Rafael Sanchez Souza

Design and Prototyping of a Development Platform for Exoskeleton Research

Sao Paulo
2018

Rafael Sanchez Souza

Design and Prototyping of a Development Platform for Exoskeleton Research

Master Thesis submitted to the University of
Sao Paulo in fulfilment of the requirements for
the degree of Master of Science

Sao Paulo
2018

Rafael Sanchez Souza

Design and Prototyping of a Development Platform for Exoskeleton Research

Master Thesis submitted to the University of
Sao Paulo in fulfilment of the requirements for
the degree of Master of Science

Field of Study:
Control and Mechanical Automation
Engineering (3152)

Supervised by:
Prof. Dr. Arturo Forner Cordero

Sao Paulo
2018

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catálogo-na-publicação

Souza, Rafael

Design and Prototyping of a Development Platform for Exoskeleton Research / R. Souza -- versão corr. -- São Paulo, 2018.
123 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecânica.

1.DESIGN 2.CONTROLE (TEORIA DE SISTEMAS E CONTROLE)
3.BIOMECÂNICA 4.BIOROBÓTICA I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecânica II.t.

ACKNOWLEDGEMENTS

To Renata, for the love and time that we share. To my family, for all I am is an extension of them. To Professor Arturo Forner Cordero, for his courage and heart. To my colleague Luis Felipe Rossi, for the hours he saved me with the hours he had worked before. To the community of the University of Sao Paulo, for all we have learned together. To my friends, for the support and chill.

The shadow of my elbow rests on
the shadow table as the shadow
ink flows over the shadow paper.

Arthur Stanley Eddington

RESUMO

Interface homem e máquina tem sido um campo crescente de pesquisa científica nos últimos anos. Robôs convencionais têm sido concebidos como estruturas metálicas rígidas que, quando em contato com o tecido humano, faz necessário romper com o modo de pensar corrente para atingir uma melhor interação. Exoesqueletos, chamados também de robôs vestíveis, compartilham destes desafios e abrangem aplicação na indústria, militar, medicina e entretenimento. Este trabalho introduz uma abordagem sistemática, baseada em Engenharia de Requisitos e Prototipagem, para projeto de uma plataforma de desenvolvimento para pesquisa em exoesqueletos. A dinâmica da junta humana e da junta robótica são modeladas para diferentes acoplamentos entre si. O Controle Adaptativo por Modelo de Referência é proposto como uma solução para controle de exoesqueletos; simulações indicam ser capaz de estimar os parâmetros da junta humana em tempo real. O controlador por Modelo de Referência foi implementado, tendo sucesso na modulação da impedância aparente da junta robótica. De uma perspectiva mais prática, é apresentado o projeto e construção de uma bancada experimental e o uso de um repositório online para desenvolvimento do software de controle. O repositório on-line viabiliza gestão de projetos colaborativos, focado em versionamento de software e contribuição científica. A bancada experimental foi projetada para atender as necessidades de diferentes *stakeholders* - a universidade, os pacientes e terapeutas - sendo de aplicação modular, de fácil operação e relativo baixo custo, é capaz de conduzir experimentos de controle motor e tarefas de reabilitação.

Palavras-chave: Engenharia Biomecatrônica. Exoesqueleto. Reabilitação. Controle Motor.

ABSTRACT

Human machine interface has been a growing field of scientific research for the last years. Conventional robots have been conceived as rigid metallic structures and, when metal meets human tissue, it is necessary to break the mindset in order to achieve better interaction. Exoskeletons, often called wearable robots, shares the same challenges with applications ranging from industry, to military, medicine and entertainment. This work introduces a systematic design of a development platform for exoskeleton research supported by Requirement Engineering and implemented through prototyping. The dynamics of the human joint and the robotic joint are modelled with different couplings between them. A Model Reference Adaptive Control is proposed as a solution for exoskeleton control and simulations indicate that it is capable of estimating human joint parameters in real time. The Model Reference controller is implemented, with successful modulation of the robotic joint apparent impedance. From a practical perspective, we present the design and construction of an experimental workbench and the use of an on-line repository for the control software development. The on-line repository results in a viable way for collaborative project management, software versioning and scientific contribution. The experimental workbench which was designed to meet the stakeholders needs - the university, patients and therapists - being of modular application, easy to operate and relatively low cost, can be used to conduct motor control experiments and rehabilitation tasks.

Keywords: Biomecatronics. System Design. Exoskeleton. Rehabilitation. Motor Control.

LIST OF FIGURES

Figure 1.1 – ModExo Joint	15
Figure 1.2 – Noonee	17
Figure 1.3 – FORTIS	18
Figure 1.4 – EksoVest	18
Figure 1.5 – Soft Exosuit	19
Figure 1.6 – HAL-5	20
Figure 1.8 – ExoArm	21
Figure 1.7 – ExoHand	21
Figure 1.9 – Againer	22
Figure 1.10–Exo-C Early Prototype	23
Figure 1.11–Dynamical Interface	24
Figure 2.1 – Exo-C Prototype	28
Figure 2.2 – Load cell Linear Regression	29
Figure 2.3 – Exo-OCP prototype	29
Figure 2.4 – Torsional Spring Design	30
Figure 2.5 – Exo Loli	31
Figure 2.6 – KAOS Framework	33
Figure 2.7 – Github Social Interface	34
Figure 2.8 – Github Insights	35
Figure 3.1 – Muscle-skeletal System	37
Figure 3.2 – Hill Musculoskeletal Model	37
Figure 3.3 – Simplified Coupled System	41
Figure 3.4 – Coupled Diagram With Relative Displacement	42
Figure 4.1 – Model Reference Adaptive Control	44
Figure 4.2 – Tracking performance and parameter estimation	47
Figure 4.3 – Tracking performance and parameter estimation	48
Figure 4.4 – Tracking performance and parameter estimation	49
Figure 4.5 – Tracking performance and parameter estimation	49
Figure 4.6 – Tracking performance and parameter estimation	50
Figure 4.7 – Tracking Error for Sinusoidal Input	51
Figure 5.1 – Load Cell Principles	53
Figure 5.2 – Wire Mess	53
Figure 5.3 – Wire Case	55
Figure 5.4 – Workbench	56
Figure 5.5 – Temporary Control Case	57
Figure 5.6 – System Description	58

Figure 5.7 – Control Architecture	59
Figure 5.8 – Telemetry Viewer	62
Figure 5.9 – Basic Network Setup	62
Figure 6.1 – Control Block Diagram	67
Figure 6.2 – Delay Test 1	68
Figure 6.3 – Delay Test 2	68
Figure 6.4 – Delay Test 3	69
Figure 6.5 – Load Cell Test Set-up	70
Figure 6.6 – Load Cell Test 1	71
Figure 6.7 – Load Cell Test 2	71
Figure 6.8 – Control Diagram	73
Figure 6.9 – Model Reference Tracking	74
Figure 6.10–Model Reference Tracking	74
Figure 7.1 – Final Portable Workbench	78
Figure 7.2 – Trials With Subjects Set-up	79
Figure 7.3 – Haptic Interface	80
Figure A.1 – KAOS Framework	89
Figure A.2– Goal Oriented Requirement Engineering	92
Figure A.3– User Movement Intention Goals	94
Figure A.4– Commercial Feasibility Goals	94
Figure A.5– Modularity Goals	96
Figure A.6– Safety Goals	97
Figure B.1 – ModExo Case Perspective	101
Figure B.2 – ModExo Case - Wooden Structure	101
Figure B.3 – ModExo Case - Height Requirement	102
Figure B.4 – ModExo Case - Ventilation Requirement	102
Figure B.5 – ModExo Case - USB Requirement	103
Figure B.6 – ModExo Case Front	103
Figure B.7 – ModExo Case Front CAD	103
Figure B.8 – ModExo Case Top	104
Figure B.9 – ModExo Case Top CAD	104

NOTATIONS

a	Coefficient Vector
$\hat{\mathbf{a}}$	Coefficient Vector
A	System Matrix
$\mathbf{A}_d$	Discrete System Matrix
B	Input Matrix
$\mathbf{B}_d$	Discrete Input Matrix
b_a	Arduino Reference Damping
b_d	Discrete Damping
b_e	Exoskeleton Joint Damping
b_h	Human Joint Damping
b_m	Model Reference Damping for Simulations
C	Output Matrix
C_h	Human Joint Centrifugal Velocity
C_e	Exoskeleton Joint Centrifugal Velocity
D	Feedforward Matrix
F_{ext}	External Force
G_e	Gravity Acting in the Exoskeleton Joint
G_h	Gravity Acting in the Human Joint
J_d	Arduino Model Reference Inertia
J_d	Discrete Inertia
J_e	Exoskeleton Joint Inertia
J_h	Human Joint Inertia
J_m	Reference Model Inertia for Simulations
k_a	Arduino Reference Stiffness
k_d	Discrete Stiffness
k_e	Exoskeleton Joint Stiffness
k_h	Human Joint Stiffness
k_m	Reference Model Stiffness for Simulations
M_e	Exoskeleton Joint Mass
M_h	Human Joint Mass
x	States Vector
u	Inputs Vector
t	time
α	Model Reference Coefficient Vector
β	Tracking Error Parameters

continue...

τ	Generic Torque
τ_B	Rigid Body Torque
τ_c	Contact Torque Between Human and Exoskeleton
τ_e	Exoskeleton Torque
τ_h	Human Torque
θ	Generic Position
θ_e	Exoskeleton Joint Position
θ_h	Human Joint Position

CONTENTS

1	INTRODUCTION	15
1.1	State of the Art	16
1.1.1	Industrial	16
1.1.2	Military	17
1.1.3	Medical	19
1.1.4	Others	20
1.2	Exoskeleton Control	22
1.2.1	Impedance Control	23
1.3	Project Goals	25
2	METHODOLOGY	26
2.1	Prototyping	27
2.1.1	Exo-C Prototype	27
2.1.2	Exo-OCP Prototype	28
2.1.3	Exo-Loli Prototype	30
2.2	Goal Oriented Requirement Engineering	31
2.2.1	KAOS	32
2.3	GitHub	32
2.3.1	Features	33
3	MODELLING	36
3.1	Human Joint Mechanics	36
3.1.1	Linearised Dynamics of Human Joint	36
3.2	Linearised Dynamics of Exoskeleton Joint	38
3.2.1	State Space Representation	39
3.2.2	System Discretization	40
3.3	Coupled Dynamics	41
3.3.1	Simplified Coupled Dynamics	41
3.3.2	Coupled system dynamics with relative displacement	41
4	ADAPTIVE CONTROL	44
4.1	Model-Reference Adaptive Control	44
4.2	Adaptive Control of a Linear System With Full State Feedback	45
4.3	Simulations and Results	46
4.4	Assistive Control	47
4.5	Parameter Uncertainty and Estimation	48

4.6	Controller Robustness	48
4.7	Discussion	48
5	DEVELOPMENT PLATFORM	52
5.1	Mechanical Hardware	52
5.1.1	Wire Protection	53
5.1.2	ModExo Portable Case	54
5.2	Control Hardware	56
5.2.1	High Level Layer	56
5.2.2	Low Level Layer	58
5.3	Software	60
5.3.1	EPOS Studio	61
5.3.2	Telemetry Viewer	61
5.3.3	CAN bus	61
5.3.4	Main Arduino Code	64
6	PERFORMANCE ANALYSIS	67
6.1	Delay Tests	67
6.1.1	Results	69
6.1.2	Discussion	69
6.2	Load Cell Calibration	70
6.2.1	Software Calibration	70
6.2.2	Static load cell tests	71
6.2.3	Results	71
6.2.4	Discussion	72
6.3	Model Reference Dynamics Control	72
6.3.1	Model Reference Tracking	73
6.3.2	System Identification	75
6.3.3	Discussion	75
7	CONCLUSION: FUNCTIONAL AND NON-FUNCTIONAL ANALYSIS	77
	REFERENCES	81
	APPENDIX A – MODULAR EXOSKELETON DESIGN: REQUIREMENT ENGINEERING WITH KAOS*	86
A.1	Introduction	88
A.2	Methodology	90
A.2.1	Goal Oriented Requirement Engineering	90
A.2.2	KAOS	91

A.3	System Description	92
A.4	Requirement Modelling	93
A.4.1	Identifying Preliminary Goals and Refinement	93
A.4.2	Formalizing Goals and Identifying Objects	93
A.4.3	Eliciting Further Through WHY and HOW Questions	95
A.4.4	Identifying potential responsibility assignments	96
A.5	Discussion	97
A.6	Conclusion	99
	APPENDIX B – MODEXO PORTABLE CASE DETAILED DESIGN	100
	APPENDIX C – CONTROL HARDWARE SCHEMATICS	105
	ANNEX A – EPOS2 DRIVER CATALOG	109
	ANNEX B – MOTOR EC90 FLAT DRIVER CATALOG	111
	ANNEX C – ENCODER MILE CATALOG	113
	ANNEX D – MA3 ENCODER CATALOG	115

1 INTRODUCTION

Biomechatronics is the study of life through the eyes of Mechatronics Design. Under this perspective, the human body is a system composed of subsystems that interact to produce results not obtainable by the subsystems alone (NASA, 2007). The human eyes are compared to robot sensors which acquire visual information for the system. Similarly, the ears and the skin provide other information about the environment such as temperature, sounds, etc. All this information is converted into electrical signals and processed to generate adequate responses from the actuation subsystem resulting in human body displacement or object manipulation.

An exoskeleton may be defined as an outer rigid structure which mechanically shares the load required for the user to move. It is a robot which closely interacts with the human being, therefore, a multidisciplinary project: it requires a systematic understanding of the human body mechanics and gathers the state of the art technical and scientific challenges in robotics. As seen repeatedly along the engineering history, a bio-inspired approach is strongly present: Nature has, for millions of years, experimented, tested, validated and discarded solutions for life. Our ultimate goal is to come up with a better design, to do so, we observe, learn and repeat.

Figure 1.1 – ModExo Joint intermediary set-up: the control architecture are final however, as shown in the next chapters, the hardware, which encompass electronics and mechanics, still has features to be developed.

Figure 1.1 shows an intermediary set-up for the modular exoskeleton joint developed in this work. This prototype derives from previous research done at the Biomechatronics Laboratory - University of Sao Paulo - where mechanical prototypes were developed but with simulations of impedance control. For instance, an upper limb exoskeleton prototype

was developed in (MIRANDA, 2014). The mechanical design has a particular feature: the segments can be divided into very similar modules formed by a joint and a link. This project presents a modular design approach, which has not been sufficiently explored in exoskeleton research (MACIEJASZ et al., 2014). More than bringing development and maintenance costs down due to standardization of components (MATSUMARU, 1995), modularity has the potential to transcend a specific application: the design aims at broader solutions. Considering the exoskeleton development in its environment, the Biomechatronics Laboratory of the University of Sao Paulo, this project is a development platform from which many students can benefit.

This work is divided in seven chapters, which can be read independently. They are linearly organized as a sequence because that is the conceived structure, however, the knowledge inside those chapters is not a sequence, but a mesh. The Introduction contains an overview of the applications of exoskeleton as a commercial product, the major challenges is exoskeleton research and the goals of this work. The Methodology contains a brief description of the two major methods used for the project development and documentation: prototyping and requirements engineering. In the Modelling chapter, the mechanics of the human joint and the exoskeleton are described mathematically to be later used in simulations and control design. The chapter Adaptive Control analyses the viability of applying the Adaptive Control strategy for assistive exoskeletons. Finally, the chapter Development Platform presents the prototype achieved by this work and is followed by a Performance Analysis in the next chapter.

1.1 State of the Art

What is it possible to do with exoskeletons? Are there commercially available prototypes? Which ones? Aiming at answering some of these questions, this section presents some of the latest prototypes.

The three main categories - Industrial, Military, Medical - and their respective application are described below. It is important to mention, however, that the overall working principle for powered exoskeletons is very similar: a criteria for activating the actuators must be defined. Normally, that actuation will be related to the user movement intention. For that reason, many exoskeletons, such as HAL and ExoHand may be used for more than one application with little or no modification.

1.1.1 Industrial

Focused on injury prevention and individual productivity boost, exoskeletons are used to reduce fatigue as with tool holding and to provide comfort as seen in chairless chairs 1.2. The industrial exoskeletons represent a growing market: most of the companies are

Figure 1.2 – Chairless Chair, passive, developed by Noonee. (Source: againer-ski.com/)

focused on medical application while the benefits with the use of this technology may lead to increase in productivity and work quality.

A classical challenge in exoskeleton research is reducing the weight of the prototypes. Since the weight is mainly due to the motors and energy storage systems - such as DC motor and batteries, the power-to-mass ratio is a critical technological limitation. FORTIS, in Figure 1.3, from Lockheed Martin, has a very simple solution for that: a passive exoskeleton, which acts as a structure, transfers the load directly from the end effector to the ground. This product greatly reduces the effort by the user, which only has to orientate the tool. Passive exoskeletons also greatly reduce the product final cost.

In Figure 1.4, EksoVest, from EksoBionics, is a powered upper body exoskeleton which elevates the wearer's arm, assisting in tasks ranging from chest height to overhead. The product weighs 4.3kg, which is lighter than most of the actuated exoskeletons.

1.1.2 Military

Military exoskeletons have similar applications as the Industrial exoskeletons: to reduce the wearer's workload and help them carry equipment around the field. In addition to the challenge of designing exoskeletons for controlled environments such as hospitals, there is a greater one of designing them for hostile scenarios with unpredictable variables, yet far from being accomplished. Other applications include energy harvesting devices which have been developed with the purpose to convert kinetic energy into electric energy as a charging device.

