and their value is fixed.

- $\varepsilon_{i,j}$ represents random deviation of j-th replicate of i-th level from $\mu + \alpha_i$, also called random error effect. Usually parametric ANOVA model assumes all $\varepsilon_{i,j}$s to be random normal variables with zero mean and same variance.

Two-way (N-way) fixed effects ANOVA can be employed when population can be grouped by *two* (or more) factor of interest. Let A and B be factors with respectively a and b levels, the model analysed can be written as:

$$y_{i,j,k} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{i,j} + \varepsilon_{i,j,k} \tag{A.12}$$

where:

- $y_{i,j,k}$ is the response to the i-th level of factor A and the j-th level of factor B in the k-th replicate, for a total of n observations per each combination (i, j), if all groups are of equal size, $n_{i,j}$ otherwise.

- μ is the overall population mean.

- α_i is called the fixed effect of factor A at level i. The *null* hypothesis on factor A states that all α_i equal zero.
- β_j is called the fixed effect of factor B at level j. The *null* hypothesis on factor B states that all β_j equal zero.
- $(\alpha\beta)_{i,j}$ is called the fixed effect of the interaction of the combination (i, j). The *null* hypothesis on interaction states that all α_i equal zero.
- $\varepsilon_{i,j,k}$ represents random deviation of k-th individual of the combination (i, j) from the mean $\mu+\alpha_i+\beta_j+(\alpha\beta)_{i,j}$, the random error effect. Parametric ANOVA model assumes all $\varepsilon_{i,j,k}$s to be random normal variables with zero mean and same variance.

The model A.4 with interactions is called a *saturated model*. Usually, the hypothesis $H_0 : \forall(i, j)(\alpha\beta)_{i,j} = 0$ is tested first. If rejected, interaction effects are significant. There is a school of thought that in this case considers main effects of no importance, even if tested significant. When, however, H_0 is not rejected, we can consider a *submodel* without interaction:

$$y_{i,j,k} = \mu + \alpha_i + \beta_j + \varepsilon_{i,j,k} \tag{A.13}$$

called *independence model*.

In this case, testing for $H_0 : \forall i\alpha_i = 0$ and $H_0 : \forall j\beta_j = 0$ is of importance to interpret reduced model A.4. If the effect of only one factor is significant (factor A, for example), then the model can be further reduced to *one-way* model with a factor levels and bn replicates each:

$$y_{i,j,k} = \mu + \alpha_i + \varepsilon_{i,j,k}. \tag{A.14}$$

Bibliography

[1] Chris Wilson Antuvan, Mark Ison, and Panagiotis Artemiadis. "Embedded Human Control of Robots Using Myoelectric Interfaces". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 22.4 (July 2014), pp. 820–827. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2014.2302212.

[2] Panagiotis K. Artemiadis and Kostas J. Kyriakopoulos. "EMG-based teleoperation of a robot arm using low-dimensional representation". In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 26. 2. IEEE, Oct. 2007, pp. 489–495. ISBN: 978-1-4244-0911-2. DOI: 10.1109/IROS.2007.4399452.

[3] P.K. Artemiadis and K.J. Kyriakopoulos. "EMG-Based Control of a Robot Arm Using Low-Dimensional Embeddings". In: *IEEE Transactions on Robotics* 26.2 (Apr. 2010), pp. 393–398. ISSN: 1552-3098. DOI: 10.1109/TRO.2009.2039378.

[4] Mohammadreza Asghari Oskoei and Huosheng Hu. "Myoelectric control systems—A survey". In: *Biomedical Signal Processing and Control* 2.4 (Oct. 2007), pp. 275–294. ISSN: 17468094. DOI: 10.1016/j.bspc.2007.07.009.

[5] Manfredo Atzori, Matteo Cognolato, and Henning Müller. "Deep Learning with Convolutional Neural Networks Applied to Electromyography Data: A Resource for the Classification of Movements for Prosthetic Hands". In: *Frontiers in Neurorobotics* 10.SEP (Sept. 2016), pp. 1–10. ISSN: 1662-5218. DOI: 10.3389/fnbot.2016.00009. arXiv: 1406.1231.

[6] Thiago Boaventura and Jonas Buchli. "Acceleration-based transparency control framework for wearable robots". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 2016-Novem. IEEE, Oct. 2016, pp. 5683–5688. ISBN: 978-1-5090-3762-9. DOI: 10.1109/IROS.2016.7759836.

