U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Hybrid Legged-Wheeled Locomotion System for Robotic Vehicles

Vítor Hugo Machado Oliveira Pinto

Supervisor: Prof. Paulo José Cerqueira Gomes da Costa

Co-Supervisor: Prof. José Alexandre de Carvalho Gonçalves

Doctoral Program in Electrical and Computer Engineering,

Department of Electrical and Computer Engineering

Faculty of Engineering, University of Porto

28th October, 2021

Abstract

Over time, society has seen the increasing use of robotic solutions that help humans performing multiple difficult tasks. This growth also reflects in the number of research projects in the autonomous robotic vehicles field. Moreover, the use of robots in collaborative and highly dynamic environments has been growing over the past few years. The use of non-rigid robotic joints can have a significant impact on the safety increase in the interaction between humans and robots, while increasing the robot's tolerance to impacts. However, it also increases the complexity of the system, requiring the use of more advanced control approaches. Hybrid legged-wheeled robots can be presented as a solution, where the advantages of versatile legged locomotion is combined with a faster navigation provided by the wheels.

This thesis presents a novel design for a non-rigid robotic rotary joint. Apart from the driving motor, this joint was designed using only passive components. It was decided to use the Series Elastic Actuator (SEA) approach, considering autonomous mobile robots power consumption restrictions. Moreover, the linear and non-linear models of the non-rigid joint were derived and estimated. The linear (or linearized) models of each core element of the joint were used to design a state-space control approach. The model including the non-linear elements was used in a simulator with incorporated physics engine. Several approaches to control the joint angular position were designed and tested. Specifically, classic position PID, compliant and state-space approaches were proposed.

These joints were applied to a custom design of an innovative hybrid legged-wheeled locomotion system. Targeted for scientific research, the platform design process considered the premises of fast prototyping, modularity and expandability. A quadruped prototype was built to serve as a testbed. The robot can use several locomotion configurations, namely purely-legged, purely-wheeled and hybrid. It incorporates a *low-power*, purely-wheeled mode, in which energy consumption is reduced. This is achieved through the use of an omni-directional wheel, strategically placed in the *foot*. The implemented kinematics for the single leg, as well as the robot are detailed throughout the respective sections. Each leg uses a smooth trajectory following strategy, to position the wheel within its workspace. The robot can follow line or curve paths, respecting a predefined orientation. The results of the performed tests are detailed in the respective section.

Resumo

Ao longo do tempo, a sociedade tem assistido ao uso crescente de soluções robóticas que ajudam os seres humanos a realizar múltiplas tarefas que, para nós, são difíceis. Este crescimento também se reflete no número de projetos de investigação no domínio dos veículos robóticos autónomos. Para além disso, a utilização de robôs em ambientes colaborativos e altamente dinâmicos tem também vindo a crescer ao longo dos últimos anos. A utilização de juntas robóticas não rígidas pode ter um impacto significativo no aumento da segurança na interacção entre humanos e robôs, ao mesmo tempo que aumenta a tolerância do próprio robô a impactos. No entanto, também aumenta a complexidade do sistema, nomeadamente na utilização de abordagens de controlo mais avançadas. Os robôs de locomoção híbrida pata-roda podem ser apresentados como uma solução viável, onde as vantagens da versatilidade da locomoção com patas são combinadas com a navegação mais rápida proporcionada pelas rodas.

Esta tese apresenta um *design* novo para uma articulação robótica não-rígida rotativa. Para além do motor, esta junta foi concebida utilizando apenas componentes passivos. Decidiu-se utilizar a abordagem *Series Elastic Actuator* (SEA), considerando as restrições de consumo de energia dos robôs móveis autónomos. Os modelos lineares e não lineares da junta não-rígida foram derivados e estimados. Os modelos lineares (ou linearizados) de cada elemento base da junta foram utilizados para conceber uma abordagem de controlo de espaço de estados. O modelo que inclui os elementos não lineares foi utilizado num simulador com motor de física incorporado. Foram desenvolvidas e testadas múltiplas abordagens de controlo para a posição angular da articulação. Mais especificamente, uma abordagem clássica utilizando um *PID* de posição, uma abordagem *compliant* e também uma abordagem utilizando controlo em espaço de estados.

Estas juntas foram aplicadas no projeto de um sistema locomoção híbrido inovador, baseado em patas-rodas. Direcionado para a investigação científica, o processo de concepção da plataforma teve em conta as premissas de prototipagem rápida, modularidade e expansibilidade. Foi construído um quadrúpede como protótipo, para servir de plataforma de testes. O robô pode utilizar várias configurações de locomoção, nomeadamente utilizando apenas as pernas, apenas as rodas e híbrida. Incorpora um modo *low-power*, em que utiliza apenas as rodas, no qual o seu consumo de energia

é reduzido. Isto é conseguido através da utilização de uma roda omnidireccional estrategicamente colocada no *pé*. As cinemáticas implementadas para a perna, bem como para o robô, são detalhadas ao longo das respectivas secções. Cada perna utiliza uma estratégia com trajectórias suaves para posicionar a roda dentro do seu espaço de trabalho. O robô consegue seguir caminhos definidos por linhas ou curvas, respeitando uma orientação pré-definida. Os resultados dos testes realizados são detalhados em cada secção respetiva.

Agradecimentos

São diversas as pessoas que merecem o meu agradecimento.

Em primeiro lugar, ao meu orientador e acima de tudo, mentor, Professor Paulo Costa. Não existem palavras suficientes para agradecer todo o apoio, companheirismo e amizade ao longo deste trabalho. Pelas nossas reuniões em que aprendi muito mais do que todo o trabalho contido nesta tese. Não fosse tudo isto, aliado à sua inigualável paciência e este trabalho nunca existiria.

Ao meu co-orientador, José Gonçalves, pela ajuda e motivação durante este processo.

Aos Professores Nuno Cruz e Aníbal Matos, por me terem ajudado a dar os primeiros passos neste caminho.

Ao Isidro Ribeiro, pela amizade e apoio. Sem toda a sua ajuda, este trabalho não teria chegado tão longe. Ao Vítor Morais por aturar os meus devaneios no i117.

Ao José Lima, pela amizade, opiniões e sugestões valiosas, e pelas oportunidades que me proporcionou sempre que pôde. À Inês Soares por toda a ajuda no decorrer deste trabalho, pelo companheirismo e espírito de equipa.

A todos os meus amigos. Em especial aos que me ajudaram no decorrer desta tese, ao André Pinto, Nolasco Napoleão, Pedro Relvas, Pedro Galvão, Cláudia Rocha, Vânia Monteiro, Mónica Oliveira, Liliana Ramos, Jorge Pinto, Agostinho Rocha, Bruno Ferreira e Daniel Campos, por me acompanharem e apoiarem durante este e outros caminhos, pelas ideias que trocamos, e por me aturarem as boas e más disposições. Ao João Matos, por tudo isto, e por todas as vezes que me ”lembrou” que eu precisava de sair um bocado, e esquecer que a tese existia. À Ana Monteiro, por todas as vezes que me apoiou e me incentivou a lutar e não desistir.

À minha família, especialmente à minha Avó, Mãe e Pai, por nunca me deixar desistir e ser sempre o meu porto de abrigo. Se consegui chegar a esta fase da minha vida, foi graças a tudo o que sempre me proporcionaram. Pelos seus exemplos de coragem, humildade e integridade, mas também por toda a sua paciência e calma para me apoiarem e incentivarem, mesmo perante os efeitos menos bons da minha teimosia. À Beatriz, por me ter acompanhado durante este caminho.

Muito obrigado a todos!

Vítor H. Pinto

"If not for the compulsions of engineers, mankind would never have seen the wheel, settling instead for the trapezoid because some Neanderthal in marketing convinced everybody it had great braking ability."