Figure 1.3 – FORTIS Exoskeleton, passive, developed by Lockheed Martin. (Source: www.lockheedmartin.com/)

Figure 1.4 – EksoVest, active, developed by Eksobionics. (Source: eksobionics.com/)

Figure 1.5 – Soft ExoSuit, active, developed in Harvard University, is not rigid as conventional exoskeletons. (Source: news.harvard.edu)

Most of the exoskeletons found rely on a rigid structure which aims at reducing the workload for the wearer. These projects normally result in many design challenges, from the interface of a rigid structure with a soft surface - such as the human body skin - to the reproduction of the degrees of freedom involved in human movement: an upper limb exoskeleton may have up to 7 degrees of freedom (PERRY; ROSEN; BURNS, 2007) and 12 degrees of freedom for the lower limb (APKARIAN; NAUMANN; CAIRNS, 1989). For that reason, soft robotics is being applied to wearable robots. The Soft ExoSuit, shown in Figure 1.5 is a flexible exoskeleton developed by Biodesign Lab at Harvard. This exoskeleton is composed by a combination of rigid and flexible materials, such as tissues, which attach the exoskeleton to the human body, and steel cables connected to the actuators, located on the back of the user. This solution also transfers the motors, from the human limbs to the back, which greatly reduces the additional inertia caused by the motors.

1.1.3 Medical

Medical exoskeletons represent the majority of the commercial exoskeletons available (EXOSKELETON...,). The most relevant are related either to help in the rehabilitation of the patient or to provide functional assistance to a person with mobility issues due to a disease or accident. Exoskeletons have been widely applied for rehabilitation as a possibility of reducing the hours needed of a health professional. The exoskeleton offers an adaptive interface, which may assist the subject in rehabilitation with different features, such as progressive assistance, regressive assistance, amongst others. In this case, the assistance is temporary, meaning that the expectancy is that the the patient will get better and the exoskeleton will no longer be needed. The exoskeleton can also be used in a functional assistance, for example, as a replacement for a wheelchair, in that case, the

device is used when it is needed.

As a classical exoskeleton example, designed to assist people with mobility issues, there is HAL for Medical Use (Hybrid Assistive Limb), from Cyberdyne Inc. The movement detection is made by a combination of informations, which includes BES (bio-electric signals) and system information such as joints position. The CPU then commands the actuators on assisting the patient movement.

Figure 1.6 – HAL-5, active, developed by Cyberdyne, Tsukuba, Japão. (Source: www.robaid.com/)

Since the release of HAL-5 in 2005, in terms of functionality, HAL represents much of what an exoskeleton project aims at achieving, it has even been approved for medical trials in Europe (TÜV...,). HAL can be considered a technological platform with many different applications, with medical being one of them. However, the price of such a device can reach up to \$130.000,00, which makes this technology out of reach for the general public yet.

1.1.4 Others

The ExoHand 1.7 is an apparatus developed by Festo. By wearing the ExoHand, the user is able to remotely manipulate objects with a robotic device that mimics the movements performed by the user. One interesting aspect of this exoskeleton is that it provides force feedback for the user, meaning that whenever the robotic arm catches an object, the user will feel the resistance that object offers. The exoskeleton uses pneumatic actuators and implements compliant control, registering positional and force parameters. This is an experimental project which is not for sale. The applications are varied, the robot can learn a repetitive task, manipulate objects in hazardous environments or assist patients who have suffered from stroke.

Figure 1.8 – ExoArm, active, opensource project. (Source: hackaday.io/)

Figure 1.7 – ExoHand, active, developed by Festo. (Source: www.festo.com/)

Although most of the products mentioned above are cutting edge technology, ExoArm, shown in figure 1.8 is a project whose purpose is, not only to develop a \$100,00 exoskeleton, but share project results and the building know-how with the open source community. The motor used is a windshield wiper motor, which has quite a lot of power for a really low cost.

Again, 1.9 is a product which managed to gather the positive aspects of exoskeleton research and managed to make it economically feasible. This exoskeleton is passive, and offers assistance for those willing to ski, but without so much strength in the legs. It's a very similar concept to the chairless chair.

Figure 1.9 – Againer, passive, developed for skiing. (Source: againer-ski.com/)

1.2 Exoskeleton Control

One of the major challenges in exoskeleton design is regarding the intervention is user movement, for the controller has to guarantee an adequate man-machine interface.

The type of control performed by the exoskeleton depends on the application and the technical resources available. A simple exoskeleton project, for example, would be of a human arm attached to a motor by two links as seen in the Figure 1.10. In that case, a trivial solution would be to have a switch turn the motor on and off in order to move the arm. Evidently, that solution would be, apart from dangerous, very limited in terms of application: it would be very difficult to manipulate objects with any precision, it would require one extra hand to turn the switch on and off, amongst other problems.

Making such an intervention is a simpler task than to have control over that intervention.

We may divide this challenge between control input and control output. The control input is related to the condition that actuates the exoskeleton joint. This challenge can also be translated as “detecting the user movement intention”. In the example above the input would

Figure 1.10 – Elbow Exoskeleton early prototype, active, developed at the Laboratory of Biomechatronics. (Source: Biomechatronics Laboratory)

be a simple on-off switch, the user movement intention would have to be indicated by an operator, or by the user itself, as is case case of the ExoArm. There are, however, other ways do detect user movement intention, such as:

EMG Stands for Electromyography and is capable of recording electrical activity produced by the muscles. Several researches try to find a linear relation between the myoelectric activity and the amount of effort performed by the user ([HOGAN, 1984a](#)). Although this technology is very popular in exoskeleton research, there are only few commercial applications, examples are HAL and MyoPro.

EEG Stands for Electroencephalography and is capable of reading electrical activity form the brain. EEG is a very difficult signal to adequately read, which makes it still a experimental technology. Some researches have managed to control robots via brain signals but with intracranial applications. Although there are commercial products which are able to interpret EEG data, such as EMOTIV, to the best of our knowledge, there are no commercial applications of EEG for exoskeleton control yet.

Mechanical Sensing Information such as motor current, joint position or interaction force can be interpreted by the system to estimate user movement intention.

On the output side, the controller must provide an intervention in the user movement, in such a way that the experience is smooth and safe.

1.2.1 Impedance Control

Impedance Control has been formalized by Hogan in ([HOGAN, 1984b](#)) and well defined as “An approach to manipulation”. This strategy fuses the above mentioned input and output and provides a solution for manipulators interfacing with unknown environments.

Impedance is a generic term from Electromagnetism used to describe electrical resistance. More specifically, it refers to the amount of electrical current associated with a difference

Figure 1.11 – If the link that couples the two systems human and exoskeleton is rigid, than any movement from any of the systems will represent a severe interference in the other. On the other hand, if the systems are attached by a damper and/or spring, than a more compliant interface is possible. (Source: (NAGARAJAN; AGUIRRE-OLLINGER; GOSWAMI, 2015))

of electric potential. In (HOGAN, 1984b), the a mechanical analogy is proposed where impedance is a relation of effort (e.g., voltage, force) - and flow (e.g., current, motion). In either uses, impedance is always referring to a relation between two systems and how they exchange energy. For example, the touch of a fruit that is green will feel different from one that has already ripen; its impedance will be different.

In biomechanics, the term impedance is also studied as a state of the musculoskeletal system as consequence of motor control. In (TEE et al., 2004), the mechanical impedance of the human arm is related to position, force and instability. i.e., the natural reaction to perturbations applied to the hand is an increase in its impedance. In (HOGAN, 1984a), the mechanism of modulation of human arm impedance and its importance for motor control is investigated.

In both cases, impedance has been addressed as a useful property for adaptation. That is very convenient, since we want both systems, user and exoskeleton, to adapt to each other.

In a practical way, the use impedance control has a similar effect to what is shown in picture 1.11: however, the dynamical behaviour of the interface is virtually defined the controller, accordingly to the following equation, where k , b and J are, respectively, the parameters stiffness, damping and inertia:

$$\tau = J\ddot{\theta} + b\dot{\theta} + k\theta \quad (1.1)$$

Impedance systems receives position (θ) and outputs torque (τ), admittance systems receives torque and outputs position (HOGAN, 1984b).

1.3 Project Goals

To design and implement an assistive impedance controller that adapt to different tasks and which does not require a precise knowledge of the system-to-be-controlled parameters.

To design a controller which doesn't require sensor setup prior to use. This excludes the use of EMG or EEG, since they need to be calibrated and placed over the subject's body. This controller should react directly to mechanical interaction with the user.

To develop a methodology of systematic exoskeleton design including definition of requirements, exploration of possible solutions, documentation of the work and procedures for a development platform for exoskeleton research, which is not restricted to a specific joint of a specific human limb.

2 METHODOLOGY

This project development is divided into technology development and knowledge management. The former is concerned with the implementation of an experimental set-up and impedance control. The latter, to gather the knowledge that resulted from the previous Labs projects, to combine it with the knowledge that resulted from this project and present them on an accessible way for the projects to come.

The approach presented has as basis some of System Design key activities: Requirement Engineering, Design Solution Definition, Prototyping, Documentation and Functional and Performance Analysis. System Design is a methodical, disciplined approach (NASA, 2007) to design any device to reach the stakeholder expectations in order to obtain:

- High-quality requirements elicitation
- Requirement modelling and analysis with traceability
- Solving conflicts and ambiguous requirements
- Decision process (for solutions) and Rationale documentation
- Design validation and verification

“A system is a construction or collection of different elements that together produce results not obtainable by the elements alone.” (NASA, 2007)

This definition of systems has a lot in common with the mechatronics approach to problems: the system is divided into subsystems, each subsystem is understood as a black box with inputs and outputs. When it is not possible to reach the desired solution only with the an overview approach, that box is opened to manipulate the outputs.

The final goal of this project is to have a working rehabilitation exoskeleton; however, there are infinite different solutions that might accomplish that goal. Requirement Engineering is a process that helps to narrow those possibilities by better defining the purpose (NUSEIBEH; EASTERBROOK, 2000) of the designed system and, hopefully, achieve a more suitable solution. In the Section 2.2, the process of applying the framework KAOS for Goal Oriented Requirement Engineering will be described.

Fast prototyping is a strategy that consists of minimal viable solutions implementation, analysing them and making decisions. In the Section 2.1, the prototypes that were used in this work and the learning from each of them are described. This is used in combination with

Requirement Engineering: when not all requirements are defined, prototyping becomes a complementary way to elicit requirements and learn during the process.

As documentation and prototyping tool, GitHub, a web platform for collaborative project development is used and described in Section 2.3. System Design, in general and the tools derived from it, were born in software development context, but can be abstracted to information-knowledge management and contribute with hardware development.

2.1 Prototyping

“Prototypes support the concept of progressive elaboration in iterative cycles of mock-up creation, user experimentation, feedback generation, and prototype revision.” (GUIDE, 2001)

Having set the goal to achieve an assistive behaviour with the exoskeleton, it is necessary to divide that goal into subgoals that can be achieved faster. The first defined milestone for this project was: to achieve transparent behaviour or, to follow the user movement with the minimal intervention. The first tests were conducted with the Exo-C, a one degree of freedom exoskeleton.

2.1.1 Exo-C Prototype

Presented in (MIRANDA et al., 2012), the EXO-C, shown in figure 2.1 is an exoskeleton with one actuated degree of freedom at the elbow joint. Actuation is provided by Maxon Motor EC32, 80[W] with a 18:1 planetary reduction acting on a ball-screw axis and driven by a Maxon EPOS2 24/5 driver. The interaction force between the arm and the exoskeleton is obtained by means of a Wheatstone bridge of 350[Ω] strain gauges glued to the exoskeleton aluminium structure attached to the forearm. The position is measured by a rotational potentiometer 2[kΩ] with 2 percent linearity coupled to the elbow joint. There are two levels of control, a lower level, provided by the driver EPOS2 24/5 from Maxon and a higher level, implemented in an embedded computer PC104. The exoskeleton can be anchored to the arm with a standard commercial orthosis, such as the shoulder orthosis Ottobock Omo-Neurexa (Otto Bock gmbH, Germany).

With this prototype, the first tests of the power system provided by Maxon were made with a controller which tried to transparently follow the human movement. The communication system was analogue: position was read by the potentiometer and contact torque was read by strain gauge. Both signals were sent to the PC104 which, running Linux Real Time, would calculate the target position according to that data and send it to the EPOS2 driver.

The most important outcomes of this prototyping step were:

- The set up was used to validate the exoskeleton use in motor control tests (MOURA et

Figure 2.1 – Exo-C prototype

al., 2016)

- The EPOS2 Driver is underused with analogue communication: it has many data from the motor, such as position, velocity and current that are only available through digital communication.
- The Exo-C prototype mechanical design result in clearances which are difficult to remove and that greatly spoil the controller performance.
- The PC104 is a powerful but complex embedded hardware for this stage of the project: the computer which compiled the software was not the same that ran it so, an extra step was necessary for testing. Also, the acquisition board library had a steep learning curve.
- The load cell with strain gauges and full Wheatstone bridge was validated, showing linear behaviour as seen in Figure 2.2.

2.1.2 Exo-OCP Prototype

In (MIRANDA, 2014)], a mechanical structure of an exoskeleton was built, including motor and reduction. This prototype can be divided into modules, which greatly influenced this systematic modular approach.

Figure 2.2 – Linear Regression for the load cell signal with different weights

Figure 2.3 – Exo-OCP prototype

The Exo-OCP actuation is provided by a Maxon Motor EC90, 90W with a 100:1 harmonic reduction and by the EPOS2 24/5 driver. The interaction force between the arm and the exoskeleton is obtained by measuring the deflection of a serial torsion spring. The deflection is provided by the arm position, measured by an absolute encoder and the motor displacement, measured by the Maxon MILE encoder.

This prototype was the basis for the Mod-Exo actual system, however, it also presented mechanical issues which made testing with it impracticable. Also, this prototype did not have a load cell yet. Some investigation was made regarding the most promising solutions for torque sensing but were not in the scope of this project. In order to continue with this prototype development, the following steps should be taken:

- A torsional spring load cell must be built.
- The torsional springs deformation, when under compression, results in an axial displacement in the actuator. Together with Matheus Guinezi, a scientific initiation student, a new design based on (SANTOS; CAURIN; SIQUEIRA, 2013) was made

Figure 2.4 – Torsional Series Spring Prototyping: to the left, the design from (MIRANDA, 2014). To the right, the design adapted from (SANTOS; CAURIN; SIQUEIRA, 2013). In the later, the displacement in the Z axis, when torque is applied is reduced.

that would prevent such displacement. The simulation results can be seen in Figure 2.4. However, up to this moment, these springs have not been produced.

- The main issue is regarding the *Harmonic Drive*: when under deceleration, an axial force pushes the Wave Generator out of the Flexspline cup, leading to the reduction decoupling. It is necessary to design a mechanical element that keeps both parts together.
- For the acquisition of the angular position of the exoskeleton, a gear transmission between the joint and digital encoder was designed. However, this transmission has clearances that spoil the observer accuracy.

2.1.3 Exo-Loli Prototype

Presented in (SOUIT et al., 2016), this lower-limb exoskeleton prototype has a lot in common with the Exo-OCP: Maxon Motor EC90, EPOS2 24/5 driver and 100:1 harmonic reduction. The contact torque, however, is provided by the same solution as the Exo-C, a load cell. Also, the mechanical project included a solution for maintaining the harmonic reduction coupled. Being so, this last prototype is the one further described in the following

Figure 2.5 – ExoLoLi with different joint configurations: to the left shows the use of a passive knee joint, in the centre, the assembly designed for full assistance, with knees and ankle joints actuated, to the right, only the knee is actuated. Adapted from (SOUIT, 2016)

chapters with which the admittance control was implemented. The ExoLoLi prototype was also developed with a modularity approach as shown in Figure 2.5.

2.2 Goal Oriented Requirement Engineering

Systems have functional and non-functional requirements (LAMSWEEERDE, 2001). Functional requirements are normally related to WHAT the system is supposed to do, stated clear and objectively. In the case of our project: "The exoskeleton must establish a dynamic interface with the user, where the parameters inertia, damping and stiffness may be defined by an operator."

On the other hand, the non-functional requirements, are normally related to the quality of the service or HOW the system will perform. Some examples of non-functional requirements are: "The exoskeleton must be comfortable", "The exoskeleton must be affordable", "The exoskeleton must not affect negatively the user self esteem".

To understand the physiological and psychological user requirements is not an easy task. Although this is already a subject of study (WOLFF et al., 2014), when it comes to making project decisions, these needs are not so often considered.

Dividing requirements in Functional and non-Functional, however, may lead to a less intuitive process. Goal Oriented Requirement Engineering (GORE) does not bring the need

to distinguish between those, for goals encompass all which is necessary to achieve both conditions. With this approach, the design process departs from high level requirements - goals - formulated in the abstract diagrammatic language KAOS and map them to a low level requirements - as technical as possible - suited to the scope of the project.

A paper with this approach was published in (SOUZA et al., 2016) and can be accessed in the Appendix A for further information.

2.2.1 KAOS

KAOS stands for "Keep All Objectives Satisfied" (LAMSWEEERDE; LETIER, 2004) or "Knowledge Acquisition in autOMated Specification" (DARDENNE; LAMSWEEERDE; FICKAS, 1993). KAOS provides a framework based on visual diagrams that helps to implement GORE. The KAOS is composed by four diagrams that relate to each other to obtain a meta-model shown in Figure 2.6. The diagrams are described below as in (SOUZA et al., 2016):

Goals Diagram The fundamental model for KAOS in which the project goals are presented. Goals are linked to subgoals through refinements. The Goal "Avoid Tissue Damage By Excessive Range Of Motion" may be refined to the Requirement "Maintain Safe Range Of Motion]" which is constrained by the Domain Property "Human Joint Safe Range of Motion" and again refined into the Requirement "Maintain Accurate Joint Angle Measure".

Object Model Allows the identification of objects, such as entity, relationship, event or agent. Defines the concepts of the application domain and the system constraints. It also allows for establishing the object characteristics as attributes. The Exoskeleton Joint and the Human Joint relation are defined as "Attached".

Agent Responsibility Model Defines the responsibilities between agents and requirements. The Controller and the Position Sensor are responsible for maintaining a safe range of motion and an accurate joint angle measure.

Operation Model Shows how the objects work together to achieve the system requirements by defining the state transitions. If angles are off limits, the Controller will bring the Exoskeleton Joint back to a safe position.

2.3 GitHub

Git is a version control system developed as a tool for Linux developing. Git works with repositories, it helps to manage all changes in documents by giving has access to the versions history.