[7] Jacques Bobet and Robert W. Norman. "Least-squares identification of the dynamic relation between the electromyogram and joint moment". In: *Journal of Biomechanics* 23.12 (Jan. 1990), pp. 1275–1276. ISSN: 00219290. DOI: 10.1016/0021-9290(90)90386-H.

[8] Andrea Calanca, Riccardo Muradore, and Paolo Fiorini. "A Review of Algorithms for Compliant Control of Stiff and Fixed-Compliance Robots". In: *IEEE/ASME Transactions on Mechatronics* 21.2 (Apr. 2016), pp. 613–624. ISSN: 1083-4435. DOI: 10.1109/TMECH.2015.2465849.

[9] Claudio Castellini et al. "Proceedings of the first workshop on peripheral machine interfaces: Going beyond traditional surface electromyography". In: *Frontiers in Neurorobotics* 8.22 (2014), pp. 1–17. ISSN: 16625218. DOI: 10.3389/fnbot.2014.00022.

[10] P. R. Cavanagh and P. V. Komi. "Electromechanical delay in human skeletal muscle under concentric and eccentric contractions". In: *European Journal of Applied Physiology and Occupational Physiology* 42.3 (Nov. 1979), pp. 159–163. ISSN: 0301-5548. DOI: 10.1007/BF00431022.

[11] Elaine A. Corbett, Konrad P. Kording, and Eric J. Perreault. "Real-Time Evaluation of a Noninvasive Neuroprosthetic Interface for Control of Reach". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 21.4 (July 2013), pp. 674–683. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2013.2251664.

[12] Eleanor Criswell. *Cram's introduction to Surface Electromyography*. Jones and Bartlett Publishers, 2011.

[13] Brian Dellon and Yoky Matsuoka. "Prosthetics, exoskeletons, and rehabilitation [Grand challenges of robotics]". In: *IEEE Robotics and Automation Magazine* 14.1 (2007), pp. 30–34. ISSN: 10709932. DOI: 10.1109/MRA.2007.339622.

[14] Guillaume Durandau, Massimo Sartori, Magdo Bortole, Juan C. Moreno, José L. Pons, and Dario Farina. "EMG-driven models of human-machine interaction in individuals wearing the H2 exoskeleton". In: *IFAC-PapersOnLine* 49.32 (2016), pp. 200–203. ISSN: 24058963. DOI: 10.1016/j.ifacol.2016.12.214.

[15] Ekso Bionics. *EksoGT, Richmond, CA*. 2017. URL: http://www.eksobionics.com/.

[16] R. M. Enoka and J. Duchateau. "Physiology of Muscle Activation and Force Generation". In: *Surface Electromyography : Physiology, Engineering, and Applications*. Ed. by Dario Farina and Roberto Merletti. Hoboken, New Jersey: John Wiley & Sons, Inc., Apr. 2016. Chap. 1, pp. 1–29. ISBN: 9781119082934. DOI: 10.1002/9781119082934.ch01.

[17] Alberto Esquenazi, Mukul Talaty, Andrew Packel, and Michael Saulino. "The ReWalk Powered Exoskeleton to Restore Ambulatory Function to Individuals with Thoracic-Level Motor-Complete Spinal Cord Injury". In: *American Journal of Physical Medicine & Rehabilitation* 91.11 (Nov. 2012), pp. 911–921. ISSN: 0894-9115. DOI: 10.1097/PHM.0b013e318269d9a3.

[18] Dario Farina, Aleš Holobar, Roberto Merletti, and Roger M. Enoka. "Decoding the neural drive to muscles from the surface electromyogram". In: *Clinical Neurophysiology* 121.10 (Oct. 2010), pp. 1616–1623. ISSN: 13882457. DOI: 10.1016/j.clinph.2009.10.040.

[19] Anders Fougner, Oyvind Stavdahl, Peter J. Kyberd, Yves G. Losier, and Philip A. Parker. "Control of Upper Limb Prostheses: Terminology and Proportional Myoelectric Control - A Review". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 20.5 (Sept. 2012), pp. 663–677. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2012.2196711.