— Scott Adams

Contents

1 Introduction **1**

1.1 Context . . . 1
1.2 Motivation . . . 3
1.3 Problem Statement . . . 4
1.4 Design and Research Requirements . . . 4
1.5 Scientific Contributions . . . 6
1.6 Relevant Publications . . . 6
1.7 Document Structure . . . 7

2 Background and Related Work **9**

2.1 Legged Robotic Platforms . . . 9
2.2 Robotic Joints . . . 11
2.3 Hybrid Legged-Wheeled Robots . . . 14
2.4 Simulators . . . 26
2.4.1 Rigid Bodies Realistic Simulators . . . 26
2.4.2 Numerical Simulators . . . 27
2.5 Conclusion . . . 28

3 Theoretical Concepts **29**

3.1 Parameter Estimation using Least Squares . . . 29
3.2 Cubic Hermite Splines . . . 30
3.3 Bessel Prototype . . . 32
3.4 PID Controllers . . . 34
3.4.1 Anti-windup Implementation . . . 35
3.4.2 Feed-forward Block Implementation . . . 35
3.5 State-Space Control Approach . . . 36
3.5.1 Obtaining the Transfer Function from the State-Space Model . . . 38

3.5.2 Discretization of the State-Space Model . . . 38
3.6 Controllability of a State-Space Control System . . . 40
3.7 State Control Feedback . . . 40
3.8 State Observer . . . 41
3.8.1 State Observer Algorithm Design . . . 44
3.9 Tracking System Design . . . 45
3.9.1 Tracking System Design Algorithm . . . 48
3.10 Conclusion . . . 49

4 Design, Modeling and Control of Robotic Joint Systems 51
4.1 Model of a Physical System . . . 51
4.2 Model of a Geared DC Motor . . . 52
4.2.1 Non-Linear DC Motor Model . . . 52
4.2.2 Model Linearization . . . 54
4.2.3 Linearized DC Motor Model . . . 55
4.2.4 Geared DC Motor . . . 56
4.2.5 DC Motor Model Parameters Estimation . . . 56
4.3 Design and Model Robotic Rotary Joints . . . 58
4.3.1 Rigid Joint . . . 59
4.3.1.1 Model . . . 59
4.3.1.2 Prototype Design . . . 60
4.3.1.3 Prototype Model Parameter Estimation . . . 62
4.3.2 Non-Rigid Joint . . . 67
4.3.2.1 Model . . . 67
4.3.2.2 Prototype Design . . . 71
4.3.2.3 Prototype Model Parameter Estimation . . . 76
4.3.2.4 Prototype Model including Non-Linear Elements . . . 80
4.3.2.5 Realistic Simulation including Non-Linear Elements . . . 82
4.4 Control for Robotic Joint Systems . . . 83
4.4.1 Angular Positioning Control . . . 83
4.4.2 Backdrivability—Force Controller, with Zero Reference . . . 84
4.4.3 Compliant Position Controller . . . 85

4.4.3.1 Without Obstacle in the Trajectory . . . 86
4.4.3.2 With Obstacle in the Trajectory . . . 86
4.4.4 Hermite Spline for Smooth Profile Generation . . . 87
4.4.5 Trajectory Following . . . 87
4.4.6 Weight Placement . . . 88
4.4.6.1 Without Reaching the Strength Limit . . . 89
4.4.6.2 Exceeding the Strength Limit . . . 89
4.4.7 State-Space Controller . . . 90
4.4.7.1 Rigid Joint . . . 90
4.4.7.2 Non-Rigid Joint . . . 93
4.5 Conclusion . . . 97

5 Proposed Leg for Legged-Wheeled Locomotion System 99
5.1 Relevant Notation . . . 99
5.2 Design . . . 100
5.2.1 Torque Sensing Link . . . 101
5.2.2 Prototype Leg *Foot* . . . 101
5.2.3 Leg . . . 102
5.3 Control . . . 107
5.3.1 Direct Kinematics . . . 107
5.3.2 Inverse Kinematics . . . 110
5.4 Software . . . 112
5.4.1 Communication System . . . 112
5.5 JointTune — Single Joint Software . . . 115
5.6 Trajectory Following . . . 116
5.7 Conclusions . . . 119

6 Proposed Hybrid Legged-Wheeled Locomotion System 121
6.1 Relevant Notation . . . 121
6.2 Design . . . 122
6.3 Control . . . 125
6.4 RobotTune — Robot Software . . . 128

6.5 Realistic Simulation 131
6.5.1 Path Following 132
6.6 Conclusions 146

7 Conclusion and Future Work **149**
7.1 Conclusion 149
7.2 Scientific Contributions 150
7.3 Future Work 152

List of Figures

2.1 Legs and Wheels Robot [58] 14
2.2 Chariot Robot (image kindly provided by Dr. Shuro Nakajima) 15
2.3 Chariot 3 Robot (image kindly provided by Dr. Shuro Nakajima) 15
2.4 RT-Mover Robot (image kindly provided by Dr. Shuro Nakajima) 15
2.5 WorkPartner Robot (image kindly provided by Dr. Aarne Halme) 16
2.6 PAW Robot (image kindly provided by Dr. James Smith) 18
2.7 Tarantula I Robot (image kindly provided by Dr. Abdullah Hayat) 18
2.8 Tarantula II Robot (image kindly provided by Dr. Abdullah Hayat) 19
2.9 ASTERISK H Robot (image kindly provided by Dr. Tomohito Takubo) 19
2.10 Mantis Robot (image kindly provided by Dr. Luca Bruzzone) 20
2.11 Momaro Robot (image kindly provided by Dr. Max Schwarz) 21
2.12 Creadapt Robot (image kindly provided by Dr. Jean-Baptiste Mouret) 22
2.13 CENTAURO Robot (image kindly provided by Dr. Navvab Kashiri) 22
2.14 ANYmal Wheels Robot (image kindly provided by Dr. Marko Hutter) 23
2.15 WLR Robot (image kindly provided by Dr. Xu Li) 24
2.16 WLR-II Robot (image kindly provided by Dr. Xu Li) 24
2.17 Complios Robot (image kindly provided by Dr. Arthur Bouton) 25
2.18 Wheel-Legged Mobile Robot (image kindly provided by Dr. Jacek Balchanowski) . . 25

3.1 Step Response of a System with Bessel Prototype Roots. 33
3.2 Generic PID Controller Block Diagram. 34
3.3 Generic Feed-Forward Block Diagram. 35
3.4 Discrete-Time State-Space Model Block Diagram. 38
3.5 Generic Plant with Digital Control and a Closed-Loop Observer. 41
3.6 State Observer ZOH Equivalent Block Diagram. 41
3.7 Design Plant with Added Feedback Vector. 46
3.8 Design Plant with Added Feedback Vector and Reference Input. 46