Figure 2.6 – The KAOS framework adapted from (RESPECT, 2007). Source: (SOUZA et al., 2016)

GitHub is a web-based platform, mostly used for hosting source codes, which offers amongst its own features, distributed version control and source code management functionality of Git. In 2017, the platform has more than 3 million users and hosts more than 60 million projects (GITHUB,). Some of the most well known hosted projects are: WordPress, GNU/Linux, Atom and Electron.

2.3.1 Features

In addition to allowing projects to be manipulated using the standard Git command-line interface, GitHub offers a web interface which simplifies Git functionality for new users. This interface also provides social networking-like functions and is shown in figure 2.7.

Versioning

Not only is it useful for project documentation and collaborative work, versioning is very useful for software developing-prototyping. *Committing* is the key activity to versioning. Every time a significant change is made in the code, apart from saving that locally (in the computer), a *commit* will save that change in the repository. Every *commit* is associated with a *commit message*, a description of the changes made in this new version.

There are two direct consequences of that: first, everybody - including the developer - is able to see what changes were made, when they were made and why. The second is, since there is a clear documentation of what is being done, it is very easy to recover previous working versions of the code which saves a lot of time when programming.

Figure 2.7 – GitHub web interface showing the active public repositories from biopmr user. It is possible to follow that user and to see it's contribution activity.

Dropbox and Google Drive, two of the most popular file hosting services have recently included the Versioning feature into their applications. However, the versioning has a time range limit and also is not associated with *committing*, which makes it more user friendly - for the simplicity - but lacks documentation.

Issues

Issues are a systematic way to track tasks and enhancements and share with the project collaborators. Issues can be shared and commented, can be used to set Milestones and accepts labels. It is also possible to mention collaborators.

Documentation with Markdown

As mentioned before, GitHub is focused on software documentation rather than hardware. However, with the features mentioned above (and others), the platform provided by GitHub offers great possibilities for any project documentation.

There is another feature, however, that greatly improves that documentation and distinguishes using GitHub from any other file hosting service: the renderization of the README.MD file. It is very common to document files with a readme.txt file. The readme.txt file contains important information regarding the content of that folder and subfolders. GitHub will automatically render that file when written with the extension .md - standing for Markdown.

Figure 2.8 – Graph with the repository activity, shows that the peak of programming activity was between June and August

Markdown is a lightweight syntax, with a structure very similar to $\text{\LaTeX}$ that can be used, not only in readme.md files, but in other places around GitHub, such as Issues.

Insights

This feature provides a visual representation of the repository activity by tracking the commits meta data, such as when it was made and who made it. This feature is very useful for project management since it provides an estimative of the work being done.

3 MODELLING

The muscle dynamics, considering the non-linearly varying parameters, show a highly non-linear behaviour. Such behaviour is subject to human desire, which is, mathematically speaking, unpredictable. Given such complexity, it is very often impractical to design a model-based controller that fits a large range of human motions and tasks.

The system model delivers a mathematical description of the observed problem and its behaviour. In the case of this study, the angular flow - or motion - associated to a torque input at the interface between exoskeleton and human joint. This chapter shows different modelling possibilities, which aim at providing a reliable representation of the motion of human and robotic joint while keeping the mathematics simple.

In the following sections, a brief review of the non-linear dynamics of the human joint is presented, a linearised model of that dynamics, an analogue model for the exoskeleton, a simplified coupled dynamics and, finally, the coupled system dynamics with relative displacement.

3.1 Human Joint Mechanics

Figure 3.1 shows a simple representation of the human elbow actuation: the muscle is attached to the skeleton in such a way that, whenever the muscle contracts, this contraction produces a movement that results in a torque in the elbow. The presence of an antagonist muscle - that is missing in the diagram - allows a better control of the dynamics of the movement.

A classical mathematical representation of the muscle mechanical response, introduced in (HILL, 1938), is given by the Hill Musculoskeletal Model (MSM) shown in Figure 3.2. In this model, the muscle is described as a Contractile Element (CE) and two non-linear springs, the Series Elastic Element (SE) and the Parallel Elastic Element (PE) in reference with the contractile element. The contractile element presents a viscous behaviour.

3.1.1 Linearised Dynamics of Human Joint

The study of the dynamics of muscle activation has been addressed for a long time, and still, there is not a consensus regarding the best modelling. In this work, two models are considered: the Hill Musculoskeletal Model (MSM) and the Spring-Damper-Inertia Model (KBI) which are widely implemented in literature (KISTEMAKER; SOEST; BOBBERT, 2007).

Although accepted to be more adequate when describing human limb dynamics, the

Figure 3.1 – This muscle-skeletal system diagram shows a typical cause of non-linearity: the torque provided by the contraction of the muscle changes according to the elbow's opening angle. Source: (SHIER; BUTLER; LEWIS, 1996)

Figure 3.2 – This descriptive model conceives the muscle as containing a contractile element and two spring elements. Source: (KISTEMAKER; SOEST; BOBBERT, 2007)

MSM implementation in adaptive control is not straightforward. For the scope of this study, the KBI Model will be used to test and validate the controller, and can be later compared with MSM. Bellow, some simplifying hypothesis are assumed for the model linearization:

- Velocity, position and acceleration are independent and linear variables;
- The tendon is very stiff: as consequence, the Series Elastic Element (SE) is no longer considered in the model.
- Muscle is assumed as having a behaviour similar to passive springs or dampers, therefore, there is a strong correlation between muscle length and strength applied to it;
- Coriolis effects not significant compared to viscoelastic proprieties of the muscle;
- Movement is made parallel to the gravity;

A simplified but satisfactory representation of a one degree of freedom human joint dynamics (KEARNEY; HUNTER, 1989) is that of a rigid link rotating around the joint axis given by the following second order equation:

$$\tau_h(t) = J_h \ddot{\theta}_h(t) + b_h \dot{\theta}_h(t) + k_h \theta_h(t) \quad (3.1)$$

where $\tau_h(t)$ is the torque applied to the human joint by the contraction of the muscle and $\theta_h(t)$ is the resulting displacement from that torque. J_h , b_h and k_h represent the human joint parameters inertia, damping and stiffness, respectively. This model has been adopted in many studies with successful implementation of exoskeleton controllers (OLAYA, 2008; AGUIRRE-OLLINGER et al., 2007).

3.2 Linearised Dynamics of Exoskeleton Joint

The exoskeleton may also modelled as a rigid link, represented as a typical manipulator:

$$\tau_e = M_e(q) \ddot{q} + C_e(q, \dot{q}) \dot{q} + G_e(q) \quad (3.2)$$

where τ_e is the motor control torque to perform movement, $M_e(q)$ being the position dependent robotic joint mass matrix, $C_e(q, \dot{q})$ the Coriolis and centrifugal velocity terms and $G_e(q)$ the gravity term.

Therefore, it can be represented analogously to the human joint as:

$$\tau_e(t) = J_e \ddot{\theta}_e(t) + b_e \dot{\theta}_e(t) + k_e \theta_e(t) \quad (3.3)$$

3.2.1 State Space Representation

Because of modularity, the state spaces is a good representation of the system for its facility of handling Multiple Input Multiple Output (MIMO) systems. Also, state spaces representation works very well with MATLAB (The Mathworks, Inc., USA) for simulations. A time invariant system is described by the following system:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

Where $\mathbf{A}$ is the matrix which describes the plant dynamics and $\mathbf{B}$, the actuator dynamics. As already seen, the relation between contact torque and exoskeleton joint position is given by the second order equation:

$$\tau_e(t) = J_e \ddot{\theta}_e(t) + b_e \dot{\theta}_e(t) + k_e \theta_e(t)$$

or, in the differential form:

$$\ddot{\theta}_e(t) = \frac{1}{J_e}(\tau(t) - b_e \dot{\theta}_e(t) - k_e \theta_e(t)) \quad (3.4)$$

If we define:

$$x_1(t) = \theta_e(t)$$

$$x_2(t) = \dot{\theta}_e(t)$$

Then:

$$\dot{x}_1(t) = x_2$$

$$\dot{x}_2(t) = \frac{1}{J_e}(\tau(t) - b_e x_2(t) - k_e x_1(t))$$

This system can be then represented in the state spaces form:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k_e}{J_e} & -\frac{b_e}{J_e} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{J_e} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad (3.5)$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (3.6)$$

3.2.2 System Discretization

In order to discretize the parameters from continuous time to state spaces, consider the continuous-time state space equation:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$$

The discrete time representation takes the following form:

$$\mathbf{x}((k+1)T) = \mathbf{A}_d(T)\mathbf{x}(kT) + \mathbf{B}_d(T)\mathbf{u}(kT) \quad (3.7)$$

Where $\mathbf{A}_d$ and $\mathbf{B}_d$ are dependent on the sampling period T and $\mathbf{u}(t) = \mathbf{u}(k)$. As in (OGATA, 1995), those matrices can be defined by:

$$\mathbf{A}_d(T) = \exp \mathbf{A}T \quad (3.8)$$

$$\mathbf{B}_d(T) = \left(\int_0^T \exp \mathbf{A}\lambda d\lambda \right) \mathbf{B} \quad (3.9)$$

Finally, in order to obtain the matrices $\mathbf{A}_d$ and $\mathbf{B}_d$, the MATLAB command are used:

```
1 >> sys_d = c2d(sys,T)
2
```

or

```
1 >> A_d = expm(A*T)
2 >> B_d = A\ (A_d-eye(2))*B
3
```

3.3 Coupled Dynamics

3.3.1 Simplified Coupled Dynamics

For the Adaptive Controller presented in the Chapter 4, human and exoskeleton are understood as a perfectly coupled system - with no displacement between them.

$$\tau_h(t) + \tau_e(t) = (J_h + J_e) \ddot{\theta}_{eq}(t) + (b_h + B_e) \dot{\theta}_{eq}(t) + (k_h + k_e) \theta_{eq}(t) \quad (3.10)$$

With this, inertia, damping and stiffness are simply summed; e.g. the equivalent inertia becomes $J_{eq} = J_h + J_e$, to finally obtain the system dynamics represented by figure 3.3. It would be possible to define the desired system behaviour by defining virtual parameters for the exoskeleton.

Figure 3.3 – Diagram representing workspace and coupled plant. θ_h and θ_e are equal, consequently, there is no relative displacement between human and exoskeleton.

3.3.2 Coupled system dynamics with relative displacement

Although the solution presented in the previous section is simple to implement on a controller, it has some limitations: for instance, it does not take into account the relative movement between user and exoskeleton. The dynamics of the interface between human and robot, which can be measured with the load cell, is one of the key aspects to estimate the user movement intention and, therefore, must be taken into account. Figure 3.4 shows the modelling of the soft tissues around the human bone and the attachment system of the exoskeleton.

Figure 3.4 – Diagram representing workspace and coupled plant. θ_h and θ_e are different, resulting in a contact torque τ_c in the interface.

If human and exoskeleton joint are coupled, but have different positions θ_h and θ_e , a contact torque τ_c will act to bring them together. The dynamics of the interface can be modelled as being stiffness, damping and no inertia:

$$\tau_c = B_c \dot{\theta}_c + K_c \theta_c \quad (3.11)$$

The equations 3.1 and 3.3 then becomes:

$$\tau_h(t) - \tau_c(t) = J_h \ddot{\theta}_h(t) + b_h \dot{\theta}_h(t) + k_h \theta_h(t) \quad (3.12)$$

$$\tau_e(t) + \tau_c(t) = J_e \ddot{\theta}_e(t) + B_e \dot{\theta}_e(t) + k_e \theta_e(t) \quad (3.13)$$

Combining equations 3.12 and 3.13 we obtain:

$$\tau_h(t) = J_h \ddot{\theta}_h(t) + b_h \dot{\theta}_h(t) + k_h \theta_h(t) + J_e \ddot{\theta}_e(t) + B_e \dot{\theta}_e(t) + k_e \theta_e(t) - \tau_c(t) \quad (3.14)$$

The final goal is to estimate the human movement intention so that the exoskeleton can assist that movement. For assistance behaviour, we assume that τ_h is very low, but represents the user movement intention. With equation 3.14 it is possible to estimate that human torque τ_h given that we know b_c , k_c , the human arm admittance, the exoskeleton impedance and the exoskeleton position. This modelling has been successfully implemented in (NAGARAJAN; AGUIRRE-OLLINGER; GOSWAMI, 2015), where stability criteria are

shown.

The values of b_c and k_c may be estimated with experiments, or taken from the literature, such as $b_c = 9,474[N.m.s/rad]$ and $k_c = 1905,043[N.m/rad]$ from (NAGARAJAN; AGUIRRE-OLLINGER; GOSWAMI, 2015). The values of the human arm admittance may also be obtained from literature as in (KISTEMAKER; ROZENDAAL, 2011) or experimentally. Values of exoskeleton impedance can be defined experimentally as shown in (SOUIT, 2016). The exoskeleton position is measured by the motor encoder and hall sensor.

4 ADAPTIVE CONTROL

Due to the nonlinear nature of the human movement, a controller which does not rely on a precise knowledge of the system parameters is needed. Studies indicate that MRAC is effective for nonlinear control (BI; YANG, 2014) and real time estimation of unknown robot parameters (SHARIFI; BEHZADIPOUR; VOSSOUGH, 2014).

This chapter presents an implementation of Model Reference Adaptive Control (MRAC) where human and exoskeleton compose the plant, as represented in the equation 3.10. Simulations confirm the effectiveness of parameter estimation, the control robustness to different input functions and, for a rehabilitation exoskeleton, the controller could be applied to assist human movement by reducing the strength needed for the user to make a movement and estimate the human arm dynamics.

4.1 Model-Reference Adaptive Control

The Model-Reference Adaptive Control (MRAC) has the structure shown in Figure 4.1:

The plant is the simplified coupled system dynamics is given by Eq.(3.10), representing the displacement resulting from an applied torque to the human joint. There are uncertainties with respect to the parameters J_h , b_h and k_h .

Figure 4.1 – Model Reference Adaptive Control

The reference model is defined accordingly to the desired system behaviour and depends on the nature of the exoskeleton task. For a rehabilitation exoskeleton, for example, an Assistive Behaviour is desired. An assistive exoskeleton provides a longer human joint displacement θ_h for the same applied human torque τ_h - or same displacement with less torque. In order to achieve that, the Model Reference impedance parameters $[J_m \ b_m \ k_m]$ are set as being smaller than the coupled system parameters $[J_{eq} \ b_{eq} \ k_{eq}]$:

$$\tau_h(t) = J_m \ddot{\theta}(t) + b_m \dot{\theta}(t) + k_m \theta(t) \quad (4.1)$$

The controller is responsible for maintaining the plant tracking the desired dynamics. In order to achieve stability, the reference model must be stable with the same order of differential equation as the controlled plant. Therefore, the model's poles must be on the left half-plane.

4.2 Adaptive Control of a Linear System With Full State Feedback

The design approach considers the most simple problem possible and then, if the performance is below the requirements, a robustness evaluation and design is done. Choices of control and adaptation law can be quite complicated, while analysing convergence is quite simple. For that reason, the design of MRAC controller is usually done by trial and error (SLOTINE; LI et al., 1991).

The following steps, applies the method presented in (SLOTINE; LI et al., 1991). It is assumed that the full system states are measurable because of the state observer implemented in (FANTINI, 2017). The second-order linear system is represented by:

$$a_2 \ddot{\theta} + a_1 \dot{\theta} + a_0 \theta = u \quad (4.2)$$

Where $\mathbf{a} = [J_{eq} \ b_{eq} \ k_{eq}]^T$ is the unknown coefficient vector, u is a generic input and the desired (virtual) coefficients are $\boldsymbol{\alpha} = [J_m \ b_m \ k_m]^T$, for the reference-model to be tracked by the controller:

$$\alpha_2 \ddot{\theta}_m + \alpha_1 \dot{\theta}_m + \alpha_0 \theta_m = \tau_h \quad (4.3)$$

With τ_h being the torque applied to the joints by the patient muscles. The control law is chosen as:

$$u = \hat{a}_2 \ddot{\theta} + \hat{a}_1 \dot{\theta} + \hat{a}_0 \theta = \mathbf{v}^T(t) \hat{\mathbf{a}}(t) \quad (4.4)$$

Where $\hat{\mathbf{a}}$ is the estimation vector and $\mathbf{v}(t)$ is defined as:

$$\begin{aligned} z(t) &= \ddot{\theta}_m - \beta_1 \dot{e} - \beta_0 e \\ \mathbf{v}(t) &= [z(t) \quad \theta_1 \quad \theta_0]^T \\ \hat{\mathbf{a}}(t) &= [\hat{a}_2 \quad \hat{a}_1 \quad \hat{a}_0]^T \end{aligned}$$

$e = \theta - \theta_m$ is the tracking error and β is the tacking error vector.

From (SLOTINE; LI et al., 1991), the adaptation law is given by:

$$\dot{\hat{\mathbf{a}}} = -\Gamma \mathbf{v} \mathbf{B}^t \mathbf{P} e \quad (4.5)$$

And:

$$\mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

4.3 Simulations and Results

In order to verify the control design, simulations were made using MATLAB for the coupled plant represented by the diagram in Figure 3.3. The vector of the plant parameters is defined a sum of the two coupled inertias, Human Arm Inertia J_h and Exoskeleton Inertia J_e , the equivalent damping b_{eq} and equivalent stiffness k_{eq} . The values were taken from (PACHLER; YABUKI, 2014; AGUIRRE-OLLINGER, 2009):

$$\mathbf{a} = \begin{bmatrix} 0.115 + 0.199 \text{ } Kg m^2 \\ 2.52 + 1.32 \text{ } Nms/rad \\ 8.6 + 5.12 \text{ } Nm/rad \end{bmatrix}$$

Being the tracking error parameters:

$$\beta = [10 \quad 40] \quad (4.6)$$

And the adaptation law parameters Γ and P :

Figure 4.2 – Tracking performance and parameter estimation for a step input of 3 Nm and parameter uncertainty of 20% and reference model set for transparent behaviour

$$\Gamma = \begin{bmatrix} 0.01 & 0 & 0.6 \\ 0 & 5 & 0 \\ 0.6 & 0 & 40 \end{bmatrix}$$

$$P = \begin{bmatrix} 0 & 3 \\ 3 & 1 \end{bmatrix}$$

4.4 Assistive Control

For a preliminary verification of the controller performance, it is necessary to check whether the controller is able to compensate the presence of the exoskeleton. In order to do that, the Model-Reference vector (or desired behaviour) is set as $\alpha = [0.112 \ 2.52 \ 8.6]$: the aforementioned modelled human arm parameters. Also, the initial parameters estimation is 20% smaller than the set plant parameters. Figure 4.2 shows the simulation for a step input U_r , the plant angular displacement and the reference model response θ_m .