[20] Juan Álvaro Gallego, Jaime Ibanez, Jakob Lund Dideriksen, José Ignacio Serrano, María Dolores del Castillo, Dario Farina, and Eduardo Rocon. "A Multimodal Human-Robot Interface to Drive a Neuroprosthesis for Tremor Management". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.6 (Nov. 2012), pp. 1159–1168. ISSN: 1094-6977. DOI: 10.1109/TSMCC.2012.2200101.

[21] David A. Gabriel Gary Kamen. *Essentials of Electromyography*. Human Kinetics, 2010. ISBN: 9780736067126.

[22] R. A R C Gopura, D. S V Bandara, Kazuo Kiguchi, and G. K I Mann. "Developments in hardware systems of active upper-limb exoskeleton robots: A review". In: *Robotics and Autonomous Systems* 75 (2016), pp. 203–220. ISSN: 09218890. DOI: 10.1016/j.robot.2015.10.001.

[23] Ashraf S Gorgey. "Robotic exoskeletons: The current pros and cons". In: *World Journal of Orthopedics* 9.9 (Sept. 2018), pp. 112–119. ISSN: 2218-5836. DOI: 10.5312/wjo.v9.i9.112.

[24] J. M. Hahne, F. Biebmann, Ning Jiang, Hubertus Rehbaum, Dario Farina, F. C. Meinecke, K.-R Muller, and L. C. Parra. "Linear and Nonlinear Regression Techniques for Simultaneous and Proportional Myoelectric Control". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 22.2 (2014), pp. 269–279. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2014.2305520.

[25] Janne M. Hahne, Marko Markovic, and Dario Farina. "User adaptation in Myoelectric Man-Machine Interfaces". In: *Scientific Reports* 7.1 (Dec. 2017), p. 4437. ISSN: 2045-2322. DOI: 10.1038/s41598-017-04255-x.

[26] Tomohiro Hayashi, Hiroaki Kawamoto, and Yoshiyuki Sankai. "Control method of robot suit HAL working as operator's muscle using biological and dynamical information". In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 2. 1. IEEE, 2005, pp. 3063–3068. ISBN: 0-7803-8912-3. DOI: 10.1109/IROS.2005.1545505.

[27] Hugh Herr. "Exoskeletons and orthoses: classification, design challenges and future directions". In: *Journal of NeuroEngineering and Rehabilitation* 6.1 (2009), p. 21. ISSN: 1743-0003. DOI: 10.1186/1743-0003-6-21.

[28] Jian Huang, Weiguang Huo, Wenxia Xu, Samer Mohammed, and Yacine Amirat. "Control of Upper-Limb Power-Assist Exoskeleton Using a Human-Robot Interface Based on Motion Intention Recognition". In: *IEEE Transactions on Automation Science and Engineering* 12.4 (Oct. 2015), pp. 1257–1270. ISSN: 1545-5955. DOI: 10.1109/TASE.2015.2466634.

[29] Mark Ison, Chris Wilson Antuvan, and Panagiotis Artemiadis. "Learning efficient control of robots using myoelectric interfaces". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2014, pp. 2880–2885. ISBN: 978-1-4799-3685-4. DOI: 10.1109/ICRA.2014.6907273.

[30] Mark Ison and Panagiotis Artemiadis. "The role of muscle synergies in myoelectric control: trends and challenges for simultaneous multifunction control". In: *Journal of Neural Engineering* 11.5 (2014). ISSN: 1741-2560. DOI: 10.1088/1741-2560/11/5/051001.

[31] Ian T. Jolliffe. *Principal Component Analysis*. 2nd edition. Springer-Verlag, 2002. ISBN: 0387954422.

[32] Nikos Karavas, Arash Ajoudani, Nikos G. Tsagarakis, Jody Saglia, Antonio Bicchi, and Darwin Caldwell. "Tele-impedance based assistive control for a compliant knee exoskeleton". In: *Robotics and Autonomous Systems* 73 (2015), pp. 78–90. ISSN: 0921-8890. DOI: 10.1016/j.robot.2014.09.027.

[33] H. Kazerooni, J.-L. Racine, Lihua Huang, and Ryan Steger. "On the Control of the Berkeley Lower Extremity Exoskeleton (BLEEX)". In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. Barcelona, Spain: IEEE, 2005, pp. 4353–4360. ISBN: 0-7803-8914-X. DOI: 10.1109/ROBOT.2005.1570790.