3.9 Design Plant with Added Feedback Vector and Reference Input - Rearrangement. . . 47

4.1 Motor Representation. 53
4.2 Hammerstein Model Block Representation. 54
4.3 Used Geared DC Motor — Real Version. 56
4.4 Motor Angular Speed vs. Voltage. 57
4.5 Motor $\frac{Voltage}{AngularSpeed}$ vs. $\frac{Current}{AngularSpeed}$. 57
4.6 Motor Angular Speed vs. $k \times$ Current. 58
4.7 Rigid Joint Conceptual Representation. 59
4.8 Improved Hip Rigid Joint (mechanical explosion). 61
4.9 Hip Rigid Joint. 62
4.10 Sinusoidal Sweep Signal. 64
4.11 Rigid Joint Model vs. Real Angular Positions. 65
4.12 Step response Rigid Joint (θ_{ref} = 0.5 rad). 66
4.13 Step response Rigid Joint (θ_{ref} = 1 rad). 66
4.14 Step response Rigid Joint (θ_{ref} = 1.5 rad). 67
4.15 Non-Rigid Joint Conceptual Representation. 68
4.16 Motor and Involute Spur Gear Assembly Explosion. 72
4.17 Non-Rigid Joint Explosion Assembly. 73
4.18 Mechanical Explosion Assembly Non-Rigid Joint. 74
4.19 Mechanical Explosion Assembly Non-Rigid Joint with Electronic Components. . . . 75
4.20 Non-Rigid Joint Absolute Encoders Connections. 76
4.21 Angular Position (rad) over Time (s). 78
4.22 Absolute Error (rad) over Time (s). 78
4.23 Non-Rigid Joint Model vs. Real Angular Positions. 80
4.24 Dead-Zone Specification. 81
4.25 (a) Simulator Setup for Observation of the Oscillatory Response of the Non-Rigid Joint. (b) Angular Position (Real vs. Estimated vs. Simulated. 82
4.26 Position Controller Block Diagram. 84
4.27 Response to a Step in the Position Reference. 84
4.28 Force Controller Block Diagram. 85
4.29 Response to Changes in the Position, Maintaining Force Reference. 85

4.30 Compliant Position and Force Controller Block Diagram. 86
4.31 Change in Position Reference with Compliant Controller. 86
4.32 Change in Position Reference with Compliant Controller with Obstacle in the Way. . 87
4.33 Full compliant controller block diagram. 88
4.34 Compliant controller trajectory following in angular position (up) and speed (down). 88
4.35 Spoon-Shaped Designed Part. 89
4.36 Compliant Controller Disturbance Rejection with Position reference. 89
4.37 Compliant Controller Disturbance Rejection with Position reference and Force limit exceeded. 90
4.38 State-space Controller Block Diagram (Rigid Joint). 91
4.39 Rigid Joint Controller (θ_{ref} = 0.5 rad). 92
4.40 Rigid Joint Controller (θ_{ref} = 1 rad). 92
4.41 Rigid Joint Controller (θ_{ref} = 1.5 rad). 93
4.42 State-space Controller Block Diagram (Non-Rigid Joint). 94
4.43 Non-Rigid Joint Controller (θ_{ref} = 0.5 rad). 96
4.44 Non-Rigid Joint Controller (θ_{ref} = 1 rad). 96
4.45 Non-Rigid Joint Controller (θ_{ref} = 1.5 rad). 97

5.1 Symbolic Representation of the Leg. 100
5.2 Load Cell Connections. 101
5.3 Mechanical Explosion Assembly Wheel. 102
5.4 Leg with axes, links, and joints numbering. 103
5.5 Improved Leg Configuration (front left leg). 104
5.6 Microcontroller Boards' Location on the Leg. 105
5.7 Single Leg Hardware Diagram. 106
5.8 Obstacle surpassing simulation (left to right). 107
5.9 Direct Kinematics Representations. 108
5.10 Inverse Kinematics Representations. 110
5.11 Leg Control Block Diagram. 111
5.12 Communication Diagram. 113
5.13 Information Flow Diagram. 114
5.14 Joint Tune. 115

5.15 Trajectory Following Results—Test Set 1. 117
5.16 Trajectory Following Results—Test Set 2. 117
5.17 Trajectory Following Results—Test Set 3. 118

6.1 Hybrid Legged-Wheeled Robotic Vehicle. 122
6.2 Robot Position and Orientation Variables. 122
6.3 CAD Render and Real Robot 123
6.4 Robot Dimensions. 124
6.5 Wheels' x, z Position Joint Control Block Diagram. 125
6.6 Robot Kinematic Variables. 126
6.7 Robot's Kinematic Control Block Diagram. 128
6.8 Robot Tune. 129
6.9 Communication Software and Hardware Macro Diagram. 130
6.10 Robot simulation model. 131
6.11 Control in the x and y axes: setting the speeds v (m/s), v_n (m/s), and w (rad/s) of the robot (all images were taken from the same perspective; the axis in the foreground corresponds to the x-axis). 132
6.12 Line Tracking Algorithm Representation 133
6.13 Horizontal Line Tracking: 2D Position (m) — Test at 0.3 m/s. 136
6.14 Horizontal Line Tracking: 2D Position (m) — Test at 0.8 m/s. 136
6.15 Diagonal Line Tracking: 2D Position (m) — Test at 0.3 m/s. 137
6.16 Diagonal Line Tracking: 2D Position (m) — Test at 0.8 m/s. 138
6.17 Horizontal Line Tracking with $\theta_r = 25^o$: 2D Position (m) — Test at 0.3 m/s. 139
6.18 Line Tracking: 2D Position (m) with Outside Approach. 140
6.20 Curve Tracking: 2D Position (m) — Test at 0.3 m/s. 143
6.21 Curve Tracking: 2D Position (m) — Test at 0.8 m/s. 144
6.22 Curve Tracking: 2D Position (m) — Test with Interior Approach. 145
6.23 Curve Tracking: 2D Position (m) — Test with Exterior Approach. 145
6.24 Curve Tracking: 2D Position (m) — Test between Interior and Exterior Approaches. 146

List of Tables

3.1 Roots of the Bessel polynomials for settling time of 1 second. 33

4.1 Estimated Motor Parameters. 58
4.2 Absolute Error Analysis. 79
4.3 Estimated Joint Parameters. 79

5.1 DH Parameters for the Leg. 108
5.2 Lengths of the Leg Links. 109
5.3 Set of x^l, z^l positions. 116
5.4 Errors in x^l, z^l Positioning (in m). 119
5.5 Errors in θ_1, θ_2 Positioning (in *rad*). 119

6.1 Robot Dimensions. 123
6.2 Individual Leg Kinematic Calculation. 127
6.3 Line Path Following Errors. 140
6.4 Curve Path Following Errors. 144

7.1 Communication Channels Transmission - Robot Tune 154
7.2 Communication Channels Reception - Robot Tune 155
7.3 Communication Channels Transmission - Joint Tune 155
7.4 Communication Channels Reception - Joint Tune 156
7.5 Communication Channels Transmission - Robot Firmware USB Port 157
7.6 Communication Channels Reception - Robot Firmware USB Port 158
7.7 Communication Channels Transmission - Robot Firmware Serial Port 158
7.8 Communication Channels Reception - Robot Firmware Serial Port 159
7.9 Communication Channels Transmission - Joint Firmware USB Port 159
7.10 Communication Channels Reception - Joint Firmware USB Port 160
7.11 Communication Channels Transmission - Joint Firmware Serial Port 161

7.12 Communication Channels Reception - Joint Firmware Serial Port 161

Acronyms and Symbols

Acronyms

3D	Three dimensional	OTA	Over-The-Air
AI	Artificial Intelligence	PD	Proportional-Derivative
ASCII	American Standard Code for Information Interchange	PI	Proportional-Integral
		PID	Proportional-Integral-Derivative
BLDC	Brushless DC	PLA	Polyactic Acid
CAN	Controller Area Network	PWM	Pulse Width Modulation
CAD	Computer Aided Design	QDD	Quasi-Direct Drive
CNC	Computer Numerical Control	RGB	Red-Green-Blue
COT	Cost of Transport	RGB-D	Red-Green-Blue-Depth
DC	Direct Current	RLS	Recursive Least Squares
DH	Denavit–Hartenberg	ROS	Robot Operating System
DOB	Disturbance Observer	RPi	Raspberry Pi
DOF	Degree of Freedom	RPM	Revolutions Per Minute
FPGA	Field Programmable Gate Array	RSEA	Rotary Series Elastic Actuator
GPIO	General Purpose Input/Output	RX	Receive Channel
GUI	Graphical User Interface	SEA	Series Elastic Actuator
HIVC	Highly Integrated Valve-controlled Cylinder	SPI	Serial Peripheral Interface
		SS	Steady State
I2C	Inter-Integrated Circuit	TX	Transmit Channel
IDE	Integrated Development Environment	USB	Universal Serial Bus
IMU	Inertial Measurement Unit	VDC	Volt Direct Current
LiPo	Lithium Polymer	ZOH	Zero Order Hold
LS	Least Squares		

CHAPTER 1

Introduction

"The dream begins with a teacher who believes in you, who tugs and pushes and leads you to the next plateau, sometimes poking you with a sharp stick called 'truth'."
— Dan Rather

The following chapter focuses on the context and motivation to carry out the presented research. In addition, it also defines the problem statement and the design and functional requirements that are relevant during the research process. Finally, the scientific contributions, the relevant publications, and the structure of the document will be presented.