The final goal of our control design is to assist the arm movement of the user. In order to achieve an effective assistance, all the model reference parameters must be changed according to the desired behaviour. Figure 4.3 shows the controller performance for $\alpha = [0.115 \ 0.95 \ 2.58]^T$ in opposition to the previously set behaviour.

Figure 4.3 – Tracking performance and parameter estimation for a step input of 3 Nm with zero prior knowledge of the plant and assistive control

4.5 Parameter Uncertainty and Estimation

The adaptive control has the advantage of dealing with unknown parameters and to estimate them. Figure 4.3 shows the system response to a step function for an initial estimative vector $\hat{\mathbf{a}} = [0 \ 0 \ 0]$. Figure 4.4 shows the system response to a sinusoidal function for the same initial estimative vector. It is possible to observe the performance getting better as the parameters estimation converge. A more complex reference signal - sin in opposition to a step - provides a better performance for the estimation.

4.6 Controller Robustness

In order to stress the controller, it was also tested with a smoothed square signal. Plant response and parameters estimation can be seen in figure 4.5.

Controller robustness is verified in the simulations shown with respect to dealing with uncertainties. Also, the parameters J_{eq} , b_{eq} and k_{eq} are varied up to 30% to simulate plant unpredicted behaviours. The result can be seen in figure 4.6.

4.7 Discussion

The adaptive controller (MRAC) presented here was able to assist movement and simultaneously estimate the plant parameters. The controller was able to implement an assist-as-needed behaviour increasing the angular displacement for the same input torque in more than 340% from figure 4.2 to figure 4.3. It was found experimentally that the response

Figure 4.4 – Tracking performance and parameter estimation for a sinusoidal input of amplitude 3 Nm with zero prior knowledge of the plant and assistive control

Figure 4.5 – Tracking performance and parameter estimation for a smoothed square input of amplitude 3 Nm with zero prior knowledge of the plant and assistive control

Figure 4.6 – Tracking performance and parameter estimation for a smoothed square input of amplitude 3 Nm with zero prior knowledge of the plant, assistive control and plant parameters variation

was bounded as in the robustness test for parameters variation up to 30%. An important aspect was to show that the MRAC controller would be able to identify the parameters and adapt with time. Figure 4.7 shows the variation of the tracking error for a sinusoidal input and it can be seen the improvement with time. This estimation behaves properly even in the case of an almost square wave input in figure 4.5. Therefore, it can be concluded that the model adapts and it can be used to identify the parameters of the coupled system. Finally, the behaviour of the controller has been tested successfully with time-varying parameters in figure 4.6. Although this is not a test of the controller behaviour with a non-linear, it is possible to consider a local linearisation of the plant model around a certain equilibrium point.

In the next chapter, the implementation of a model reference controller on an exoskeleton development platform is presented.

Figure 4.7 – Tracking error for sinusoidal input

5 DEVELOPMENT PLATFORM

More than a physical experimental workbench, this chapter describes a development platform for exoskeleton research and the results achieved with it. The development of this platform took under consideration the goals defined in the chapter 1 and is composed by the following elements:

- Mechanical Hardware
- Control Hardware
- Software

This platform offers different mechatronic solutions with the design rationale available in this document. The modularity is strongly present: it is possible to develop one element further, find other solutions and fit them back in the platform. The possibility to develop these elements in parallel, by different teams, is a property of modularity that contributes to collaborative work.

The work presented in this chapter is the result of this proposed iterative design process: having set some requirements, to design the solutions, select one, implement and evaluate the result.

Therefore, the mechanical hardware was tested with three different prototypes (as described in chapter 2), the control hardware was modified with the swap from PC104 to Arduino and the implementation of Controller Area Network (CAN bus) communication. The software has two major functions: to communicate the Arduino via CAN bus with the EPOS2 and to implement the impedance control.

5.1 Mechanical Hardware

The Mechanical Hardware refers to the parts that physically interacts with user and the structure that supports the joint and the control hardware. As mentioned before, the actual ModExo system uses the joint solution from the ExoLoLi prototype presented in (SQUIT et al., 2016).

The load cell is another very important legacy from the ExoLoLi prototype: it provides the contact torque between user and exoskeleton. Further described in (SQUIT, 2016), the load cell is a transducer that creates electrical signal from force input. It has strain gauges,

assembled in full Wheatstone bridge as seen in 5.1 and is designed to elastically deform, to optimize the strain gauge sensitivity, when force is applied.

Figure 5.1 – The strain gauges are glued to the surface of the load cell, when the strain gauges deform, due to compression or tension, the electrical resistance changes. Source: <http://www.floridascale.com/customer-faqs.html>

5.1.1 Wire Protection

The final result after wiring the designed load cell is shown in Figure 5.2. It is possible to observe that the wires could be damaged when coupling the system to a human for testing.

In order to prevent that and to increase the wearing comfort, the following goal was set: “The user arm must not be in contact with the wires”.

The solutions considered included wire repositioning, use of isolated copper wire, for being thinner than the actual used or building a protection case for the wires. The last solution

Figure 5.2 – Wiring and the strain gauge are exposed

was chosen for being faster to implement with a 3D printer and for possible unfoldings in the project. The following subgoals were determined to guide the design of the case:

- The wire protection must be easy to fit and remove
- The wire protection must be compact
- The wire protection must provide comfortable interface

5.1.2 ModExo Portable Case

In the beginning of this project, there was no adequate experimental workbench at the Biomechatronics Laboratory for hardware and software testing: the wiring was exposed, the control hardware was spread over the table, there was no emergency button, amongst other issues. When the control tests began, the need for more robustness, safety and structure became evident. In Figure 5.4 it is possible to see a first prototype of the new set-up.

After the first round of prototyping, it became clear that a case for the hardware protection and portability was needed. The following requirements for an experimental set-up were determined:

- All the Mechanical Hardware and control hardware must fit a portable case
- The height of the case must be as low as possible
- There should be an emergency that cuts the power supply
- There should be communication between the control hardware inside the case and the equipment outside the case
- There should be space for additional control hardware, in the case that another degree of freedom is added
- The control hardware must be visible with the case closed
- The interior of the case must be easy to access for maintenance
- There must be an exit for the air expelled by the power supply fan
- The motor must be supported by a rigid structure that can support the weight of a person's arm

The definition of those requirements were very useful for designing the case in collaboration with Priscila Veiga who would later build the case. The result can be seen in Figure 5.6.

Figure 5.3 – Wire Case Prototyping. Work was done in collaboration with Matheus Guinezi and Gabriel Ascânio. A next prototype of this case could include attachments proper human-exoskeleton coupling

Figure 5.4 – Still very far from the project needs, this new set-up already helped with portability, robustness and readiness for tests.

5.2 Control Hardware

The final prototype of the ModExo system is divided in two application layers: the High Level Layer (HLL) and the Low Level Layer (LLL). The HLL consists of the central control processing. The LLL consists of actuation nodes or joints, composed by sensors, motor and driver. The number of nodes may vary depending on the application (e.g., for elbow rehabilitation, only one actuation node is necessary; for elbow and wrist, two actuation nodes will be needed). This architecture aims at obtaining a flexible exoskeleton, which can increase or decrease the number of working joints, a property desired for modular systems.

5.2.1 High Level Layer

The High Level Layer (HLL) consists of a central controller, which operates in real time, receives data from the nodes' sensors, processes the data to obtain relevant state parameters and sends the desired joint angular position to the driver. The central controller is composed by the Arduino UNO and the SeeedStudio CAN bus Shield.

As a desired property, regardless of the number of nodes, the HLL should not change. Consequently, the software must be modular, which is a future development to be made.

Figure 5.5 – A temporary case for part of the control hardware allowed the Arduino and the Load Cell to be moved around, making the software development possible anywhere. The components are: an Arduino, an absolute encoder, an amplification and filtering board and the wiring.

Arduino UNO

Arduino UNO is a microcontroller board based on the ATmega328P microprocessor. It was chosen for being a fast prototyping tool and its successful implementation in many robotic applications. There is plenty of documentation shared on the internet which provides many different solutions for problems that might appear during the implementation of the system. It is smaller and cheaper than the previous embedded hardware (PC104) and also has libraries that communicate directly with MATLAB, the simulation tool actually used in the Biomechatronics Laboratory. In this project, Arduino acts as the High Level Controller (HLC), implementing the Impedance Control.

Figure 5.6 – In this picture it is possible to see most of the hardware of the final prototype. The CAN bus has extra slots in the case of connecting new nodes. The subject's elbow rotation axis would be aligned with the Origin Z Axis. An extension could be fit in the Load Cell Link.

SeeedStudio CAN bus Shield

A CAN bus interface is also needed to manage communication between the central controller and the rest of the system. Most available boards adopt the MCP2515 CAN bus controller and the MCP2551 CAN transceiver. For this project, we chose the Seeedstudio CAN bus Shield V1.2 which has SPI interface up to 10 MHz and CAN communication up to 1 Mb/s. The reason for this choice was the reliability of the manufacturer, the documentation available, and because the board is a shield that plugs directly on the top of the Arduino, offering a robust connection between them.

5.2.2 Low Level Layer

The Low Level Layer (LLL) consists of the actuation nodules of the system. These nodules communicate with the master device - the Arduino - and are composed by the EPOS2 70/10, the Maxon Motor EC90 flat and the sensors.

A property of the LLL is that the number of nodes will increase as the number of degrees of freedom of the system increase. A future hardware development to be made is that the attachment of new joints should be easy, plug'n play, similar to what is seen with LittleBits (<https://www.youtube.com/watch?v=MMYqtCUN5X4>).

EPOS2 70/10

EPOS2 is a digital positioning controller manufactured by Maxon which offers an integrated solution of driver and motor. Chosen for its flexibility, it has a wide variety of

Figure 5.7 – An overview of the control architecture. The interaction between the motor, the load cell and the human joint is not exactly part of the control architecture, but was included for illustration purposes.

operating modes, receives external commands via analog and digital inputs and communicates via USB, RS232 and CANOpen. In this project, it acts as the Low Level Controller (LLC), receiving position commands from the Arduino and transmitting them to the motor.

Maxon Motor EC90 Flat

The Maxon Motor EC90 Flat comes with the MILE encoder and was chosen for its high torque-velocity curve. It also offers high power-to-mass ratio, with a slim design, it is easy to fit in an exoskeleton. It is a 90 Watt brushless motor, with Nominal Torque of 444mNm and Nominal Speed of 2590 rpm and weights 600g.

Sensors

In order to implement impedance control, it is necessary to have position information and contact torque information. Therefore, each node needs at least one sensor providing position and another providing contact torque.

The exoskeleton position information is given by the Maxon Motor, which is integrated with two sensors: a hall sensor and an encoder.

The contact torque information is given by the load cell with strain gauges assembled in Wheatstone Bridge. That torque is a combination of the human joint torque and the exoskeleton joint torque.

The MA3 Encoder is a 12-bit resolution digital encoder used for manual positioning, for debugging and other functions.

The signals from the MA3 encoder and the load cell are acquired by an amplification board that sends both sensor information to the CAN bus. This board was developed in the Biomechatronics Laboratory and also filters and amplifies the strain gauge signal.

CAN bus Analyzer

The CAN bus Analyzer, seen in figure ?? is a hardware developed by Microship. Although it is not exactly part of the ModExo system, it is very useful for CAN Network debugging, allowing messages to be sent and read through its Graphical User Interface.

5.3 Software

This section will describe the EPOS Studio, the standard interface used for controlling the EPOS, the CAN bus communication protocol, the main code, developed in C/C++ with Arduino functions and the Telemetry Viewer, an open source interface adapted for the ModExo application.

5.3.1 EPOS Studio

The EPOS Studio is a robust and functional Graphical User Interface used for configuring the EPOS2 Driver. It offers many features, the most important for our application being: PID calibration, Object Dictionary and Position Mode, Homing Mode.

The object dictionary contains all parameters that define the EPOS functioning such as Actual Position, Target Position, Maximum Velocity, Derivative Gain, amongst others. You can change those parameters manually via the EPOS Studio interface.

The EPOS Studio, however, has some limitations: the control modes are pre-defined, so it is not possible to write a different algorithm, for example to use external sensor data, such as the load cell. Also, it demands a computer connected to it the whole time during the system operation, which affects the portability of the system.

With an external controller, such as the Arduino, we can bypass the EPOS Studio by writing and reading the objects from the objects dictionary. Since EPOS is a CANopen slave, it can be commanded by CAN, USB or RS232.

5.3.2 Telemetry Viewer

The Telemetry Viewer is an open source graphical user interface project hosted in: http://www.farrellf.com/projects/software/2017-02-11_Telemetry_Viewer_v0.3/ that is used for real time data visualisation with Arduino. It has a very simple interface, with possibility of plotting multiple graphs, with many different options. Since the project is open source, we managed to implement another functionality: to send messages via serial, other than just to visualize them.

5.3.3 CAN bus

A major concern of this work is due to the control architecture and its implementation. We wish a modular architecture, which allows multiple nodes to input and to read information.

The CAN standard was initially designed for automotive use. It offers robust communication for several different Electronic Control Units such as sensors, controllers and others subsystems. The physical set-up consists of a high wire and a low wire separated by two resistors of 120 Ω at their end. The host and the nodes may connect to these wires and get access to the system. The structure of the communication protocols makes the attachment of different applications simple, which is convenient for the proposed modular structure. It has been verified as been a reliable solution for exoskeleton application (ZHANG; SHI; ZHANG, 2014).

Communication with EPOS2 driver is established via Controller Area Network (CAN) interface which follows the CiA CANopen specifications (<http://www.can-cia.org>) DS-301

Figure 5.8 – The Telemetry Viewer was adapted for our application: a text box was added in a way that it is possible to send messages to the Arduino via serial. The interface shows four plots for the differentialControl Mode: loadcell data, reference position, exoskeleton position and EPOS current. The layout can be divided to plot more or less data.

Figure 5.9 – ISO 11898 Basic Network Set-up. Source: (MOTOR,)

communication profile and DSP-402 device profile (Device Profile for Drives and Motion Control).

Data is sent in “Little Endian” format, or else, Least-Significative-Byte-First. In CAN network, objects are addresses pointing to data location. Objects have 8 bytes, arranged in the following structure:

Table 5.1 – CAN Message Protocol

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
command	index	index	sub-index				

Objects can be read in order to obtain information about system parameters (such as *actual current*, *actual position*, *max acceleration*, etc.) and can be written, for commands and system configuration. Data writing is made with the command 22 and reading, with the command 40.

Communication is established via *Process Data Objects (PDO)* and *Service Data Objects (SDO)*. PDOs are non confirmed messages, which makes it a faster than SDO. It is typically used for data that changes a lot along the process, therefore is used for the sensors. SDO transmission uses confirmed messages and accepts transmissions of any size. It is usually used for setting up the device, for example to configure the PDOs.

CAN_BUS_Shield is the Arduino library used for CAN communication. Its documentation is available at: https://github.com/Seeed-Studio/CAN_BUS_Shield. The main functions implemented up to this moment in the code are:

Check Receive

```
1 INT8U MCP_CAN::checkReceive(void);
```

Check for messages in the CAN Network. Returns 1 if frames arrive.

Get CAN ID

```
1 INT8U MCP_CAN::getCanId(void);
```

Returns the ID of the message sender. Sensor data is sent in the network with the respective ID's:

- 181: EPOS Current Data
- 321: Amplification board containing data from the MA3 Encoder and the Load Cell
- 381: EPOS Position Data

Send Data

Sends data to the CAN Network with the following structure:

```
1 CAN.sendMsgBuf(INT32U id, INT8U ext, INT8U len, INT8U *buf);
```

Where ID is relative to who sends the message, EXT indicates if the frame is standard or extended, LEN represents the length of the frame and BUF is the content of the message. As an example of a message being sent, we have:

```
1 unsigned char setpoint_position[8] = {0x22, 0x62, 0x20, 0x00, 0, 0, 0, 0};  
2  
3 CAN.sendMsgBuf(0x601, 0, 8, setpoint_position);
```

This message results in the positioning of the motor to the position indicated by the last 4 bytes of the array *setpoint_position*. The first byte, containing the number *0x22* indicates that this message is a command or - that the data in the message will be written in the Object. The Object is specified by the second, third and fourth bytes, the Index (LSB), Index (MSB) and Sub-Index, respectively. In the example above, the number 0 is written in the Object *0x2062*, which is relative to Position Setpoint. As result, the motor will return to its initial position.

Receives Data

```
1 CAN.readMsgBuf(INT8U *len, INT8U *buf);
```

Receives and stores the data from the CAN Network in the variable BUF.

5.3.4 Main Arduino Code

The code was developed to read data from the sensors and make decisions based on that information. The code is written prioritizing Self-Documented Code over Commented Code. The overall idea is that comprehensible code should have descriptive functions and variable names and avoid comments unless extremely necessary.

A simple user interface was implemented by Vitor Sternlicht which allows to switch between operation modes as in a state machine. While the main loop is running, the user may send message strings via Serial Monitor or Telemetry Viewer defining the desired operation. For example, the sending of the letter “q” will start Encoder Control operation mode.

In order to stablish communication, the *CAN_BUS_Shield* library described above is used. The actual *ModExo* code directs most of the messages to the ID 601 because we are only dealing with one degree of freedom yet. The code should be modular: at some point, it shall be able to determine the nodes connected to the network and communicate to them accordingly.

The code is structured in a way that the sensor data is read in the same function as the control is done: the setpoint is calculated after each Load Cell data arrives. Since this information is sent with a periodicity of $50[ms]$, the control algorithm runs only at $50[ms]$ sample time. In the future, this function should be broken in at least one function for data reading and another function for setpoint calculating.

The main functions are described below:

doStartup()

This function initializes the EPOS2 and defines operational configurations, such as “Max Acceleration”, “Max Following Error”, amongst others.