[34] Terry K. K. Koo and Arthur F. T. Mak. "Feasibility of using EMG driven neuromusculoskeletal model for prediction of dynamic movement of the elbow". In: *Journal of Electromyography and Kinesiology* 15.1 (2005), pp. 12–26. ISSN: 10506411. DOI: 10.1016/j.jelekin.2004.06.007.

[35] Dd Lee and Hs Seung. "Algorithms for non-negative matrix factorization". In: *13th International Conference on Neural Information Processing Systems*. Denver, CO: MIT Press, 2000, pp. 535–541.

[36] Tommaso Lenzi, Stefano Marco Maria De Rossi, Nicola Vitiello, and Maria Chiara Carrozza. "Intention-Based EMG Control for Powered Exoskeletons". In: *IEEE Transactions on Biomedical Engineering* 59.8 (Aug. 2012), pp. 2180–2190. ISSN: 0018-9294. DOI: 10.1109/TBME.2012.2198821.

[37] David G. Lloyd and Thor F. Besier. "An EMG-driven musculoskeletal model to estimate muscle forces and knee joint moments in vivo". In: *Journal of Biomechanics* 36.6 (June 2003), pp. 765–776. ISSN: 00219290. DOI: 10.1016/S0021-9290(03)00010-1.

[38] Ho Shing Lo and Sheng Quan Xie. "Exoskeleton robots for upper-limb rehabilitation: State of the art and future prospects". In: *Medical Engineering & Physics* 34.3 (2012), pp. 261–268. ISSN: 13504533. DOI: 10.1016/j.medengphy.2011.10.004.

[39] Joan Lobo-Prat, Peter N. Kooren, Arvid Q.L. Keemink, Micha I. Paalman, Edsko E.G. Hekman, Peter H. Veltink, Arno H.A. Stienen, and Bart F.J.M. Koopman. "Design and control of an experimental active elbow support for adult Duchenne Muscular Dystrophy patients". In: *5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*. IEEE, Aug. 2014, pp. 187–192. ISBN: 978-1-4799-3128-6. DOI: 10.1109/BIOROB.2014.6913774.

[40] Joan Lobo-Prat, Peter N Kooren, Arno Ha Stienen, Just L Herder, Bart Fjm Koopman, and Peter H Veltink. "Non-invasive control interfaces for intention detection in active movement-assistive devices". In: *Journal of NeuroEngineering and Rehabilitation* 11.1 (2014), p. 168. ISSN: 1743-0003. DOI: 10.1186/1743-0003-11-168.

[41] Sergey Lobov, Nadia Krilova, Innokentiy Kastalskiy, Victor Kazantsev, and Valeri A. Makarov. "Latent factors limiting the performance of sEMG-interfaces". In: *Sensors (Switzerland)* 18.4 (2018), pp. 1–19. ISSN: 14248220. DOI: 10.3390/s18041122.

[42] Paweł Maciejasz, Jörg Eschweiler, Kurt Gerlach-Hahn, Arne Jansen-Troy, and Steffen Leonhardt. "A survey on robotic devices for upper limb rehabilitation." In: *Journal of neuroengineering and rehabilitation* 11.1 (2014), p. 3. ISSN: 1743-0003. DOI: 10.1186/1743-0003-11-3.

[43] Simone Marcheschi, Fabio Salsedo, Marco Fontana, and Massimo Bergamasco. "Body Extender: Whole body exoskeleton for human power augmentation". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, May 2011, pp. 611–616. ISBN: 978-1-61284-386-5. DOI: 10.1109/ICRA.2011.5980132.

[44] Nurhazimah Nazmi, Mohd Abdul Rahman, Shin-Ichiroh Yamamoto, Siti Ahmad, Hairi Zamzuri, and Saiful Mazlan. "A Review of Classification Techniques of EMG Signals during Isotonic and Isometric Contractions". In: *Sensors* 16.12 (Aug. 2016), p. 1304. ISSN: 1424-8220. DOI: 10.3390/s16081304.

[45] Robert W. Norman and Paavo V. Komi. "Electromechanical delay in skeletal muscle under normal movement conditions". In: *Acta Physiologica Scandinavica* 106.3 (July 1979), pp. 241–248. ISSN: 00016772. DOI: 10.1111/j.1748-1716.1979.tb06394.x.

[46] Domen Novak and Robert Riener. "A survey of sensor fusion methods in wearable robotics". In: *Robotics and Autonomous Systems* 73 (Nov. 2015), pp. 155–170. ISSN: 09218890. DOI: 10.1016/j.robot.2014.08.012.