1.1 Context

In recent years, the use of robotics to perform multiple types of tasks has increased significantly. Some tasks are impossible, or almost, for humans to perform, such as lifting and transporting extremely heavy loads. Others are just repetitive and continuously performed, where robots can improve accuracy and repeatability. Even performing tasks in harsh or inaccessible environments where humans cannot reach, robotic systems can help. From a more industrial perspective, there are two main types of systems in use - robotic manipulators and robotic vehicles. In repetitive tasks, robotic manipulators are massively used, improving accuracy and reducing operator fatigue. Manipulators can also be used in the context of cargo handling, to place loads on a conveyor. Concerning mobile robotics, one of the main applications is to serve as a cargo carrier, but they can also be used as support vehicles, depending on its configuration.

As far as robotic vehicles are concerned, some difficulties arise concerning the locomotion system, in particular because the ground on which the vehicle moves may vary along its path. Moreover, the presence of stairs is a problem that still exists today, being difficult for classical traction systems to

deal with. To solve the first problem, vehicles with leg traction are usually used, but these types of vehicles generally have a lower speed and consume more energy to move, in comparison. A vehicle using wheel-based traction has also increased difficulty to deal with the floor irregularities. Usually, vibrations are reduced using damping systems, but at the expense of increasing the system's complexity and its cost.

As stated by Jaiem et al. [1], energy consumption is a crucial question when developing autonomous mobile vehicles. This is a significant restriction that strongly influences their working autonomy.

The practice of "the stiffer, the better" has been used in the joint design process, since rigid joints present an increased accuracy in position control. However, with the growing use of robotic systems in environments shared with humans, and the proliferation of collaborative robots, it is necessary to search for safer systems. Non-rigid joints try to overcome the difficulty of working in highly dynamic environments by improving the robot tolerance to impacts. This configuration also helps increasing safety in collaborative scenarios, since non-rigid joints can be used in compliant control configurations.

Concerning the non-rigid joint design and control, two main paths are being explored - the Quasi-Direct Drive (QDD) and the Series Elastic Actuator (SEA). However, as stated by Grandesso et al. [2], no mechanism stands out among others. This is due to the fact that the suitability of each of them is directly related to the type of task to be performed, the environment, the type of system that is used, among others.

QDD is a development trend that uses one or more Brushless DC (BLDC) motors combined with low ratio gearboxes, or even without gearbox, that can use or not chains or belts to transmit the power [3–5]. In these solutions, current sensors are used and accurate force control capability is achieved, without the need to use force/torque sensors. The low-reduction transmission ratio decreases the backlash, reflected inertia and friction, while increasing the backdrivability, very useful in robotic manipulation. The disadvantages that can be pointed out to these systems are the constant high energy consumption, since the motor is working in high current regimes, and also low output torque. Due to the high current regimes, the windings heat up, consequently increasing Joule losses.

SEA integrates one or more elastic components that provide the system with a mechanical energy storage and source, increasing the system's mechanical compliance [6–8]. Typically, the motor is connected to a high-reduction gearbox and its shaft is connected to a spring or system of springs,

that in turn are connected to the joint output. The advantages are the use of passive components that can store and regenerate energy, increased impact tolerance and high output torque. As the previous, this solution also brings out disadvantages, like low bandwidth and increased difficulty while controlling the systems sudden movements. Comparing to the previous solution, the increased mechanical fabrication cost is also a downside.

The QDD is a possible solution to implement and is used for projects similar to the one presented. Nevertheless, after studying the best one to fit our requirements, the SEA stands out to be a slightly better option to our research, given the energy consumption and the buying cost. Considering the tradeoffs, SEA joints are considered the best solution.

1.2 Motivation

Using non-rigid joints increases the system tolerance to impacts, thus improving the system ability to work in highly dynamic environments. Furthermore, this joint configuration allows for force control, enabling compliance control, thus increasing its capabilities in collaborative scenarios.

Focusing on the current research, the use of non-rigid joints allows the vehicle to adapt more easily to non-flat terrains and highly dynamic environments. Furthermore, when used in a robotic manipulator, this configuration increases the safety of the robot, but also of humans who work in the same space, in collaborative environments.

Legged-wheeled locomotion systems try to combine the advantages of both legged and wheeled systems, while avoiding their disadvantages. However, this premise is very difficult to achieve, since both systems are very different, typically use different control methods, and use different hardware techniques. Currently, few projects address legged-wheeled vehicles using mammal-alike legs, given that their design, construction and control are more complex. Another issue is that these systems tend to be more expensive, making it difficult to include them in a project.

So, one of the main motivations for this work was to design, model and control a modular, cost-effective system that can be used to develop a wide range of algorithms, from positioning, trajectory planning, performance analysis, among others.

Typically, the works under development use brushless DC motors, or hydraulic systems which increases the cost of the project, and decreases its autonomy, due to the increased energy consumption.

Considering this, the platform should be designed to combine each traction system's best features and try to reduce their disadvantages. An important feature that will be addressed is the introduction

of a *low-power* mode, which allows for a decrease in energy consumption.

Finally, there is the motivation to create a distinct locomotion system, giving emphasis on other degrees of freedom, thus obtaining a different omnidirectional configuration.

1.3 Problem Statement

The design of a hybrid legged-wheeled locomotion system must combine the advantages of both legged and wheeled systems, while trying to avoid their disadvantages. Currently, only few projects address legged-wheeled vehicles, since their design, construction and control are more complex. Since more actuators and sensors are required, these systems tend to be more expensive, as well as exhibiting increased power consumption.

Non-rigid joints increase the system tolerance to impacts, thus improving the system ability to work in highly dynamic environments. This configuration allows for force control, enabling compliance control, thus increasing its capabilities in collaborative scenarios. However, its design is more complex and the positioning precision is inferior. To overcome these disadvantages, more accurate models and improved controllers are required.

1.4 Design and Research Requirements

In the case of autonomous legged vehicles, the problem of energy consumption becomes critical, as this type of locomotion typically consumes more than the classical wheeled one, as supported by Silva and Machado [9]. Therefore, from the development of the very first prototype of the non-rigid joint, it was decided to use a passive system, apart from the motor. The remaining components are passive, to decrease energy consumption.

Regarding the mechanical design, several requirements were defined to reach the desired solution:

- The purpose of this solution is both indoor and outdoor navigation;
- The solution should be cost-effective and modular;
- The proposed solution should use brushed DC motors;
- The vehicles can operate in environments with different characteristics;

.