PDOConfig()

This function configures the EPOS2 to send position and current data on the network via PDO. The configuration determines that this data will be sent in the network every time a SYNC message is sent.

sync()

A simple message, sent to the ID 80 (meaning “everybody”) containing only an empty byte message. This message causes all configured PDOs to send their actual information.

In order to send the sync, an interruption was configured at 1kHz.

encoderControl()

Enables the motor to be manually positioned with the use of the MA3 Encoder.

This function reads the encoder position, converts it to quadrature counts (qc) and sends it as a setpoint to the motor. Lucas Novak has been working to improve this function and merge it with differentialControl().

differentialControl()

Implements a simple Impedance Control algorithm that calculates a new target position depending on the Load Cell data. The function runs an Euler integration which can be changed for a 4th Order Runge-Kutta integration in the future.

```
1 x_1 = x_1 + dt*x_2;  
2 x_2 = x_2 + dt*x_3;  
3 x_3 = 1/J_d*(-b_d*x_2 - k_d*x_1 + contactTorque);
```

x_1 , x_2 and x_3 represent, respectively, position and its first and second derivatives.
 dt is the sampling time.

dataPrint()

Gathers and prints all the data we wish to visualize. Data generated comes in .csv format which can be read by many softwares such as Excel, MATLAB, etc.

The variables actually printed are: reference position, velocity and acceleration, contactTorque, EPOS Current, EPOS Position, MA3 Encoder position and the Time-stamp. It's very important

to reflect before changing those printed variables, for we wish data to generated in a standard in a manner that it is possible to compare results across time.

6 PERFORMANCE ANALYSIS

The previous chapter showed the design and construction of a development platform for exoskeleton research. In this chapter, more details regarding the implementation of the control are shown: the control block diagram, the calibration of the load cell, the discretization of the system, delay tests and impedance control, the respective performance analysis and directions to improve it.

The .csv files containing data generated by the Arduino function `println()` are shared in the GitHub folder: <https://github.com/biopmr/ModExo/tree/master/experimental/csvData>, with images files containing, print-screens, photos, etc., containing graphs plotted in MATLAB, settings, set-up other information. Also, in the GitHub folder <https://github.com/biopmr/ModExo/tree/master/matlab> the scripts and workspaces used in this chapter are available.

The overall control scheme is represented below. In this implementation, there is no feedback from the system to the Arduino, the only inputs to the Arduino are the load cell and the MA3 Encoder. The calculated reference position will depend on the selected control mode.

Figure 6.1 – The HLC determines the position setpoint and sends it to the LLC. The LLC implements a PID that commands the motor to the desired position.

Tests were performed with sampling rate of approximately 50ms.

6.1 Delay Tests

In these tests, the controller mode used was “EncoderControl”, in which the reference position is given by turning the MA3 encoder by hand. The joint response to a manual step input was tested for three different derivative gains. The first PID gains, in Figure 6.2, were defined by the Regulation Tuning Wizard from EPOS Studio 2. We hypothesised that the delay would decrease as the Derivative Gain increased. So, for the Figures 6.3 and 6.4, the derivative gains were simply increased to verify the change in the system response.

Figure 6.2 – Delay test with D-Gain = 6919

Figure 6.3 – Delay test with D-Gain = 15000

In table 6.1 are listed the datasets and the respective Lower Level Controller (the EPOS) PID gains. These datasets are the CSV data generated as the output of the ModExo code and are available in the GitHub.

Table 6.1 – Delay Tests Data

Test	P-Gain	I-Gain	D-Gain	Overshoot	Settling Time(s)	Delay(s)
Delay 1	2533	4772	6919	0	1.67	0.76
Delay 2	2655	4948	15000	0	0.91	0.1
Delay 3	2655	4948	30000	0	1.01	0.1

Figure 6.4 – Delay test with D-Gain = 30000

6.1.1 Results

The graphs show two variables: the Theta Reference - in blue - corresponding to the signal of the MA3 Encoder, which gives the reference position for the low level controller and the Theta Tracking - in red - which corresponds to joint position given by the Joint Position Sensor.

The variable Delay is defined as the time gap between the instant of change in the theta reference and the instant of change in the theta tracking. As seen in the graphs, when the Derivative Gain is increased from 6919 to 15000 the Delay goes from 0.76 to 0.1 seconds, however, when increased from 15000 to 30000, no change is observed in the Delay.

The variable Settling Time is defined as the time gap between the instant when the theta reference reaches its final position and the instant when the theta tracking reaches its final position. Similarly to the Delay, when the Derivative Gain is increased from 6919 to 15000 the Settling Time goes from 1.67 to 0.91 seconds, however, when increased from 15000 to 30000, no significant change is observed in the Settling Time.

6.1.2 Discussion

Since the system dynamics will vary depending on the application, we are not concerned on finding an optimal relation between Derivative Gain and Delay: the focus is to understand the overall behaviour of the system.

The results indicate that an increase in the Derivative Gain will reduce the system response time up to a limit, which could be explained by a saturation of the actuators.

Further tests can be performed to observe the influence of the Integrative Gain, which

should also result in a decrease of the Settling Time. Also, the step input can be made by software, to provide repeatability and other analysis possibilities.

6.2 Load Cell Calibration

Figure 6.5 – Load Cell Test Set-up

6.2.1 Software Calibration

The load cell calibration is defined after the amplified signal from the wheatstone bridge. That signal is the load cell force (F_L), the input of the equation:

$$\tau_c = A \times F_L + B$$

A and B , respectively the slope and the offset, are constants that provide the relation between τ_c and F_c . The calibration shown in the code below converts the raw data from the variable *force*, relative to F_c , to the variable *loadcell_data_double*, relative to τ_c , in mNm:

```

1 loadcell_data_double = A*(force) + B
2
3 float A = 0.00764; // slope steepness
4 float B = 118712.7+5000; // offset
5 float d = 0.105; // distance between the load cell centers = 10,5cm
6
7 contactForce = (loadcell_data + B)*A;
8

```

Figure 6.6 – Load Cell Test 1

Figure 6.7 – Load Cell Test 2

6.2.2 Static load cell tests

Static load cell tests were performed to evaluate the software calibration. The joint was positioned in the horizontal position, at $\theta = \pi/2$, while different loads were attached to the extremity of the load cell as shown in the Figure 6.5.

Three different loads were applied to the load cell in sequence: $Load1 = 5000g$, $Load2 = 2137.4g$ and $Load3 = 528.8g$. This procedure was made twice.

6.2.3 Results

The graphs show two variables: Load Cell Data and Calculated Torque. The first corresponds to the acquired data with the LoadCell. The second is the theoretical Torque, calculated according to the different loads applied to the joint arm. It's also possible to notice a noise, which happens with the changing of the loads.

The table 6.2 shows the theoretical calculated τ_T torque for each of the different loads, the mean measured contact torque $\bar{\tau}_c$, the mean absolute error $\bar{e}_{abs}$ and the mean relative error $\bar{e}_{rel}$.

Mean Relative Error Load 1 = 0.0551 Mean Relative Error Load 2 = 0.0374 Mean

Relative Error Load 3 = 0.339

Table 6.2 – Load cell absolute and relative error for different loads.

Loads	e	$e\%$
Load 1	2533	4772
Load 2	2655	4948
Load 3	2655	4948

6.2.4 Discussion

From the plots and the table shown above, we may observe that there was a similar behaviour between Test 1 and Test 2: very close absolute error and relative error for each applied load.

The Load 2 showed poorest performance for the relative error reached more than 33% and the value of τ_c did not stabilized: it kept dropping until another load was applied. A further evaluation of the reason for this behaviour is needed.

As a suggestion for the works to come, a study of the range of the contact torque during daily - or specific - activities should be done. This study would help to optimize the load cell calibration for that range.

6.3 Model Reference Dynamics Control

The model reference controller is requirement for the implementation of the MRAC. Therefore, the High Level Controller calculates the reference position to be achieved by the Low Level Controller. The key feature desired is the possibility to modulate it's admittance via software. This section presents a controller which implements a simple integration algorithm that calculates the reference position of the joint according to the given contact torque input. With such reference model, it is possible to define a desired dynamic behaviour for the joint - that is, its admittance. In figure 6.8, it is possible to observe the implemented control diagram.

In order to implement this behaviour via software, we must discretize the system model. In that case, the discretized system becomes:

$$\tau_c[k] = J_d\theta[k+2] + b_d\theta[k+1] + k_d\theta[k]$$

The differential equation becomes a differences equation:

$$\theta[k+2] = \frac{1}{J_d}(\tau - b_d\theta[k+1] - k_d\theta[k])$$

J_d , b_d and k_d are the weights of the variables $\theta[k]$, $\theta[k+1]$ and $\theta[k+2]$ and, therefore, define the system behaviour in discrete time. We assume J_d , b_d and k_d to be unknown. They

Figure 6.8 – The High Level Controller calculates the reference position based on the Contact Torque. The information is sent via CAN network to the EPOS2 which commands the Motor with current.

will be estimated by the observation of the system response to a torque step.

This modelling is implemented via Arduino with Euler Integration by the following code:

```

1 x_1 = x_1 + T*x_2;
2 x_2 = x_2 + T*x_3;
3 x_3 = 1/Ja*(-Ba*x_2 - Ka*x_1 + contactTorque);

```

Where x_1 , x_2 and x_3 represent, respectively, position and its first and second derivatives. T is the sampling period and `contactTorque` was calculated as shown in the subsection Software Calibration. Note that the virtual parameters J_a , B_a and K_a could have a physical correspondence to Inertia, Damping and Stiffness (J_d , B_d and K_d). However, because the process is discrete, they have no physical meaning and can be simply assumed as coefficients multiplying x_1 , x_2 and x_3 .

6.3.1 Model Reference Tracking

As already mentioned, in the control structure, the High Level Controller calculates the reference model position for a given input. The Low Level Controller's task is to make the joint position as close as possible to that virtual position. In figures 1 and 2, it is possible to observe the tracking performance for the joint with no extra mass.

Figures 6.9 and 6.10 show two plots, θ_m refers to the Model Reference Position calculated by the Arduino when submitted by a 50mNm torque input. θ_e refers to the joint position given by the MILE Encoder in the motor. It is possible to observe that the curves are really close. Further experiments are required with higher mechanical impedance in the joint, however, for the analysis that will follow, we will consider that there is no significant difference between the virtual impedance imposed by the Reference Model and the observed impedance performed by the Plant.

Figure 6.9 – Model Reference Tracking for a torque step of 50mNm

Figure 6.10 – Model Reference Tracking for a torque step of 50mNm

Table 6.3 – Differential Control System Identification

Ja	Ba	Ka	Ka/Ba	Kd/Bd	Kc/Bc	Ad		Ac	
0,05	0,2	2	10			0	1	0	1
				-0,4642	4,4983	-0,8342	1,797	-16,3099	-3,6258
0,05	0,2	2	10			0	1	0	1
				-0,4643	4,5025	-0,8344	1,7972	-16,3057	-3,6215
0,05	1	2	2			0	1	0	1
				-0,3744	1,2534	-0,5829	1,5569	-13,5293	-10,7942
0,05	1	2	2			0	1	0	1
				-0,3723	1,2956	-0,577	1,5497	-14,248	-10,9972
0,05	1	5	5			0	1	0	1
				-0,2520	1,6798	-0,3178	1,261	-38,5065	-22,9237
0,05	1	5	5			0	1	0	1
				-0,2488	1,6718	-0,3124	1,2554	-38,8987	-23,2678

6.3.2 System Identification

The position generated by the Arduino from the previous section is analysed with the MATLAB toolbox *System Identification*. With the input-output data, this tool creates models that estimates the system parameters and reproduces the observed response.

We wish to investigate the influence of virtual dynamics imposed by the coefficients J_a , B_a and K_a in the actual dynamics of the system. Three variations of J_a , B_a and K_a were tested for two different input torques ($50mNm$ and $100mNm$), resulting in a total of 6 datasets. With *System Identification* Toolbox, the systems were identified with the settings:

1. State Space Model Estimation
2. Model Order: 2
3. Both Continuous and Discrete-time, with sampling rate of $0.05s$
4. Model Structure Form: Observability Canonical

In the table 6.3, the first three columns correspond to the parameters set in the Arduino's Differential Control mode. The columns "Ad" and "Ac" correspond to the estimated discrete and continuous-time state spaces A matrix in the canonical form.

6.3.3 Discussion

As described in Chapter 3, the state space representation contains the impedance parameters in the terms $\frac{K}{J}$ and $\frac{B}{J}$. In order to evaluate if there is a correspondence between

the virtual dynamics and the identified system dynamics, the ratio between stiffness and damping is analysed. The columns $\frac{K_a}{B_a}$, $\frac{K_d}{B_d}$ and $\frac{K_c}{B_c}$ contains those ratios.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{K}{J} & -\frac{B}{J} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{J} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad (6.1)$$

When comparing the virtual parameters ratio $\frac{K_a}{B_a}$ with the discrete parameters ratio $\frac{K_d}{B_d}$ there is no apparent correlation. However, when comparing the virtual parameters ratio $\frac{K_a}{B_a}$ with the continuous parameters ratio $\frac{K_c}{B_c}$, although non linear, it is possible to observe a trend that correlates them: the highest $\frac{K_a}{B_a}$ ratio remains the highest $\frac{K_c}{B_c}$, that is also true for the lowest ratios.

When the sampling times tends to zero, the discrete-time system becomes closer to the continuous-time system. This is coherent to the observed behaviour of the robotic joint, where it is possible to feel a physical change in damping and stiffness when the correspondent virtual parameters are changed.

Since the Arduino runs in discrete time, virtual parameters are discrete too. The lack of correlation between the virtual parameters and the discrete-time parameters could be due to the MATLAB discretization method used.

7 CONCLUSION: FUNCTIONAL AND NON-FUNCTIONAL ANALYSIS

This work presented the design and prototyping of a development platform for exoskeleton research. Biomechanics was combined with system engineering to conceive a project that comprehends several fundamental aspects of the exoskeleton research. The most significant outcome of this work is definitely more questions and more work to be done.

Human and robot were compared and modelled for different needs. The human limb mechanics is complex: the human brain commands result in a contraction of the muscle which, attached to the human skeleton, creates a lever that moves the human arm. The modelling of this phenomenon is simplified by a second order linear model which is extensively used in literature (BENNETT et al., 1992; OLAYA, 2008; AGUIRRE-OLLINGER et al., 2007; KEARNEY; HUNTER, 1989). This linear model does not perfectly represents the human movement - some claim it's not even close (KISTEMAKER; ROZENDAAL, 2011). However, since the goal is to have a compliant human-exoskeleton interaction, despite the difference in dynamics between them, the interaction achieved with the linear model can offer that compliant interaction. Future work on modelling should include non-linearities to this model, such as gravity. Another interesting investigation would be to implement the Hill Musculoskeletal Model.

Model Reference Adaptive Control was investigated as being a promising solution for exoskeleton control and has been successfully implemented in MATLAB. It has the feature of estimating the system parameters in real time (SLOTINE; LI et al., 1991) and showed good performance in simulations. Model Reference Control experimentally demonstrated satisfactory performance: the High Level Controller calculates the reference position and the Low Level Controller manages to reach it. The impedance parameters, inertia, damping and stiffness were manipulated virtually and indicated physical correspondence in experiments, although that correspondence is not clear. Future work in the Adaptive Control should integrate the Model Reference Control with the Parameter Estimation. In order to do that it is still necessary to adapt the MATLAB code to c language.

The developed software is actually the embryo of a library for controlling the exoskeleton. Most of the developed code is concerned with CAN bus Communication: to determine a setpoint for the Low Level Controller, to read the motor position, velocity and current, to configure communication protocols and to configure system parameters such as maximum acceleration and maximum position. The differentialControl() function implements the Model Reference Control, which calculates the reference position based on the Contact Torque

Figure 7.1 – Final prototype of the workbench designed in this work.

provided by the load cell. It is also possible to virtually set the impedance parameters: inertia, damping and stiffness. An interface for switching between control modes was implemented. Yet, there is much more to be done than it was done so far. Future work can implement Runge Kutta or any other numerical integration instead of Euler. The functions need to be better structured. For example, the `differentialControl()` reads the sensor data, calculates the contact torque and then calculates the reference position. By decoupling this into at least three different functions, the code is made better, easier to understand, faster to read and more modular. The sampling time achieved is of $50[ms]$ and that can be greatly reduced. Also, software safety systems, as position limits, should be developed.

Defined as a functional goal by the systems approach, the classical engineering problem is undertaken: the technological challenge of building, from hardware to software, a functional exoskeleton prototype. Many aspects of the experimental workbench were carefully considered: portability, electronics robustness, modularity, volume, height, aesthetics, etc. In the laboratory, we have, for some time tried to develop a completely portable exoskeleton, with which the user could move around while wearing it. This project takes a step back with the choice of fixing the exoskeleton on the workbench for believing that many technological aspects yet need to be mastered before becoming completely portable. With the developed prototype, the Laboratory of Biomechatronics now has an experimental workbench for testing control strategies, for testing observers (such as Kalman filter), testing other hardware (besides Arduino) and even to make trials with subjects.

Figure 7.2 – The robotic joint supports the elbow and the forearm, with the Joint Axis and the Elbow Axis aligned.

The experimental workbench was designed for motor control and exoskeleton control research. The decisions taken along the project led to a commercially viable solution - meaning that it is easily reproducible at a reasonable cost. The robotic joint supports the elbow and the forearm, with the Joint Axis and the Elbow Axis aligned. From this set-up, it is possible to mention a few applications for the workbench:

- Human limb parameters estimation
- Control strategies testing and evaluation
- Rehabilitation with assisted behaviour
- Muscular training with resistive behaviour
- Motor control studies, as in (QUADRADO; NORIEGA; FORNER-CORDERO, 2014)

Even though the system has safety features, such as emergency button and EPOS position limits, this is still an open issue, for example, position should be limited by the high level controller and by hardware, with mechanical and electrical solutions. Another way to ensure safety is to provide a fragile attachment between user and exoskeleton - this could be designed with the help of a 3D printer. Another open issue may be seen in Figures 7.2 and 7.3: there is a wire coming out of the robotic joint, which is connected to the load cell.