[47] Max Ortiz-Catalan, Faezeh Rouhani, Rickard Branemark, and Bo Hakansson. "Offline accuracy: A potentially misleading metric in myoelectric pattern recognition for prosthetic control". In: *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS* 2015-November (2015), pp. 1140–1143. ISSN: 1557170X. DOI: 10.1109/EMBC.2015.7318567.

[48] Paul Bernhardt. *#MyoCraft: EMG in the Bluetooth Protocol.* 2015. URL: https://web.archive.org/web/20180910103025/https://developerblog.myo.com/myocraft-emg-in-the-bluetooth-protocol/.

[49] Alessandra Pedrocchi et al. "MUNDUS project: MUltimodal Neuroprosthesis for daily Upper limb Support." In: *Journal of neuroengineering and rehabilitation* 10 (2013), p. 66. ISSN: 1743-0003. DOI: 10.1186/1743-0003-10-66.

[50] Angkoon Phinyomark, Pornchai Phukpattaranont, and Chusak Limsakul. "Feature reduction and selection for EMG signal classification". In: *Expert Systems with Applications* 39.8 (June 2012), pp. 7420–7431. ISSN: 09574174. DOI: 10.1016/j.eswa.2012.01.102.

[51] Angkoon Phinyomark, Franck Quaine, Sylvie Charbonnier, Christine Serviere, Franck Tarpin-Bernard, and Yann Laurillau. "EMG feature evaluation for improving myoelectric pattern recognition robustness". In: *Expert Systems with Applications* 40.12 (Sept. 2013), pp. 4832–4840. ISSN: 09574174. DOI: 10.1016/j.eswa.2013.02.023.

[52] Tommaso Proietti, Vincent Crocher, Agnes Roby-Brami, and Nathanael Jarrasse. "Upper-limb robotic exoskeletons for neurorehabilitation: a review on control strategies". In: *IEEE Reviews in Biomedical Engineering* PP.99 (2016), pp. 1–1. ISSN: 1937-3333. DOI: 10.1109/RBME.2016.2552201.

[53] Calyampudi Radhakrishna Rao. *Linear statistical inference and its applications.* Vol. 2. Wiley New York, 1973.

[54] Robert Riener. "The Cybathlon promotes the development of assistive technology for people with physical disabilities". In: *Journal of NeuroEngineering and Rehabilitation* 13.1 (2016), p. 49. ISSN: 1743-0003. DOI: 10.1186/s12984-016-0157-2.

[55] Marieke Rohde and Marc O. Ernst. "Time, agency, and sensory feedback delays during action". In: *Current Opinion in Behavioral Sciences* 8 (Apr. 2016), pp. 193–199. ISSN: 23521546. DOI: 10.1016/j.cobeha.2016.02.029.

[56] Massimo Sartori, Dario Farina, and David G. Lloyd. "Hybrid neuromusculoskeletal modeling to best track joint moments using a balance between muscle excitations derived from electromyograms and optimization". In: *Journal of Biomechanics* 47.15 (2014), pp. 3613–3621. ISSN: 18732380. DOI: 10.1016/j.jbiomech.2014.10.009.

[57] Massimo Sartori, Leonardo Gizzi, David G. Lloyd, and Dario Farina. "A musculoskeletal model of human locomotion driven by a low dimensional set of impulsive excitation primitives". In: *Frontiers in Computational Neuroscience* 7.June (2013), pp. 1–22. ISSN: 1662-5188. DOI: 10.3389/fncom.2013.00079.

[58] Massimo Sartori, David G. Lloyd, and Dario Farina. "Neural data-driven musculoskeletal modeling for personalized neurorehabilitation technologies". In: *IEEE Transactions on Biomedical Engineering* 63.5 (2016), pp. 879–893. ISSN: 15582531. DOI: 10.1109/TBME.2016.2538296.

[59] Massimo Sartori, Monica Reggiani, Enrico Pagello, and David G Lloyd. "Modeling the Human Knee for Assistive Technologies". In: *IEEE Transactions on Biomedical Engineering* 59.9 (2012), pp. 2642–2649. DOI: 10.1109/TBME.2012.2208746.