Considering the mentioned requirements, some other following items were decided:

- The proposed solution should use four limbs, which must follow a legged-wheeled configuration, with a wheel at the bottom of the leg;
- The system must be able to navigate in three modes: purely legged, purely wheeled, and hybrid;
- To reduce energy consumption, the system should incorporate a *low-power* configuration, in which the legs of the vehicle rest on a minimum z. This presents a consumption similar to a mobile robot with four-wheel drive classic traction system;
- Each limb and its integrated parts must be easily assembled and disassembled;
- The uppermost joint should allow each leg to rotate, at least 180^{o}, allowing the vehicle to move across the local x and y axes, as well as rotating around the vehicle's own axis;
- Considering the previous items, the robot must be controllable in the z axis. Each limb must have, at least, three DOF;
- To accommodate impacts, floor irregularities, and better deal with obstacles, at least one of the joints in the vertical movement of each limb must be non-rigid;
- The limb configuration must be able to sense force, enabling compliant movements.

Considering that the system is proposed for an autonomous mobile hybrid platform, from a researching point of view, the design was focused on three principles:

1. *Cost-Effective*—The system's construction must be affordable so that any changes that needed to be performed do not have a significant impact on the vehicle's prototyping and maintenance budget;
2. *Expandable*—The vehicle must be easily expanded, thus it must be designed to have easy adaptability. This allows, for example, the incorporation of new sensors, actuators, and tests with different systems and algorithms without much mechanical effort;
3. *Modular*—The system must be designed to be easily changed only in each one of its parts. This means it must be easily changed locally, not requiring to change the robot, since this task can be highly time-consuming.

In addition to these principles, the vehicle should also be able to deal with non-flat terrains, as it is the closest to reality. This means that the system has to be mechanically robust and capable of dealing with these irregularities.

For the limb design, it was decided to use 3D printing as the main source of development of the mechanical parts. Furthermore, it was decided to use Polylactic Acid (PLA), given its quality-cost ratio. This solution greatly reduces the cost of obtaining parts for the vehicle, both economically and in terms of time. The material cost is significantly lower, compared to metallic ones, as well as 3D printing speed is considerably higher compared to Computer Numerical Control (CNC) machining.

1.5 Scientific Contributions

The presented work aims to contribute with several solutions to the problem stated in 1.3, such as:

- The design of a joint with a passive approach to its non-rigid elements, enhancing the system's performance and increasing safety and robustness;
- The model conception and estimation, containing non-linear elements applied to the proposed non-rigid joints;
- The proposal of several control modes for the system, capable of driving the proposed joint with accuracy and compliance;
- The conception and design of a new platform using a legged-wheeled locomotion system, which is modular, cost-effective and reconfigurable, and can be used for multiple purposes;
- The proposed platform presents several different locomotion modes, capable of optimizing energy consumption, speed and locomotion through irregular environments.

1.6 Relevant Publications

As a result of the research carried out during the PhD program enrollment, there were published three relevant conference articles and four relevant journal articles. These relevant contributions are detailed below:

Conference Articles

- Pinto, Vítor H., José Gonçalves, and Paulo Costa. ”Model of a DC motor with worm gearbox.” Portuguese Conference on Automatic Control. Springer, Cham, 2020. [10]
- Pinto, Vítor H., José A. Gonçalves, and Paulo Costa. ”Introduction to DC motors for engineering students based on laboratory experiments.” 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT). IEEE, 2019. [11]
- Pinto, Vítor H., et al. ”Modeling of an Elastic Joint: An Experimental Setup Approach.” Portuguese Conference on Automatic Control. Springer, Cham, 2020. [12]

Journal Articles

- Pinto, V.H.; Gonçalves, J.; Costa, P. Modeling and Control of a DC Motor Coupled to a Non-Rigid Joint. Appl. Syst. Innov. 2020, 3, 24. https://doi.org/10.3390/asi3020024. [13]
- Pinto, V.H.; Gonçalves, J.; Costa, P. Towards a More Robust Non-Rigid Robotic Joint. Appl. Syst. Innov. 2020, 3, 45. https://doi.org/10.3390/asi3040045. [14]
- Pinto, V.H.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of a Single Leg for a Legged-Wheeled Locomotion System with Non-Rigid Joint. Actuators 2021, 10, 29 https://doi.org/10.3390/act10020029. [15]
- Pinto, V.H.; Soares, I.N.; Rocha, M.; Lima, J.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of an Autonomous Legged-Wheeled Hybrid Robotic Vehicle with Non-Rigid Joints. Appl. Sci. 2021, 11, 6116. https://doi.org/10.3390/app11136116. [16]

1.7 Document Structure

This thesis is divided into seven chapters.

Chapter 1 presents a brief context on mobile robotics and hybrid locomotion systems. It also lays out the motivation, some predefined requirements and the scientific contributions of the presented work.

The background and existing solutions related to robotic joints and their applications, legged robotic vehicles and hybrid legged-wheeled robotic platforms is described across Chapter 2.

Chapter 3 is focused on introducing the theoretical concepts that were used during the conducted research.

Chapter 4 sets out the modeling and control concepts, important to define the used models for the motor and both joints. The concepts used to define the model, estimate their parameters and control each system are presented.

Throughout Chapter 5, the proposed solution for a single leg will be presented, along with the designing definitions, assembling instructions and details of all the used systems.

In Chapter 6, the relevant aspects of the robot prototype, kinematics and developed software are mentioned. The path following algorithms are detailed, along with the obtained results for the robot control.

Finally, in Chapter 7 the conclusion of this thesis is presented, as well as some suggestions for possible future work.

CHAPTER 2

Background and Related Work

"Strive for perfection in everything you do. Take the best that exists and make it better. When it does not exist, design it."

— Sir Henry Royce

Historically, robots started to be applied as tools to support industrial manufacturing repetitive tasks, impossible to be performed by humans, or even that presented high safety risks. One of the first robotic manipulators that have appeared was the Unimate, presented in 1959 [17], designed and built by George Devol. Before that, in 1957, SPURV was built at the University of Washington's Applied Physics Laboratory by Murphy, François and, later, Ewart [18] as a unique robotic system for underwater research. Later, in 1966, Nilsson and his team at the Stanford Institute of Research presented one of the first mobile robot controlled by artificial intelligence (AI) - Shakey [19]. From a historical point of view, after almost 60 years, research in robotics still continues. Whether due to the fact that technology is constantly advancing, or that new areas arise where robots become viable.

2.1 Legged Robotic Platforms

Some platforms that can be found in literature are the legged robots. These have a diverse number of non-transformable members, which are used for their movement. An overview of these vehicles was developed by Tenreiro and Silva [20], who defined the advantage of a legged vehicle in its mobility in naturally uncertain terrains. These vehicles are able to support their members in different positions, as well as their fault tolerance during locomotion is increased.

However, this type of vehicle also has limitations, such as its stability on smooth terrain, the increased use of actuators for moving, which inevitably increases the vehicle's cost and weight, and the complexity of its control. There are several examples in the literature regarding multi-legged

vehicles, such as the one presented by Flores et al. [21] and the predecessor of the famous Spot, the BigDog [22].

Regarding innovative solutions in the legged vehicles field, the ScarlETH [23] solution is a good example. This platform was developed at ETH Zurich as a Series Compliant Articulated Robotic Leg, inspired in biological systems, using a series elastic actuators (SEA) connected through two chains and a belt. The solution uses only one DC motor to control the leg, and uses several springs to deal with impact collisions. Also, a miniature chain drive is designed as a mechanical fuse, to be the weakest part for the knee, with a breaking torque around 20% below the allowable for the gearboxes. The leg weights about 3 kg, segments around 0.2 m and has a maximum payload of 3.3 kg, the system is controlled at a 10 kHz frequency and the motors velocity controller receives commands at 1 kHz. The used interface was CAN (Controller Area Network)-bus and the position/torque control loops are implemented at 500 Hz on a Field Programmable Gate Array (FPGA). The movement speed of the leg is nearly 0.6 m/s.