This wire has to be constantly handled during the ModExo operation in order to prevent breaking.

Towards a non-functional goal achievement, an important digital platform was conceived: to improve collaborative work in the Lab and to invest in knowledge management. With GitHub, the software becomes more than a code fit for one project. The access to the knowledge acquired is clearly written in the code and the access to this information is open for anyone: The communication between the High Level Controller and the Low Level Controller is no longer an issue. The mapping of the EPOS2 objects and how to configure them was understood and, most importantly, made available to others. Many projects from the Laboratory already have benefited from this platform, as is seen in this final graduation work which used the CAN bus Communication, and this platform was benefited with the feedback from that use. The publication of the results and software for anyone who wishes to access it is a form of scientific contribution: innumerable are the times that engineering (and any other science) resort to the information available on the internet to solve technical challenges; without this information, this work would have achieved much less.

Figure 7.3 – Haptic interface which adapts the platform developed in this project. The collaboration results beneficial for both projects: the code is developed by two teams instead of one.

Above all, this work represented an opportunity to learn how to work collaboratively, from the technicians of the University to the great theorists of robotics. This work is definitely a work of thousands hands. A lot of effort was invested in understanding how science is conceived and figuring how this researcher believes it should be done. The word collaboration is in everyone's mind, and there is a long way - we have shown in this work - from the brain to begin an actual movement.

REFERENCES

AGUIRRE-OLLINGER, G. *Active impedance control of a lower-limb assistive exoskeleton*. Tese (Doutorado) — NORTHWESTERN UNIVERSITY, 2009. Cited in page 46.

AGUIRRE-OLLINGER, G. et al. Active-impedance control of a lower-limb assistive exoskeleton. In: IEEE. *2007 IEEE 10th International Conference on Rehabilitation Robotics*. [S.l.], 2007. p. 188–195. Cited 2 times in pages 38 and 77.

AL-SUBAIE, H. S.; MAIBAUM, T. S. Evaluating the effectiveness of a goal-oriented requirements engineering method. In: IEEE. *Comparative Evaluation in Requirements Engineering, 2006. CERE'06. Fourth International Workshop on*. [S.l.], 2006. p. 8–19. Cited in page 98.

APKARIAN, J.; NAUMANN, S.; CAIRNS, B. A three-dimensional kinematic and dynamic model of the lower limb. *Journal of biomechanics*, Elsevier, v. 22, n. 2, p. 143–155, 1989. Cited 2 times in pages 19 and 88.

BENNETT, D. et al. Time-varying stiffness of human elbow joint during cyclic voluntary movement. *Experimental Brain Research*, Springer, v. 88, n. 2, p. 433–442, 1992. Cited in page 77.

BI, Q.; YANG, C.-j. Human-machine interaction force control: using a model-referenced adaptive impedance device to control an index finger exoskeleton. *Journal of Zhejiang University SCIENCE C*, Springer, v. 15, n. 4, p. 275–283, 2014. Cited in page 44.

BI, Z.; ZHANG, W.-J. Concurrent optimal design of modular robotic configuration. *Journal of Robotic systems*, v. 18, n. 2, p. 77–87, 2001. Cited in page 89.

DARDENNE, A.; LAMSWEERDE, A. V.; FICKAS, S. Goal-directed requirements acquisition. *Science of computer programming*, Elsevier, v. 20, n. 1-2, p. 3–50, 1993. Cited in page 32.

EXOSKELETON Report. <<http://exoskeletonreport.com/>>. Accessed: 2017-09-30. Cited in page 19.

FANTINI, D. M. Development and implementation of a kalman filter estimator for states reconstruction and parametric identification in an exoskeleton. 2017. Cited in page 45.

FRYER, J. et al. Configuring robots from modules: an object oriented approach. In: IEEE. *Advanced Robotics, 1997. ICAR'97. Proceedings., 8th International Conference on*. [S.l.], 1997. p. 907–912. Cited 2 times in pages 88 and 89.

GITHUB. <<https://github.com/>>. Accessed: 2017-09-30. Cited in page 33.

GUIDE, A. Project management body of knowledge (pmbok® guide). In: *Project Management Institute*. [S.l.: s.n.], 2001. Cited in page 27.

HEAVEN, W.; FINKELSTEIN, A. Uml profile to support requirements engineering with kaos. *IEE Proceedings-Software*, IET, v. 151, n. 1, p. 10–27, 2004. Cited in page 91.

- HILL, A. The heat of shortening and the dynamic constants of muscle. *Proceedings of the Royal Society of London B: Biological Sciences*, The Royal Society, v. 126, n. 843, p. 136–195, 1938. Cited in page 36.
- HOGAN, N. Adaptive control of mechanical impedance by coactivation of antagonist muscles. *IEEE Transactions on Automatic Control*, IEEE, v. 29, n. 8, p. 681–690, 1984. Cited 2 times in pages 23 and 24.
- HOGAN, N. Impedance control: An approach to manipulation. In: IEEE. *American Control Conference, 1984*. [S.l.], 1984. p. 304–313. Cited 3 times in pages 23, 24, and 88.
- JANSCKEK, K. *Mechatronic Systems Design*. [S.l.]: Springer, 2012. Cited in page 89.
- KEARNEY, R. E.; HUNTER, I. W. System identification of human joint dynamics. *Critical reviews in biomedical engineering*, v. 18, n. 1, p. 55–87, 1989. Cited 2 times in pages 38 and 77.
- KISTEMAKER, D. A.; ROZENDAAL, L. A. In vivo dynamics of the musculoskeletal system cannot be adequately described using a stiffness-damping-inertia model. *PloS one*, Public Library of Science, v. 6, n. 5, p. e19568, 2011. Cited 2 times in pages 43 and 77.
- KISTEMAKER, D. A.; SOEST, A. J. K. V.; BOBBERT, M. F. A model of open-loop control of equilibrium position and stiffness of the human elbow joint. *Biological cybernetics*, Springer, v. 96, n. 3, p. 341–350, 2007. Cited 2 times in pages 36 and 37.
- KRUCHTEN, P. *The Rational Unified Process: An Introduction*. [S.l.]: Addison-Wesley, 2004. Cited in page 90.
- LAMSWEERDE, A. V. Goal-oriented requirements engineering: A guided tour. In: IEEE. *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*. [S.l.], 2001. p. 249–262. Cited 3 times in pages 31, 89, and 91.
- LAMSWEERDE, A. V.; LETIER, E. From object orientation to goal orientation: A paradigm shift for requirements engineering. In: *Radical Innovations of Software and Systems Engineering in the Future*. [S.l.]: Springer, 2004. p. 325–340. Cited 2 times in pages 32 and 89.
- LETIER, E. et al. *Reasoning about agents in goal-oriented requirements engineering*. Tese (Doutorado) — PhD thesis, Université catholique de Louvain, 2001. Cited in page 93.
- MACIEJASZ, P. et al. A survey on robotic devices for upper limb rehabilitation. *J. Neuroeng. Rehabil*, v. 11, n. 3, p. 10–1186, 2014. Cited 2 times in pages 16 and 88.
- MATSUMARU, T. Design and control of the modular robot system: Tomms. In: IEEE. *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*. [S.l.], 1995. v. 2, p. 2125–2131. Cited 3 times in pages 16, 88, and 93.
- MIHELJ, M.; NEF, T.; RIENER, R. Armin ii-7 dof rehabilitation robot: mechanics and kinematics. In: IEEE. *Robotics and Automation, 2007 IEEE International Conference on*. [S.l.], 2007. p. 4120–4125. Cited in page 88.
- MIRANDA, A. B. W. *Exoesqueleto Robótico De Membro Superior Com Três Graus De Liberdade Ativos*. Dissertação (Mestrado) — Escola Politécnica da USP, 2014. Cited 5 times in pages 16, 28, 30, 88, and 92.

- MIRANDA, A. B. W. et al. Bioinspired mechanical design of an upper limb exoskeleton for rehabilitation and motor control assessment. In: IEEE. *Biomedical Robotics and Biomechanics (BioRob), 2012 4th IEEE RAS & EMBS International Conference on*. [S.l.], 2012. p. 1776–1781. Cited 2 times in pages [27](#) and [92](#).
- MOTOR, M. *Documentation Communication Guide*. [S.l.]. Cited in page [62](#).
- MOURA, R. et al. Exoskeleton application to assist learning of a coincident timing motor task of the arm using passive mechanical perturbations. In: IEEE. *Biomedical Robotics and Biomechanics (BioRob), 2016 6th IEEE International Conference on*. [S.l.], 2016. p. 984–988. Cited in page [28](#).
- NAGARAJAN, U.; AGUIRRE-OLLINGER, G.; GOSWAMI, A. Integral admittance shaping for exoskeleton control. In: IEEE. *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. [S.l.], 2015. p. 5641–5648. Cited 3 times in pages [24](#), [42](#), and [43](#).
- NASA, N. Systems engineering handbook. *Systems Engineering*, v. 6105, p. 360, 2007. Cited 2 times in pages [15](#) and [26](#).
- NUSEIBEH, B.; EASTERBROOK, S. Requirements engineering: a roadmap. In: ACM. *Proceedings of the Conference on the Future of Software Engineering*. [S.l.], 2000. p. 35–46. Cited 2 times in pages [26](#) and [89](#).
- OGATA, K. *Discrete-time control systems*. [S.l.]: Prentice-Hall Englewood Cliffs, NJ, 1995. v. 8. Cited in page [40](#).
- OLAYA, A. F. R. Sistema robótico multimodal para análisis y estudios en biomecánica, movimiento humano y control neuromotor. *Universidad Carlos III de Madrid. Departamento Ingeniería de Sistemas y Automática*, 2008. Cited 2 times in pages [38](#) and [77](#).
- PACHLER, C.; YABUKI, D. K. Implementação de um controle de impedâncias modular para exoesqueleto robótico de membro superior. 2014. Cited in page [46](#).
- PATOGLU, V. et al. Design requirements for a tendon rehabilitation robot: results from a survey of engineers and health professionals. In: AMERICAN SOCIETY OF MECHANICAL ENGINEERS. *ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. [S.l.], 2010. p. 85–94. Cited in page [99](#).
- PERRY, J. C.; ROSEN, J. Design of a 7 degree-of-freedom upper-limb powered exoskeleton. In: IEEE. *Biomedical Robotics and Biomechanics, 2006. BioRob 2006. The First IEEE/RAS-EMBS International Conference on*. [S.l.], 2006. p. 805–810. Cited in page [88](#).
- PERRY, J. C.; ROSEN, J.; BURNS, S. Upper-limb powered exoskeleton design. *Mechatronics, IEEE/ASME Transactions on*, IEEE, v. 12, n. 4, p. 408–417, 2007. Cited in page [19](#).
- QUADRADO, V. H.; NORIEGA, C.; FORNER-CORDERO, A. Experimental assessment of a coincident timing motor task of the arm under a passive mechanical perturbation. In: IEEE. *Biomedical Robotics and Biomechanics (2014 5th IEEE RAS & EMBS International Conference on*. [S.l.], 2014. p. 616–620. Cited in page [79](#).
- RESPECT, I. *A KAOS Tutorial*. [S.l.]: Objectiver, 2007. Cited 2 times in pages [33](#) and [89](#).

- RUIZ, A. et al. Exoskeletons for rehabilitation and motor control. In: IEEE. *Biomedical Robotics and Biomechatronics, 2006. BioRob 2006. The First IEEE/RAS-EMBS International Conference on*. [S.l.], 2006. p. 601–606. Cited in page 88.
- SANFILIPPO, F. et al. Modgrasp: an open-source rapid-prototyping framework for designing low-cost sensorised modular hands. In: IEEE. *Biomedical Robotics and Biomechatronics (2014 5th IEEE RAS & EMBS International Conference on*. [S.l.], 2014. p. 951–957. Cited in page 88.
- SANTOS, W. M. dos; CAURIN, G. A.; SIQUEIRA, A. A. Torque control characterization of a rotary series elastic actuator for knee rehabilitation. In: IEEE. *Advanced Robotics (ICAR), 2013 16th International Conference on*. [S.l.], 2013. p. 1–6. Cited 2 times in pages 29 and 30.
- SHARIFI, M.; BEHZADIPOUR, S.; VOSSOUGH, G. Nonlinear model reference adaptive impedance control for human–robot interactions. *Control Engineering Practice*, Elsevier, v. 32, p. 9–27, 2014. Cited in page 44.
- SHIER, D.; BUTLER, J.; LEWIS, R. *Hole's Human Anatomy*. [S.l.]: McGraw-Hill, 1996. Cited in page 37.
- SILVA, J. M.; SILVA, J. R. Combining kaos and ghenesys in the requirement and analysis of service manufacturing. *IFAC-PapersOnLine*, Elsevier, v. 48, n. 3, p. 1634–1639, 2015. Cited in page 98.
- SILVA, J. M.; SILVA, J. R. Using goal-oriented requirement analysis in the design of product-service artifacts. In: SBA. *Proc. of the XII Brazilian Symposium in Automation*. [S.l.], 2015. p. 1601–1606. Cited in page 91.
- SLOTINE, J.-J. E.; LI, W. et al. *Applied nonlinear control*. [S.l.]: prentice-Hall Englewood Cliffs, NJ, 1991. v. 199. Cited 3 times in pages 45, 46, and 77.
- SOUIT, C. *Exoesqueleto De Membro Inferior Com Dois Graus De Liberdade Ativos*. Dissertação (Mestrado) — Escola Politécnica da USP, 2016. Cited 3 times in pages 31, 43, and 52.
- SOUIT, C. et al. Design of a lower limb exoskeleton for experimental research on gait control. In: IEEE. *Biomedical Robotics and Biomechatronics (BioRob), 2016 6th IEEE International Conference on*. [S.l.], 2016. p. 1098–1103. Cited 2 times in pages 30 and 52.
- SOUZA, R. S. et al. Modular exoskeleton design: Requirement engineering with kaos. In: IEEE. *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. [S.l.], 2016. p. 978–983. Cited 2 times in pages 32 and 33.
- TEE, K. P. et al. A model of force and impedance in human arm movements. *Biological cybernetics*, Springer, v. 90, n. 5, p. 368–375, 2004. Cited in page 24.
- TÜV Rheinland Issues EC certificate for Cyberdyne's Medical Robot Suit HAL. <https://www.tuv.com/jp/japan/about_us_jp/press_2/news_1/news_contentjp_en_168321.html>. Accessed: 2017-09-30. Cited in page 20.
- WERNECK, V. M. B.; OLIVEIRA, A. d. P. A.; LEITE, J. C. S. do P. Comparing gore frameworks: i-star and kaos. In: WER. [S.l.: s.n.], 2009. Cited in page 89.

WOLFF, J. et al. A survey of stakeholder perspectives on exoskeleton technology. *Journal of neuroengineering and rehabilitation*, BioMed Central Ltd, v. 11, n. 1, p. 169, 2014. Cited 4 times in pages [31](#), [88](#), [93](#), and [99](#).

ZHANG, X.; SHI, Y.; ZHANG, Y. A can-based inertial sensor network for lower limb exoskeleton. In: IEEE. *Communication Problem-Solving (ICCP), 2014 IEEE International Conference on*. [S.l.], 2014. p. 473–476. Cited in page [61](#).

APPENDIX A – MODULAR EXOSKELETON DESIGN: REQUIREMENT ENGINEERING WITH KAOS*

Modular Exoskeleton Design: Requirement Engineering with KAOS*

Rafael S. Souza¹, Filippo Sanfilippo², José R. Silva³
and Arturo F. Cordero^{4*,†,‡,§,¶}

2018

Abstract

Wearable robots, such as exoskeletons, closely interact with the wearer. Mechanical features are combined with control techniques to transparently follow human movements in several degrees of freedom. Such behaviour brings special project requirements, safety being the most critical one. In literature, modular robot design approach provides flexible and yet robust solutions that meets stakeholder requirements.

Semi-formal design approach, Goal Oriented Requirement Engineering (GORE) has been used as a tool for different systems; however, it has been more widely adopted in software than in hardware engineering. GORE is implemented here, with the KAOS method, to fully exploit the integrated design modular exoskeleton - an adaptive mechatronic system.

The balance of requirements with user safety constraints are analysed to advance in the project first steps. It is shown that, with some initial efforts from the designer regarding goals formulation, requirement modelling provides a more comprehensive system overview and documentation. Finally, the semi-formal language justifies why a modular exoskeleton is a good choice when design aims for user experience.

**This work was supported by CNPq and FAPESP (2010/17181-0)

^{†1}Rafael Sanchez Souza is with Department of Mechatronics and Mechanical Systems Engineering, Polytechnic School of University of Sao Paulo, Sao Paulo, Brazil. rafaelsouza@usp.br

^{‡2}Filippo Sanfilippo is with the Department of Engineering Cybernetics, Norwegian University of Science and Technology, 7491 Trondheim, Norway. filippo.sanfilippo@ntnu.no

^{§3}José Reinaldo Silva is with Department of Mechatronics and Mechanical Systems Engineering, Polytechnic School of University of Sao Paulo, Sao Paulo, Brazil. reinaldo@usp.br

^{¶4}Arturo Forner Cordero is with Department of Mechatronics and Mechanical Systems Engineering, Polytechnic School of University of Sao Paulo, Sao Paulo, Brazil. aforner@usp.br

A.1 Introduction

The development of new technologies has led complex and advanced robotic systems, such as exoskeletons, to achieve a high level of performance at reasonable production costs. Therefore, the design and construction of exoskeletons for rehabilitation, functional assistance or to improve the quality of life has become feasible (RUIZ et al., 2006). The research and development of such kind of robots binds technology and science with a common goal: to produce a robust, safe, comfortable and economical healthcare solutions inspired by nature. Nevertheless, unlike many other engineering products, there is no systematic procedure to design exoskeletons.

There are major challenges involved in exoskeleton design. One of the most demanding tasks is that the system must be able to reproduce human movements to a functional level. This means up to 7 degrees of freedom for upper limbs (PERRY; ROSEN, 2006) and up to 12 degrees of freedom for lower limbs (APKARIAN; NAUMANN; CAIRNS, 1989); it must be able to intervene in the exactly desired way (MIRANDA, 2014) in the subject's movement - to control the interaction force between user and robot. Many groups have been researching these technologies and some solutions have been proposed and implemented, such as the exoskeleton ARMin II (MIHELJ; NEF; RIENER, 2007) and the Impedance Control (HOGAN, 1984b).