[60] Massimo Sartori, Utku Ş Yavuz, and Dario Farina. "In Vivo Neuromechanics: Decoding Causal Motor Neuron Behavior with Resulting Musculoskeletal Function". In: *Scientific Reports* 7.1 (Dec. 2017), p. 13465. ISSN: 2045-2322. DOI: 10.1038/s41598-017-13766-6.

[61] Erik J. Scheme and Kevin B. Englehart. "Validation of a selective ensemble-based classification scheme for myoelectric control using a three-dimensional Fitts' law test". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 21.4 (2013), pp. 616–623. ISSN: 15344320. DOI: 10.1109/TNSRE.2012.2226189.

[62] Helge Toutenburg and Shalabh. *Statistical Analysis of Designed Experiments, Third Edition.* Third. Vol. 45. Springer Texts in Statistics 2. New York, NY: Springer New York, 2009. ISBN: 978-1-4419-1147-6. DOI: 10.1007/978-1-4419-1148-3.

[63] F.J. Valero-Cuevas, Heiko Hoffmann, M.U. Kurse, J.J. Kutch, and E.A. Theodorou. "Computational Models for Neuromuscular Function". In: *IEEE Reviews in Biomedical Engineering* 2.October (2009), pp. 110–135. ISSN: 1937-3333. DOI: 10.1109/RBME.2009.2034981.

[64] Allan Joshua Veale and Shane Quan Xie. "Towards compliant and wearable robotic orthoses: A review of current and emerging actuator technologies". In: *Medical Engineering and Physics* 38.4 (2016), pp. 317–325. ISSN: 18734030. DOI: 10.1016/j.medengphy.2016.01.010.

[65] Marina M.C. Vidovic, Han-Jeong Hwang, Sebastian Amsuss, Janne M. Hahne, Dario Farina, and Klaus-Robert Muller. "Improving the Robustness of Myoelectric Pattern Recognition for Upper Limb Prostheses by Covariate Shift Adaptation". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 24.9 (Sept. 2016), pp. 961–970. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2015.2492619.

[66] Kiri Wagstaff. "Machine Learning that Matters". In: *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*. Ed. by John Langford and Joelle Pineau. Edinburgh, Scotland: Omnipress, June 2012, pp. 529–536. ISBN: 978-1-4503-1285-1. arXiv: 1206.4656.

[67] WHO. *World Report on Disability - Summary*. WHO/NMH/VIP/11.01. 2011, pp. 1–23. ISBN: 9789241564182. URL: http://www.who.int/disabilities/world_report/2011/en/.

[68] Jack M Winters. "Hill-Based Muscle Models: A Systems Engineering Perspective". In: *Multiple Muscle Systems*. New York, NY: Springer New York, 1990, pp. 69–93. ISBN: 978-1-4613-9030-5. DOI: 10.1007/978-1-4613-9030-5_5.

[69] Michael T Wolf, Christopher Assad, Adrian Stoica, You Kisung, Henna Jethani, Matthew T. Vernacchia, Joshua Fromm, and Yumi Iwashita. "Decoding static and dynamic arm and hand gestures from the JPL BioSleeve". In: *2013 IEEE Aerospace Conference*. IEEE, Mar. 2013, pp. 1–9. ISBN: 978-1-4673-1813-6. DOI: 10.1109/AERO.2013.6497171.

[70] Jamie Wolff, Claire Parker, Jaimie Borisoff, W Mortenson, and Johanne Mattie. "A survey of stakeholder perspectives on exoskeleton technology". In: *Journal of NeuroEngineering and Rehabilitation* 11.1 (2014), p. 169. ISSN: 1743-0003. DOI: 10.1186/1743-0003-11-169.

[71] F.E. Zajac. "Muscle and tendon: Properties, models, scaling, and application to biomechanics and motor control". In: 17 (Feb. 1989), pp. 359–411.

[72] Ganwen Zeng and Ahmad Hemami. "An overview of robot force control". In: *Robotica* 15.5 (Sept. 1997). ISSN: 02635747. DOI: 10.1017/S026357479700057X.

[73] Shenquan Zhang, Xu Zhang, Shuai Cao, Xiaoping Gao, Xiang Chen, and Ping Zhou. "Myoelectric Pattern Recognition Based on Muscle Synergies for Simultaneous Control of Dexterous Finger Movements". In: *IEEE Transactions on Human-Machine Systems* 47.4 (Aug. 2017), pp. 576–582. ISSN: 2168-2291. DOI: 10.1109/THMS.2017.2700444.