Another solution, developed by Shi et al. [24], uses hydraulic series elastic actuators with the intention of using this leg in a quadruped vehicle weighing over 150 kg. The authors designed a 2 DOF planar articulated leg, with a maximum length of 650 mm, using springs and two modular elastic elements for the hip and knee joints. The hip and joints' rest angles are 70 and 140 degrees, with an angular course of 45 and 40 degrees, respectively. The authors use a highly integrated valve-controlled cylinder (HIVC) with an elastic stiffness of the elements of around 10 N·m/degree, and springs that have an elastic module of approximately 190 GPa. The authors have performed some positioning and impact experiments, using impedance control, resulting in errors in position of about 10% and 5% in the hip and knee joints, respectively, as well as a maximum error in torque of around 2 N·m.

Concerning hybrid platforms, which is commonly a combination of two mobility technologies, several examples can be found in the literature. There are some fascinating concepts, including some hybrid legged-wheeled robots that will not be detailed in the following subsection, since they are not precisely in line with the proposed solution, but still worth mentioning. One example is the HyLMoR Robot [25], a hybrid UAV/quadruped with terrain and aerial locomotion ability. It uses four brushless motors to rotate the aerial propellers and three servo motors in each leg, turning this robot into a very reliable hybrid solution.

Another solution, designed to work with laboratory rats is the WR-3 [26], that was designed as a wheeled-legged robot. The authors have studied the movements of the rats to create a biomimetic

robot, using the average size of the rats. Then, they have successfully introduced the robot to interact with the rats and the robot has blended in, being able to interacting with other rats and influencing their behaviour in some degree. EHR [27] is a hybrid amphibian capable of navigating in water and land. Its design is based on the premise that the robot must be able to move in water and on land, so the authors developed a locomotion mechanism based on wheels, but took advantage of the outer half-moon of the wheel to create a compartment with water, being capable of creating fluctuation that allows the robot to stay afloat.

A very popular wheel-leg concept is the transformable one, like Quattroped [28], its sucessor TurboQuad [29] and WheeLeR [30]. This concept divides the robotic wheels in two or three parts, connecting and disconnecting them to create a circular wheel or legs in the form of some kind of paddles. In the case of Quattroped/TurboQuad the vehicle has four of these mechanisms, like a quadrupedal robot, and in the case of WheeLeR it only has two mechanisms, a biped, with a support point on the front of the body.

2.2 Robotic Joints

Regarding the actuation of the joints designed for robotic systems, there are three different types: hydraulic, electric and hybrid. The hydraulic actuators provide the joints with high power, high force density and high-performance force control. However, they are very complex systems, expensive, with high noise and with high inertia, but applied in the HyQ robot [31]. Electrical actuation has become widely used in more recent years, being divided into two main areas - the quasi direct driven (QDD) - used in the robot MIT Cheetah 3 [32] and the series elastic actuation (SEA) - used in the robot Valkyrie [33]. The first uses low gear ratio, with low kilo RPM/Volt (KV) Brushless DC (BLDC) motors, resulting in motor current control of the motors, equivalent to the regulation of the output force. The latter typically uses high reduction ratio gears, using input and output encoders and elastic components applied to store energy in the system. However, SEA joints have a lower dynamic response than the QDD ones. The hybrid configuration uses both actuators to perform the movement of the joints.

To increase safety and robustness in the performance of industrial robots, by decreasing the chance to easily break their joints or hurt someone, the use of non-rigid joints is being increasingly found in the literature and in the industry. These mechanisms contain elastic components (such as springs), which introduce mechanical energy storage in the system, potentially increasing the maximum output

power [34]. Despite the evolutionary effort for using flexible and more sophisticated solutions for the market [35–40], it is not yet very easy to find solutions in mature states with which research teams can explore and perform experimental work, as Hoffman et al. [41] report.

Even so, some works should be mentioned, since they present varied solutions. The work from Wang et al. [42] is one of them, which resorts to six degrees of freedom (DOF) fully parallel manipulator with elastic joints. The elastic component is used as the last link of an industrial manipulator and consists of three links connected by an elastic ball joint and a Hooke's joint on each side. This joint is an example of a purely mechanical solution, but it does not have a damping effect on the system, which is very useful in case of an impact, given its important characteristic.

A different solution was proposed by Yoneyama and Ito [43], which presents a flexible translational joint mechanism. The authors consider an adaptable adaptive joint with adjustable elasticity that can slide in multiple directions. The joint is built with servos that control the flexibility of a rubber band that locks into a ball-bearing bed. Such joints are used as lower leg parts for a humanoid robot. Despite adding an elastic component to the joint, this solution also does not have a damping component. Another characteristic of this joint is its variable elasticity, however it consumes energy for its control.

Kakogawa and Ma [44] have designed a fully passive elastic joint to be used on an inspection robotic vehicle. Each of the three links uses a spring, which can be different from each other, to introduce a torsional elasticity capability, that allows the joints to be flexible even when the wheel is stopped. This joint also has no damping component, which could be an advantage for the use intended for this system.

For Human-Robot interaction applications, Kong et al. [45] have developed a precise actuating system to perform some tests regarding force control of robotic systems interacting with humans. A Rotary Series Elastic Actuator (RSEA) was designed and is presented, resorting to a 200 W Brushless DC (BLDC) motor attached to a gearbox, connected to a torsional spring in its shaft. This system is intended to be similar to human knees or ankles, as stated in the paper, and these were used as a reference for choosing and modeling the components of the developed system. It has a relative encoder connected to the motor shaft, as well as an absolute encoder after the spring, that is connected to the motor. The implemented controller was a force feedback one, through PD controllers, and using a disturbance observer (DOB) to precisely generate the desired torque.

Robinson et al. [46] have developed a SEA prototype, that was later used in the RoboKnee [47],

using a brushless DC motor rigidly linked to a ball screw, being the latter connected to the output through four compression springs. This prototype can output over 1350 N and moves at a speed of 0.25 m/s. The described model uses a spring component but a damping component as well, while using a simple motor saturation model, having later used this model to select the 315 kN/m spring constant.

Zhang et al. [8] developed a solution for a SEA that consists of using a closed-loop concept in order to reduce the size of the traditional structure, which implies the placement of an elastic element after the motor. The authors used a DC motor, a planetary gearbox, an encoder, as is common in this type of joints, but after that developed a module, which they called four-spoke where they connected linear springs, connecting the whole system through Bowden cables. The length of the joint is 0.140 m, its diameter is 0.08 m and its weight is around 0.85 kg. To control the system, the authors have designed a torque controller with PD compensation for the joint's motion and friction, as well as a disturbance observer, achieving a feedback control, being that later was implemented an impedance controller.

Regarding the control of an elastic joint, some works address it, from the 1980s to the present day [48–54], indicating that this topic is still an open line of research. Since the non-rigid joint increases the flexibility of the robot, the control methods must be adequate for the movement and permissiveness of such specific elastic joints.

The work from Zolo et al. [55] also stands out, since a compliance control for an anthropomorphic robot with elastic joints is used. It resorts to a PD plus gravity controller and the addition of a gravity-biased motor position variable. One thing that is anticipated by the control method is that, if the manipulator comes into contact with the environment, some laws achieve compliant behavior during this contact. The compliance control formulations were tested experimentally.

Ferretti et al. [56] used an impedance control strategy in industrial manipulators to perform tasks where compliant position motion is required. This method focuses on creating a relation between errors in contact forces and position during a movement. The authors present a controller designed to use an impedance filter, as well as their experiments on a physical manipulator, which obtained relevant results.