Yet, systems have functional - as the aforementioned - and non-functional requirements. Although many works discuss, amongst others, wearability, usability and cost (WOLFF et al., 2014), non-functional requirements are still often disregarded during technology development, as also is the case with exoskeleton research. Therefore, the user acceptance represents a key factor decision making along the project development.

In (MIRANDA, 2014), an upper limb exoskeleton prototype was developed. The mechanical design has a particular feature; the segments can be divided into very similar modules formed by a joint and a link. This modular approach, while promising, has not been sufficiently explored in exoskeleton research (MACIEJASZ et al., 2014). Modularity brings development and maintenance costs down due to standardization of components, increasing portability, adaptability (MATSUMARU, 1995) and robustness (FRYER et al., 1997). Rapid prototyping may also be combined to obtain faster results (SANFILIPPO et al., 2014; ??). With modularity, non-functional requirements are also expected to be contemplated.

The Requirement Engineering phase has been frequently skipped and these works started directly in the modelling phase. In this perspective, our main focus is to evaluate the application of Goal Oriented Requirement Engineering (GORE), a goal-oriented approach belonging to the new generation of Requirement Engineering (RE) methods, to a modular exoskeleton design. Some benefits of the goal-oriented approach are: investment in the initial design phase, analysis and specification of changes before costs are higher; possibility

Figure A.1 – KAOS framework adapted from (RESPECT, 2007)

of using different modelling languages; traceability (which could link further changes with resilient requirements), investment in project documentation; the possibility of a better system monitoring and control (LAMSWEEERDE; LETIER, 2004) and a better maintenance process.

There is a gap in the literature concerning formal exoskeleton design. Yet while many works refer to “Exoskeleton Design”, most start from the modelling phase, skipping requirements specification. Although RE has been proven to be very efficient in the design of complex software as well as mechanical devices, it is sometimes criticised for its practical effectiveness as it can be time-consuming, bureaucratic (NUSEIBEH; EASTERBROOK, 2000) and the benefits of using it are hard to measure (LAMSWEEERDE, 2001). In addition, this method has been employed for an integrated design of both software and hardware, but only in few mechatronics design cases.

The goal of this paper is to present the first steps of a systematic exoskeleton design lifecycle and evaluate the method. While some authors have already presented interesting formal modular robotic designs (FRYER et al., 1997; BI; ZHANG, 2001), so far this method has not been applied to exoskeletons. It shows how to depart from high level requirements - goals - formulated in abstract diagrammatic language and to map them to a low level requirements - as technical as possible suited to a modular exoskeleton project. More effort is placed on the requirements phase with the use of the KAOS diagrammatic language (Fig.A.1 which can lead to formal requirements representation, expecting it to be as effective with a mechatronic system as with complex software (WERNECK; OLIVEIRA; LEITE, 2009) or other mechatronic artefacts (JANSCKEK, 2012).

The paper is organised as follows. The methodology is described in Section II. Successively,

the selected system is described in Section III. The requirement modelling phase is outlined in Section IV. A discussion on the proposed method is provided in Section V. Finally, conclusions and future work are outlined in In Section VI.

A.2 Methodology

In this Section, we first shortly introduce important design concepts. Then, the adopted Goal Oriented Requirement Engineering (GORE) tool is described. Successively, the KAOS method is presented in its fundamentals. Systems Design has become the basis for designing any device, integrating product-centred approach and service or process-approach in what is called a systemic view. Therefore, any device intended to have an operational coupling with a final user (a characteristic from “services”) has to face a design process in which requirements are modelled and analysed beforehand. If hardware and software have to be integrated, modern approaches based on Model Driven Engineering and SysML modelling language could be used to combine static object-oriented design with process design. This process could be based on Petri Nets or other schematic formal representation. A complex product such as an exoskeleton should be considered a cluster of sub-systems responsible for specific behaviour which contributes to an overall goal.

User and exoskeleton are understood herein as a system composed of a set of agents with a collective goal extracted and analysed in the requirements phase. For this, we propose a design lifecycle which is more concerned with single phases than with rationales and superposition as preconised by Rational Unified Process ([KRUCHTEN, 2004](#));

- High-quality requirements elicitation
- Requirement modelling and analysis with traceability
- Solving conflicts and ambiguous requirements
- Decision process (for solutions) and Rationale documentation
- Design validation and verification

A.2.1 Goal Oriented Requirement Engineering

In the literature, exoskeleton design usually starts with the modelling phase which is a normal application domain familiar to the design. As a consequence, the requirement phase is reduced, which could have a negative impact on the integration between system and final user.

Classic requirement analysis procedures have to cope with the dichotomy between functional and non-functional requirements, which makes the process less intuitive. In a

goal-oriented approach, we do not have the same problem since goals already encompass all the necessary to achieve them including non-functional conditions. “A goal is an objective the system under consideration should achieve” (LAMSWEEERDE, 2001). Goals are the fundamental stone on which the method elements such as objects, agents, events etc., will be supported.

- Goals de precise criterion for completeness of a requirement specification
- Goals provide precise criterion for rem Goals provide the are providedrationale for requirements
- Goal refinement is a natural mechanism for structuring complex requirements documents

A.2.2 KAOS

KAOS is a representation schema to implement GORE (HEAVEN; FINKELSTEIN, 2004) which is based on visual diagrams, which could also be transformed - once requirements compose a stable model - in a formal representation in LTL (Linear Tree Logic) or in Petri Nets (SILVA; SILVA, 2015b). The graphical user interface for KAOS implementation used herein was the Objectiver software, developed by Respect-IT. The KAOS framework is based on four linked diagrams that provide both designer and stakeholder a wider system overview of goals and operationalization. These four diagrams are briefly explained below (see Fig.A.1):

1. Goals Diagram. The fundamental model for KAOS, in which the project goals are presented. Goals are linked to subgoals through refinement. The Goal Avoid[TissueDamageByExce may be refined to the Requirement Main- tain[SafeRangeOfMotion] which is constrained by the Domain Property “Human Joint Safe Range of Motion” and again refined into the Requirement Maintain[AccurateJointAngleMeasure].
2. Object Model. Admits the identification of objects, such as entity, relationship, event or agent. Defines the concepts of the application domain and the system constraints. Allows establishing the object characteristics as attributes. Exoskeleton Joint and Human Joint relation is defined as “Attached”.
3. Agent Responsibility Model. Defines the responsibilities between agents and requirements. The Controller and the Position Sensor are respectively responsible for maintaining a safe range of motion and an accurate joint angle measure.
4. Operation Model. Shows how the objects work together to achieve the system requirements by defining the state transitions. If angles are off limits, the Controller shall bring the Exoskeleton Joint back to a safe position.

Figure A.2 – The underlying idea of applying Goal Oriented Requirement Engineering (GORE), a software design tool borrowed from software engineering, for the integrated hardware and software design of a modular exoskeleton.

A.3 System Description

A typical Systems Design approach considers the known system the system-as-is, that is, the legacy system from where we take knowledge. As represented by Fig.A.2, the device proposed - the modular exoskeleton Mod-Exo - is treated as the system-to-be. We took the EXO-C (MIRANDA et al., 2012) prototype as the legacy system.

In (MIRANDA, 2014)], a mechanical structure of an exoskeleton was built which can be divided into modules and it's referred here as Mod-Exo. Modularity is a property that can also be applied to both software and hardware and KAOS is expected to help this integrated design.

The Mod-Exo actuation is provided by a Maxon Motor EC32, 80 Watt with a 100:1 harmonic reduction and by the EPOS2 24/5 driver. The interaction force between the arm and the exoskeleton is obtained by means of a Wheatstone bridge of 350 ohm strain gauges glued to the exoskeleton aluminium structure linked to the forearm. The position is measured by a rotational potentiometer 2 Kohm with a linearity of 2 percent coupled to the elbow joint. There are two levels of control, a lower level one provided by the driver EPOS2 24/5 from Maxon and a higher level implemented in an embedded computer PC104. There are different high level controllers that can be easily implemented. So far, a force-follower and an impedance controller have been implemented. Human arm-exoskeleton coupling occurs with an adapted shoulder orthosis Ottobock Omo-Neurexa (Otto Bock gmBh, Germany).

A.4 Requirement Modelling

Requirement Modelling occurs immediately after a primary goal specification by the stakeholder and before the system modelling and analysis. The underlying idea is to develop a diagram containing the project constraints, which should be as complete, precise and unambiguous as possible. The goal is to improve the quality of the information provided by the stakeholder and translate it into tangible information for the engineer. The target is to prevent the engineer from working on requirements that will not fit the stakeholder's needs and to reduce the design iterations.

This section shows a brief implementation of the method. The steps followed were fully described in (LETIER et al., 2001). Note that this is not done once only, but continuously, along the project development.

A.4.1 Identifying Preliminary Goals and Refinement

Preliminary goals were retrieved from the literature and from the Biomechatronics Laboratory previous works. They were considered to elaborate the three sub-projects main diagrams, represented in figures A.3, A.4, A.5; the diagrams are quite self-explanatory, which makes them useful for documentation, as follows:

1. The system should intervene in user movement. This goal concerns the control of the exoskeleton, that is, the inputs and outputs for interfering in the user's movement. Also, which human joints and how many degrees of freedom are required;
2. The system must be commercially feasible. This goal targets an exoskeleton that goes beyond the academic environment. It gathers economical, environmental challenges and user's acceptance - some taken from (WOLFF et al., 2014), such as comfort, costs and ease of use;
3. User safety. A primary concern of any wearable robot;
4. The system must be a modular robot. There is strong evidence that this property may be a key to satisfy the system requirements and the goal is here treated as a hypothesis. Modular preliminary subgoals were identified from (MATSUMARU, 1995).

A.4.2 Formalizing Goals and Identifying Objects

As a case study for KAOS implementation, the next steps are followed towards the goal Maintain User Safety, a primary concern of any exoskeleton. The Goals are defined, formally defined and then derived to objects.

Figure A.3 – Mod-Exo: User's Movement Intervention Goals

Figure A.4 – Mod-Exo: Commercial Feasibility Goals

Considering the subgoals Avoid [Muscle Bone Stress] and Avoid[Joint Hyper Extension Flexion]: both are simultaneously necessary for the goal Maintain [User Safety]. This is represented by a circle as an AND refinement in Fig.A.6. These subgoals can be refined into requirements, represented by the thick border parallelogram Maintain[Joint Torque Limit] and Maintain[Joint Angle Limit]. If the human joint torque limit is respected, the exoskeleton will not harm the user. Evidently, the same will happen if the human joint angles ranges of motion are respected. This is represented by the OR refinement at Maintain[Join Angle Limit] in Fig.A.6. Now goals are defined differently, in a formal language, which helps to identify project parameters.

Goal Maintain[SafeRelativeTorque]

Definition The Exoskeleton Joint torque should stay below the maximum torque the human joint can handle.

FormalDef

$$\forall ej : ExoJoint, hj : HumanJoint \text{ Attached}(ej, hj) \Rightarrow ej.Torque \leq hj.SafeTorque. \quad (A.1)$$

Goal Maintain[SafeRangeOfMotion]

Definition The Exoskeleton Joint angle should stay between the maximum and minimum angles the human joint can handle.

FormalDef

$$\forall ej : ExoJoint, hj : HumanJoint \text{ Attached}(ej, hj) \Rightarrow hj.AngleInfLimit \leq ej.Angle \leq hj.AngleSupLimit. \quad (A.2)$$

From the formal definition, objects and attributes can be identified as seen in Fig. A.6: ExoJoint and HumanJoint are the objects corresponding to the exoskeleton module and the user arm; “Attached” defines the relationship between them; ej.Torque and hj.SafeTorque are declared as their respective attributes.

A.4.3 Eliciting Further Through WHY and HOW Questions

WHY and HOW questions are asked at this step. WHY identifies higher-level parent goals, which provide the rationale for the project documentation. HOW refines a goal into subgoals until they can be assigned to agents. By asking HOW in Maintain[SafeRangeOfMotion], a redundant security solution is also defined: in the case of a software failure, the power supply should be switched off.

Goal Maintain[SystemShutdownWhenAngleLimit]

Definition Electrically constraints operational angles with sensors attached directly to power source.

Figure A.5 – Mod-Exo: Modularity Goals

FormalDef

$$\begin{aligned}
 &\forall ej : ExoJoint, hj : HumanJoint : \\
 &(hj.AngleInfLimit \leq ej.Angle \leq hj.AngleSupLimit) \wedge \\
 &Attached(ej, hj) \Rightarrow ej.Power = "Off".
 \end{aligned} \tag{A.3}$$

By asking the Modular Robot Goal WHY, the higher-level goals: Robust System, Parts Standardization, Minimise Wearing Effort, Portability, Adaptive Degrees of Freedom, Minimise Costs, Collaborative Development were identified, completing the diagram shown in Fig. A.5.

In Fig. A.6, “Human Joint Safe Working Torque” and “Human Joint Safe Range of Motion”, inside the pentagons are Domain Properties. These are properties related to the environment such as laws of nature.

A.4.4 Identifying potential responsibility assignments

The agents identified through the previous steps were: Load Cell and Position Sensor, responsible for monitoring the system; the Controller, which centralises information, processes it and outputs commands; and the Micro Switch, an electric switch that should be positioned according to range of motion requirements. There are goals directly related to these agents:

Goal Maintain[AccurateJointAngleMeasure]

Figure A.6 – Safety Goals: Eliciting Further

Definition The angle measure should equal the actual angle of the exoskeleton joint.

FormalDef

$$\begin{aligned}
 &\forall ej : ExoJoint, hj : HumanJoint, c : Controller \\
 &\quad CtrlJoint(c, ej) \wedge Attached(ej, hj) \Rightarrow \\
 &\quad c.AngleMeasure = ej.Angle.
 \end{aligned}
 \tag{A.4}$$

Goal Maintain[AccurateRelativeTorqueMeasure]

Definition The torque measure should equal the actual torque between user and exoskeleton.

FormalDef

$$\begin{aligned}
 &\forall ej : ExoJoint, hj : HumanJoint, c : Controller \\
 &\quad CtrlJoint(c, ej) \wedge Attached(ej, hj) \Rightarrow \\
 &\quad c.TorqueMeasure = ej.Torque.
 \end{aligned}
 \tag{A.5}$$

A.5 Discussion

We have applied a formal and structured design method to define the requirements of an exoskeleton. One of the questions was if the use of this approach would really be advantageous, considering the concerns about its effectiveness. As the goals were developed and the definitions made, the use of the method and its implementation in KAOS language

became clearer. The learning curve was slightly steep, since there are several aspects to master before a proper model is achieved. The initial effort was placed on the formulation and refinement of the goals, once a bad goal definition may lead to a dead end. Due to the uncommon application of the KAOS method - an mechatronic system - there are few references in literature to rely on when struggling with a procedure concept: sensors, for example, in this paper were defined as agents but they can also be entities.

According to (AL-SUBAIE; MAIBAUM, 2006), the KAOS method evaluation should take Requirement Engineering Objectives, such as Pertinence, Correctness, Traceability and Understandability, into consideration.

The models for each defined goal came out differently. The “Maintain User Safety” model was considered the most successful; with the refinements, it was possible to reach tangible project requirements departing from an abstract “Maintain User Safety” and even think of safety solutions, such as the micro switch, conceived during the development of the models. “Intervention in User Movement” was also considered quite successful, but it still needs developing. We believe that the difference in complexity of each goal is directly related to the model effectiveness. This was verified in the two models, where the first is simpler than the second.

Although not so straightforward, both “Commercially Feasible” and “Modular Robot” goals provided some interesting results for the exoskeleton designer. In the beginning of the study, the modularity goal was stated as a possible solution to meet the exoskeleton design requirements. These are observed to be met, when asking HOW to commercial feasibility and WHY to modularity. Also, the complexity of the diagrams indicates the challenge of producing a commercially feasible exoskeleton: one that besides solving control and kinematic challenges, meets user’s needs.

The models highlight important project parameters such as “Human Joint Safe Working Torque” and “Human Joint Safe Range of Motion” which must be investigated before the exoskeleton prototype is built. The bolts linking “Minimise Costs” and “Minimise Environmental Impact” and “Maintain Comfort” in Fig.A.4 indicate conflicts between the goals. The designer has to reflect upon these indicatives given by the models.

Although connected to requirement elicitation, which is not formal, requirements analysis using the goal-oriented approach and the KAOS language resulted in a formal description in LTL (Linear Tree Logic) or in Petri Nets to cover the dynamics of the system (SILVA; SILVA, 2015a).

A.6 Conclusion

Using the goal-oriented approach for requirement engineering enriches the early phase of exoskeleton design because it highlights constraints, inserts traceability into the process, and can also result in a formal specification, the first goal in the classic process.

To the best of our knowledge, there are a small number of works seeking to generate exoskeleton requirements. Some works, such as (WOLFF et al., 2014; PATOGLU et al., 2010) raise requirements by means of a survey. They are complementary to the work presented here: KAOS provides tools to evaluate those requirements pertinence and to further develop those requirements.

With respect to the modelling, refinement time grows exponentially in order to cover all the system features. It is strongly suggested that this method is used objectively and according to predefined milestones. Once a certain requirement modelling time is reached, the designer should move forward to the next design phases and update the models along the project development.

The next steps should be to advance to the next design phases, which comprehend modelling and implementing the solutions that meet the requirements raised by the KAOS method. Other research groups are encouraged to try the method and to improve the models presented herein.

APPENDIX B – MODEXO PORTABLE CASE DETAILED DESIGN

Detailed pictures of the designed portable case are shown. The CAD Design was made by Priscila Veiga, the conception of the portable case was the result of a collaborative work.

Figure B.1 – Perspective view of the portable case CAD

Figure B.2 – Wooden structure of the portable case, built by Priscila Veiga.

Figure B.3 – The motor height imposes a minimal height for the ModExo case.

Figure B.4 – The parts corresponding to the ventilation and the USB connectors support were 3D printed.

Figure B.5 – It's possible to connect to the Arduino and the EPOS with USB through this connectors.

Figure B.6 – Front view of the portable case

Figure B.7 – Front view of the portable case CAD

Figure B.8 – Top view of the portable case

Figure B.9 – Top view of the portable case CAD

APPENDIX C – CONTROL HARDWARE SCHEMATICS

Schematics of the control hardware communication are shown. The CAD Design was made in collaboration with Gabriel Riqueti.

Sheet: /High_Level_Layer/
File: high_level_layer.sch

Title: High Level Layer Communication

Size: A4 Date:
KiCad E.D.A. eeschema 4.0.4-stable

Rev:
Id: 2/3

ANNEX A – EPOS2 DRIVER CATALOG

EPOS2 24/5

Matched with DC brush motors with encoder or brushless EC motors with Hall sensors and encoder, from 5 to 120 watts.

EPOS2 50/5

Matched with DC brush motors with encoder or brushless EC motors with Hall sensors and encoder, from 5 to 250 watts.

EPOS2 70/10

Matched with DC brush motors with encoder or brushless EC motors with Hall sensors or encoder, from 80 to 700 watts.

Controller versions		
CANopen Slave	CANopen Slave	CANopen Slave
Electrical data		
11 - 24 VDC	11 - 50 VDC	11 - 70 VDC
11 - 24 VDC	11 - 50 VDC	11 - 70 VDC
0.9 x V _{CC}	0.9 x V _{CC}	0.9 x V _{CC}
10 A	10 A	25 A
5 A	5 A	10 A
50 kHz	50 kHz	50 kHz
10 kHz	10 kHz	10 kHz
1 kHz	1 kHz	1 kHz
1 kHz	1 kHz	1 kHz
25 000 rpm (sinusoidal); 100 000 rpm (block)	25 000 rpm (sinusoidal); 100 000 rpm (block)	25 000 rpm (sinusoidal); 100 000 rpm (block)
15 µH / 5 A	22 µH / 5 A	25 µH / 10 A
Input		
H1, H2, H3	H1, H2, H3	H1, H2, H3
A, A', B, B', I, I' (max. 5 MHz)	A, A', B, B', I, I' (max. 5 MHz)	A, A', B, B', I, I' (max. 5 MHz)
6 (TTL and PLC level)	11 (7 optically isolated, 4 differential)	10 (7 optically isolated, 3 differential)
2 (12-bit resolution, 0...+5 V)	2 (differential, 12-bit resolution, ±10 V)	2 (differential, 12-bit resolution, 0...+5 V)
configurable with DIP switch 1...7	configurable with DIP switch 1...7	configurable with DIP switch 1...7
Output		
4	5 (4 optically isolated, 1 differential)	5 (4 optically isolated, 1 differential)
	1 (12-bit, 0...10 V)	
+5 VDC, max. 100 mA	+5 VDC, max. 100 mA	+5 VDC, max. 100 mA
+5 VDC, max. 30 mA	+5 VDC, max. 30 mA	+5 VDC, max. 30 mA
V _{CC} , max. 1300 mA	+5 VDC, max. 150 mA	+5 VDC, max. 150 mA; +5 VDC (R _i = 1 kΩ)
Interface		
RxD; TxD (max. 115 200 bit/s)	RxD; TxD (max. 115 200 bit/s)	RxD; TxD (max. 115 200 bit/s)
high; low (max. 1 Mbit/s)	high; low (max. 1 Mbit/s)	high; low (max. 1 Mbit/s)
Data+; Data- (max. 12 Mbit/s)	Data+; Data- (max. 12 Mbit/s)	Data+; Data- (max. 12 Mbit/s)
Indicator		
green LED, red LED	green LED, red LED	green LED, red LED
Environmental conditions		
-10...+55°C	-10...+45°C	-10...+45°C
+55...+83°C; Derating: -0.179 A/°C	+45...+80°C; Derating: -0.143 A/°C	+55...+85°C; Derating: -0.250 A/°C
-40...+85°C	-40...+85°C	-40...+85°C
5...90%	5...90%	5...90%
Mechanical data		
Approx. 170 g	Approx. 240 g	Approx. 330 g
105 x 83 x 24 mm	120 x 93.5 x 27 mm	150 x 93 x 27 mm
Flange for M3-screws	Flange for M3-screws	Flange for M3-screws
Part numbers		
367676 EPOS2 24/5	347717 EPOS2 50/5	375711 EPOS2 70/10
Accessories		
309687 DSR 50/5 Shunt regulator	309687 DSR 50/5 Shunt regulator	235811 DSR 70/30 Shunt regulator
Order accessories separately, see page 437	Order accessories separately, see page 437	Order accessories separately, see page 437

ANNEX B – MOTOR EC90 FLAT DRIVER CATALOG

EC 90 flat Ø90 mm, brushless, 90 Watt

maxon flat motor

M 1:2

- Stock program
- Standard program
- Special program (on request)

Part Numbers

with Hall sensors

323772 429271 244879

Motor Data

Values at nominal voltage		24	36	48
1 Nominal voltage	V	24	36	48
2 No load speed	rpm	3190	3120	2080
3 No load current	mA	544	348	135
4 Nominal speed	rpm	2590	2510	1610
5 Nominal torque (max. continuous torque)	mNm	444	560	533
6 Nominal current (max. continuous current)	A	6.06	4.76	2.27
7 Stall torque	mNm	4940	7480	4570
8 Stall current	A	70	69	21.1
9 Max. efficiency	%	84	87	85
Characteristics				
10 Terminal resistance phase to phase	Ω	0.343	0.522	2.28
11 Terminal inductance phase to phase	mH	0.264	0.625	2.5
12 Torque constant	mNm/A	70.5	109	217
13 Speed constant	rpm/V	135	88	44
14 Speed/torque gradient	rpm/mNm	0.659	0.423	0.462
15 Mechanical time constant	ms	21.1	13.6	14.8
16 Rotor inertia	gcm ²	3060	3060	3060

Specifications

Thermal data		
17 Thermal resistance housing-ambient	1.91 K/W	
18 Thermal resistance winding-housing	2.6 K/W	
19 Thermal time constant winding	46 s	
20 Thermal time constant motor	283 s	
21 Ambient temperature	-40...+100°C	
22 Max. winding temperature	+125°C	
Mechanical data (preloaded ball bearings)		
23 Max. speed	5000 rpm	
24 Axial play at axial load	< 15 N	0 mm
	> 15 N	0.14 mm
25 Radial play	preloaded	12 N
26 Max. axial load (dynamic)		183 N
27 Max. force for press fits (static)		8000 N
28 Max. radial load, 5 mm from flange		68 N

Other specifications

29 Number of pole pairs	12
30 Number of phases	3
31 Weight of motor	600 g

Values listed in the table are nominal.

Connection

Pin 1	Hall sensor 1
Pin 2	Hall sensor 2
Pin 3	V _{Hall} 4.5...18 VDC
Pin 4	Motor winding 3
Pin 5	Hall sensor 3
Pin 6	GND
Pin 7	Motor winding 1
Pin 8	Motor winding 2

Wiring diagram for Hall sensors see p. 37

Cable

Connection cable Universal, L = 500 mm	339380
Connection cable to EPOS2, L = 500 mm	354045

Operating Range

Comments

■ **Continuous operation**
In observation of above listed thermal resistance (lines 17 and 18) the maximum permissible winding temperature will be reached during continuous operation at 25°C ambient.
= Thermal limit.

■ **Short term operation**
The motor may be briefly overloaded (recurring).

— Assigned power rating

maxon Modular System

Overview on page 20-27

Planetary Gearhead
Ø52 mm
4 - 30 Nm
Page 355

Recommended Electronics:	
Notes	Page 26
ESCON Mod. 50/4 EC-S	417
ESCON Mod. 50/5	417
ESCON 50/5	418
ESCON 70/10	418
DEC Module 50/5	420
EPOS2 24/5, 50/5, 70/10	425
EPOS2 P 24/5	428
EPOS4 Module 50/8	431
EPOS4 Comp. 50/8 CAN	431
MAXPOS 50/5	435

Encoder MILE
512 - 6400 CPT,
2 channels
Page 381

ANNEX C – ENCODER MILE CATALOG

Integrated into motor

Direction of rotation cw (definition cw p. 150)

Part Numbers

453234	409996	453233	411964	453232	411965	453231	411966
--------	--------	--------	--------	--------	--------	--------	--------

[illegible]

+ Motor	Page	+ Gearhead	Page	+ Brake	Page	Overall length [mm] / ● see Gearhead							
EC 90 flat	305					29.2	29.2	29.2	29.2	29.2	29.2	29.2	29.2
EC 90 flat	305	GP 52. 4 - 30 Nm	355			●	●	●	●	●	●	●	●

[illegible]

Connection example

Connection motor	Connection Encoder
Pin 1 Hall sensor 1*	Pin 1 N.C.
Pin 2 Hall sensor 2*	Pin 2 V _{CC}
Pin 3 V _{Hall} 4.5...18 VDC	Pin 3 GND
Pin 4 Motor winding 3	Pin 4 N.C.
Pin 5 Hall sensor 3*	Pin 5 Channel A
Pin 6 GND	Pin 6 Channel A
Pin 7 Motor winding 1	Pin 7 Channel B
Pin 8 Motor winding 2	Pin 8 Channel B
	Pin 9 Do not connect
	Pin 10 Do not connect

*Internal pull-up (10 kΩ) on pin 3

Encoder, Line Driver, AM26C31QD

V_{CC}

GND

Channel $\bar{A}$

Channel A

Channel $\bar{B}$

Channel B

R

R

Line receiver
Recommended IC's:

- MC 3486
- SN 75175
- AM 26 LS 32

maxon sensor 381

ANNEX D – MA3 ENCODER CATALOG

Description

The MA3 is a miniature rotary absolute shaft encoder that reports the shaft position over 360 ° with no stops or gaps. The MA3 is available with an analog or a pulse width modulated (PWM) digital output.

Analog output provides an analog voltage that is proportional to the absolute shaft position. Analog output is only available in 10-bit resolution.

PWM output provides a pulse width duty cycle that is proportional to the absolute shaft position. PWM output is available in 10-bit and 12-bit resolutions. While the accuracy is the same for both encoders, the 12-bit version provides higher resolution.

Three shaft torque versions are available. The standard torque version has a sleeve bushing lubricated with a viscous motion control gel to provide torque and feel that is ideal for front panel human interface applications.

The no torque added option has a sleeve bushing and a low viscosity lubricant (that does not intentionally add torque) for low RPM applications where a small amount of torque is acceptable.

The ball bearing version uses miniature precision ball bearings that are suitable for high speed and ultra low torque applications. The shaft diameter for ball bearing version option is 1/8" rather than 1/4".

Connecting to the MA3 is simple. The 3-pin high retention snap-in 1.25mm pitch polarized connector provides for +5V, output, and ground.

Features

- ▶ Miniature size (0.48" diameter)
- ▶ Non-contacting magnetic single chip sensing technology
- ▶ -40C to 125C operating temperature range
- ▶ 10-bit Analog output - 2.6 kHz sampling rate
- ▶ 10-bit PWM output - 1024 positions per revolution, 1 kHz
- ▶ 12-bit PWM output - 4096 positions per revolution, 250 Hz

Mechanical Drawing

MA3 Miniature Absolute Magnetic Shaft Encoder

US DIGITAL 1400 NE 136th Avenue
Vancouver, Washington 98684, USA

info@usdigital.com
www.usdigital.com

Local: 360.260.2468
Toll-free: 800.736.0194

UNITS: INCHES (MM)
METRIC SHOWN FOR REFERENCE ONLY

Environmental

Parameter	Value
Operating Temperature	-40C to +125C
Humidity, Non-condensing	5% to 85%
Vibration (5Hz to 2kHz)	20 G.
Electrostatic Discharge, Human Body Model MIL-STD-883E, Method 3015.7	± 2 kV

Mechanical

Specification	Sleeve Bushing	Ball Bearing
Moment of Inertia	4.1×10^{-6} oz-in-s ²	4.1×10^{-6} oz-in-s ²
Max. Shaft Speed (1)	100 rpm	15000 rpm
Max. Acceleration	10000 rad/sec ²	250000 rad/sec ²
Max. Shaft Torque	0.5 ± 0.2 in-oz (D - torque option) 0.3 in-oz (N- torque option)	0.05 in-oz
Max. Shaft Loading	2 lb. dynamic 20 lb. static	1 lb.

Specification	Sleeve Bushing	Ball Bearing
Bearing Life (2)	> 1,000,000 revolutions	$L_{10} = (18.3/F_r)^3$ Where L_{10} = bearing life in millions of revs, and F_r = radial shaft loading in pounds
Weight	0.46 oz.	0.37 oz.
Max. Shaft Total Indicated Runout	0.0015 in.	0.0015 in.
Technical Bulletin TB1001 - Shaft and Bore Tolerances		Download

(1) When a pulley, gear, or friction wheel drives the shaft, the Ball Bearing option is recommended instead of the Sleeve Bushing. The chip that decodes position uses sampled data. There will be fewer readings per revolution as the speed increases. The formula for number of readings per revolution is given by:

10-bit PWM:

$$n = 625200 / \text{rpm}$$

12-bit PWM / Analog:

$$n = 156600 / \text{rpm}$$

(2) only valid with negligible axial shaft loading

Mounting

Parameter	Value
Hole Diameter	0.375" +0.005 / -0.0
Panel Thickness	0.125" max.
Panel Nut Max. Torque	20 in.-lbs.

Materials

Component	Material	Torque Option(s)
Shaft	Stainless	Sleeve Bushing (-D and -N options)
	Brass	Ball Bearing (-B option only)
Bushing	Brass	-

Magnetic Field Crosstalk

The MA3 absolute encoder contains a small internal magnet, mounted on the end of the shaft that generates a weak magnetic field extending outside the housing of each encoder. If two MA3 units are to be installed closer than 1 inch apart (measured between the center of both shafts), a magnetic shield, such as a small steel plate should be installed in between to prevent one encoder from causing small changes in reported position through magnetic field cross-talk.

Electrical

Parameter	Min.	Typ.	Max.	Units
Power Supply	4.5	5.0	5.5	Volts
Supply Current	-	16	20	mA
Power-up Time	-	-	50	mS

Analog Output Operation

Analog output is only available in 10-bit resolution. The analog output voltage is ratiometric to the power supply voltage and will typically swing within 15 millivolts of the power supply rails with no output load. This non-linearity near the rails increases with increasing output loads. For this reason, the output load impedance should be $\geq 4.7k\Omega$ and less than 100pF. The graphs below show the typical output levels for various output loads when powered by a 5V supply.

Parameter	Min.	Typ.	Max.	Units
Position Sampling Rate	2.35	2.61	2.87	kHz
Propagation Delay	-	-	384	?S
Analog Output Voltage Maximum (1)	-	4.987	-	Volts
Analog Output Voltage Minimum (1)	-	0.015	-	Volts
Output Short Circuit Sink Current (2)	-	32	50	mA
Output Short Circuit Source Current (2)	-	36	66	mA
Output Noise (2)	160	220	490	μ Vrms
Output Transition Noise (3)	-	0.03	-	Deg. RMS

(1) With no output load. See graphs below.

(2) Continuous short to +5V or ground will not damage the MA3.

(3) Transition noise is the jitter in the transition between two adjacent position steps.

PWM Output Operation

The magnetic sensor chip in the MA3 has an on-chip RC oscillator which is factory trimmed to 5% accuracy at room temperature (10% over full temperature range). This tolerance influences the sampling rate and pulse period of the PWM output. If only the PWM pulse width t_{on} and the nominal pulse period is used to measure the angle, the resulting value also has this timing tolerance. However, this tolerance can be cancelled by measuring both t_{on} and t_{off} and calculating the angle from the duty cycle.

Parameter	Min.	Typ.	Max.	Units
PWM Frequency (-40C to 125C)				
10-bit	0.877	0.975	1.072	kHz
12-bit	220	244	268	Hz
Minimum Pulse Width				
10-bit	0.95	1.00	1.05	µS
12-bit	0.95	1.00	1.05	µS
Maximum Pulse Width				
10-bit	974	1025	1076	µS
12-bit	3892	4097	4302	µS
Internal Sampling Rate				
10-bit	9.38	10.42	11.46	kHz
12-bit	2.35	2.61	2.87	kHz

Parameter	Min.	Typ.	Max.	Units
Propagation				
10-bit	-	-	48	?S
12-bit	-	-	384	?S
Output Transition Noise, 12-bit version (1)		0.03		Deg. RMS
Output Transition Noise, 10-bit version (1)		0.12		Deg. RMS
Output High Voltage (V OH: @4mA Source) (2)	V _{CC} -0.5	-	-	V
Output Low Voltage (V OL: @4mA Sink) (2)	-	-	0.4	V

(1) Transition noise is the jitter in the transition between two adjacent position steps.

(2) Continuous short to +5V or ground will not damage the MA3.

10-bit PWM:

$$x = ((t_{on} * 1026) / (t_{on} + t_{off})) - 1$$

If $x \leq 1022$, then Position = x

If $x = 1024$, then Position = 1023

12-bit PWM:

$$x = ((t_{on} * 4098) / (t_{on} + t_{off})) - 1$$

If $x \leq 4094$, then Position = x

If $x = 4096$, then Position = 4095

Pin-outs

Analog Output (MA3-A):

Pin	Name	Description
1	5	+5VDC power
2	A	Analog output
3	G	Ground

PWM Output (MA3-P10, MA3-P12):

Pin	Name	Description
-----	------	-------------

1	5	+5VDC power
2	A	PWM output
3	G	Ground

Ordering Information

MA3 -

-

-

Interface

A10 = 10-Bit
Analog

P10 = 10-Bit
PWM

P12 = 12-Bit
PWM

Shaft Diameter

125 = 1/8"

236 = 6mm

250 = 1/4"

Torque

D = Sleeve Bushing,
Most Drag

N = Sleeve Bushing,
Somewhat Lighter Drag

B = Ball Bearing, Free
Spinning (Least Drag)

Notes

- Cables and connectors are not included and must be ordered separately.
- US Digital warrants its products against defects in materials and workmanship for two years. See complete warranty for details.