Also applied to a robotic manipulator, Yin et al. [57] use a compliant control strategy for a single tendon-sheath actuator. The modeling and validation methods are presented, being performed experimentally. The model parameter identification is presented and the compliant control results are stated, even in the presence of collision detection, obtaining favorable results.

2.3 Hybrid Legged-Wheeled Robots

Hybrid locomotion systems can be defined as mechanical systems capable of using multiple and different propulsion methods, such as legs, wheels, tracks, among others. The number of possible combinations between them is significant, being distinct the modes that the vehicles use to move. Legged-wheeled locomotion systems are particular cases used for robots, increasing their degrees of freedom. The system's specific design can, in certain ways, be useful for navigating through straight clear paths or irregular ones. This particular characteristic tries to enhance the locomotion properties, taking advantage of both legged and wheeled systems, and trying to avoid each disadvantage. The legged-wheeled structure is interesting, since it can change the posture of the body, adapting it to the terrain conditions, for example.

Back in 1986, Oomichi and Ibe [58] have developed a hybrid vehicle with wheels and legs. Even though the locomotion is reported as under development in the paper, it is noticeable that the vehicle resorts to 6 wheels and 8 joints, having 14 degrees of freedom (DOFs). It appears to have a combination of brushed and brushless DC motors, in the configuration presented in Figure 2.1. Through this example, it is possible to conclude that the theme of hybrid vehicles that use legs and wheels locomotion systems dates decades ago. However, and through the examples that will be presented below, it is also easy to conclude that this is still an open research topic, with several different solutions.

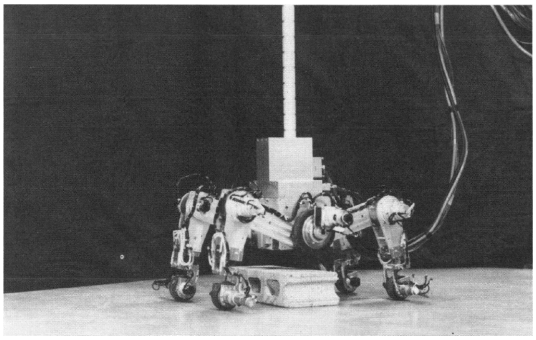


Figure 2.1: Legs and Wheels Robot [58]

Also, one of the first leg-wheel robotic projects is Chariot [59], developed by Kimura et al., iterated to Chariot 2 [60] and used in its improved version - Chariot 3 (Figure 2.3) - at least until 2009 by Nakajima and Nakano [61]. The first version of Chariot has two 0.32 m radius wheels located in both sides of its body, 4 legs with a length of 0.85 m when extended and weighs about 20 kg. On a flat plane, it is mentioned that the vehicle can achieve speeds up to 5.56 m/s on a flat terrain. Chariot 3 uses

15 DC servo motors from 40 W to 60 W and the same physical configuration with two wheels. It is capable of overtaking obstacles up to 0.1 m and slopes up to 0.35 rad. Each leg has 0.78 m, the wheel diameter is 0.64 m, the body's dimensions are 0.92 x 0.40 m (L x W) and the vehicle weighs 65 kg. The robot was tested on unevenness dimensions with 0.2 m differences, being capable of overcoming these difficulties.



Figure 2.2: Chariot Robot (image kindly provided by Dr. Shuro Nakajima)

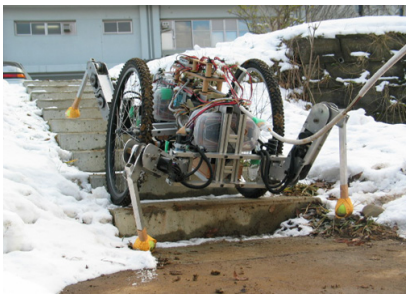


Figure 2.3: Chariot 3 Robot (image kindly provided by Dr. Shuro Nakajima)

A later work by the same author, Nakajima [62] is the development of the RT-Mover 2.4 leg-wheel mechanism. The platform has 0.80 x 0.63 x 0.46 m (L x W x H) with 0.20 m diameter wheels and weighs 28 kg. It uses ten 24 VDC servo motors and the maximum vehicle speed is 0.63 m/s. The experimental work presented in the paper shows that the vehicle was able to cross 0.52 rad angular slopes in full-wheel mode, as well as going up and down slopes with 0.15 m in leg mode, taking it around 50 s to achieve one step-down.



Figure 2.4: RT-Mover Robot (image kindly provided by Dr. Shuro Nakajima)

WorkPartner is an interactive service robot designed at the Intelligent Machine and Special Robotics Institute from the Helsinki University of Technology for outdoor applications [63, 64]. This centaur-like robot uses a hybrid system philosophy, with the purpose of increasing the capacities obtained with a standard system. The vehicle sets on a hybrid platform(Figure 2.5), which allows to increase motion flexibility and control heading, weighing around 280 kg, and having a payload of 40 kg. It has a fully electrical actuator system, despite the power system being a combination of batteries and combustion generator, capable of providing 1 kW of energy. The robot has 3 navigation modes, a fully wheeled, a legged and wheeled and fully legged. Being fully wheeled it always moves with a 4 wheel drive configuration and it can reach speeds of around 1.95 m/s, while the walking mode is designed for slow-motion only. The platform is divided in half and a joint is linking them, used for heading control when driving fully wheeled. In the legs, a combination of off-the-shelf and designed components is used, recurring to Brushless DC linear actuators (250 W, 48 V) and a tailor-made gear. It has a de-centralized hardware configuration, using a central computing system (PC104) and 4 leg independent controllers. Unfortunaltely, despite the authors mention that the "operation time (...) must be long enough, preferably several hours", a precise estimation of the robot's autonomous mission expected time is not provided, neither the full power consumption. The communication is achieved through CAN protocol between the main processor and the sub-units. As a curiosity, in 2015 this robot was donated to the Museum of Technology, in Helsinki, and is displayed until the present.



Figure 2.5: WorkPartner Robot (image kindly provided by Dr. Aarne Halme)

Across the years, the team from the Division of Machine Theory and Mechatronical Systems

at Wroclaw University of Technology [65, 66] have designed and built the LegVan, a wheel-legged robot that can be used for travelling on wheels or walking, depending on the needs. The developed vehicle has four limbs with a wheel attached to each one of them, weighs 35 kg, and its dimensions are 0.75 x 0.45 x (0.35-0.65) m (L x W x H), being capable of overcoming obstacles with heights up to 0.30 m. The authors have developed a special suspension system that enhances the levelling and walking functions. For the robotic suspension structure it was assumed that the robot would navigate through urbanized terrains, although it can encounter unevennesses like slopes, steps and thresholds, as well as it is assumed that the obstacles must be walked over. The vehicle uses fourteen motors, being four of them for driving, two to turn applied to the rear limbs, four responsible for horizontal positioning and the final four for obstacle avoidance. The robot mainly uses its wheels to navigate, using the four-bar linkage to level the robot body and for obstacle negotiating. The robot is divided into central control computer (PC104), wirelessly communicating with the user computer. Besides this, the robot has 3 sub controller modules (MC68332), in charge of controlling the motors for motion execution and to obtain sensor measurements.

Platform for Ambulanting Wheels (PAW) is a hybrid wheeled-leg robot developed by Smith *et al.* [67] with maximum dimensions of 0.494 x 0.366 x 0.170 m (L x W x H) including 0.066 m diameter wheels on the bottom of the legs, as pictured in Figure 2.6, weighing 15.7 kg, with each leg weighing 1.2 kg. The vehicle has four 90 W Brushed DC motors to drive each hip joint and four 20 W brushed DC motors to drive each one of the wheels. Also, each leg includes a pair of extension springs. Electrically, the hardware is centralized in a PC104 computer, remote motor drives and 3 battery packs with a capacity up to 9.5 Ah. Regarding control, each joint is controlled with a Propostional-Integral-Derivative (PID) controller for either position or speed and each wheel has a speed PID. Through the experimental tests, the authors could achieve peak speeds up to 2.0 m/s in straight line. The operational range is defined as over 2562 m within a time frame of 59 minutes, with four interruptions. The mentioned power consumption is 25 W when the robot is resting on the ground, around 58 W when braking on a slope and between 51 and 56 W while navigating at 1.4 and 2.0 m/s, respectively.

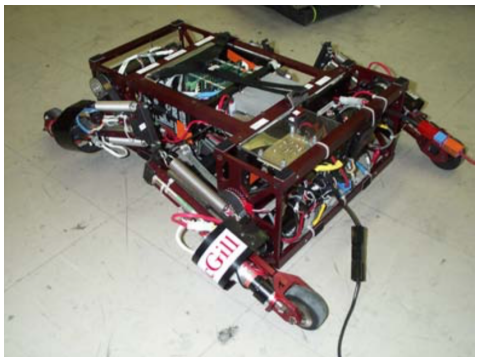


Figure 2.6: PAW Robot (image kindly provided by Dr. James Smith)

The Engineering Product Development team from Singapore University of Technology and Design has developed two prototypes for drain mapping, the Tarantula series, consisting of Tarantula I [68] and its evolution Tarantula II [69], being the robots depicted in Figures 2.7 and 2.8, respectively. Tarantula I has a central body and four legs attached, each one with four DOF via RPRR (Revolute/Prismatic) joints and a telescopic extension mechanism. Tarantula II gained an extra DOF in each leg, since it now has the ability to fold in the middle. Both versions rely on only two brushed DC motors (12 V, 80 Revolutes per Minute (RPM)) to lower and raise the vehicle, being two legs actuated by the same motor. Tarantula I configuration joins rear and front legs, while Tarantula II chooses to divide in right and left. Both have wheels on the bottom of the legs, with a DC motor for driving and a Brushless DC motor for rotating it, using a total of ten actuators. The robots use a Raspberry Pi (RPi) for central computing and Arduino Boards for decentralized control and the whole system is powered by 24 V batteries. Tarantula I does not mention the programming language, but Tarantula II operating system is ROS, installed in the RPi, being the communication between the parts achieved through CAN protocol.

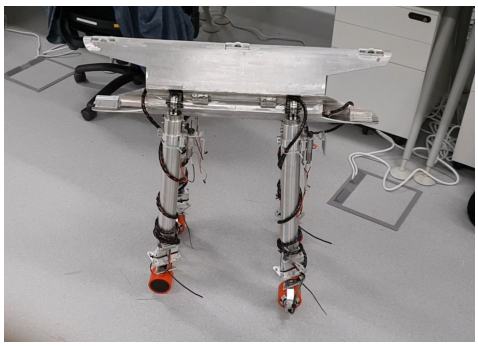


Figure 2.7: Tarantula I Robot (image kindly provided by Dr. Abdullah Hayat)

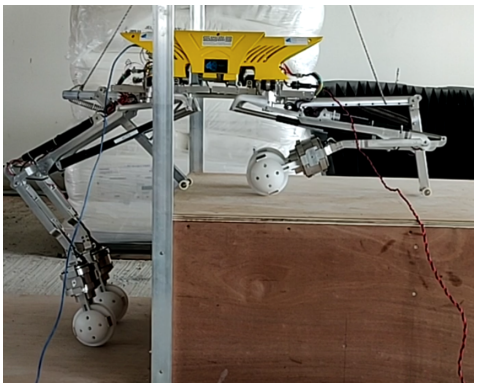


Figure 2.8: Tarantula II Robot (image kindly provided by Dr. Abdullah Hayat)

ASTERISK H [70] is a legged-wheeled vehicle hexapod developed in Osaka and Yamagata Universities to travel through rough terrains. It can use three locomotion modes, fully-legged, hybrid and fully-wheeled. The vehicle, which can be viewed in Figure 2.9, uses thirty DC servo motors from Dynamixel, it has a driving wheel on the bottom of each leg and the maximum dimensions are 0.815 x 0.706 x 0.334 m (L x W x H), weighing 3.4 kg. Each leg has a maximum length of 0.33 m and uses five motors to achieve four DOF, excluding the driving wheel, and its running speed is 0.3 m/s. The vehicle's vertical symmetry was designed to provide equal workability for all the limbs. The motors have built-in encoders and torque sensors, used for the robot to detect on which surface it is moving. While on fully-wheeled mode, the robot uses a tripod configuration, but when a step of any kind is sensed, the other legs are actuated to help overcome these obstacles.

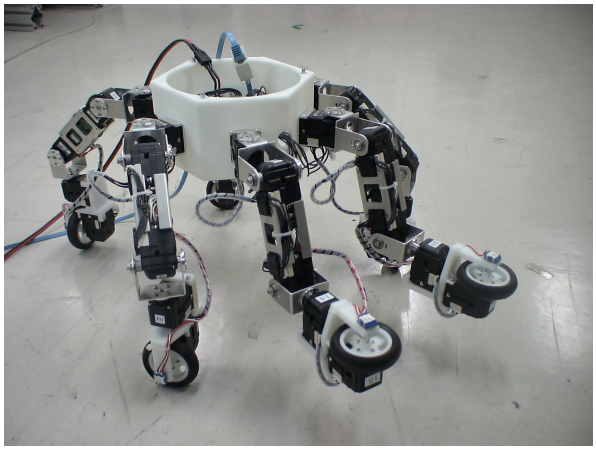


Figure 2.9: ASTERISK H Robot (image kindly provided by Dr. Tomohito Takubo)

Bruzzone et al. have designed Mantis Robot in 2014 [71], but improved its concept in 2017 [72].

The vehicle's picture is presented in Figure 2.10 and has a very different design, using four wheels (2 x 0.055 m + 2 x 0.045 m) for wheeled mobility and two structures called legs to help overcome obstacles, which have small wheels (2 x 0.015 m) as some kind of knee to the leg. Its dimensions are 0.35 x 0.3 x 0.2 m (L x W x H), has a maximum payload of 1 kg and is said to be suitable for surveillance tasks. It can overcome steps up to 0.16 m, climbs slopes up to 35.6 degrees, with a maximum speed of 0.64 m/s. The vehicle uses four DC motors with gearbox, two for the frontal wheels and another two for controlling each leg independently.

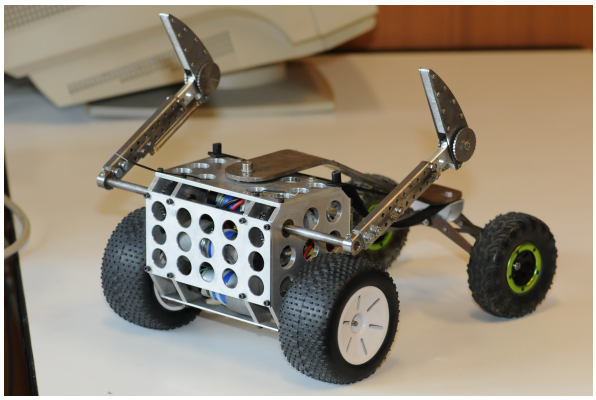


Figure 2.10: Mantis Robot (image kindly provided by Dr. Luca Bruzzone)

A team from researchers at Bonn University - NimbRo Rescue - has developed a hybrid driving-stepping robot - Momaro - presented in Figure 2.11 capable of participating in the DARPA Robotics Challenge. From the papers published by Schwarz et al. [73, 74] the focus will only be on the locomotion of the vehicle, which sets on four legs, each with four DOF and with a double wheel, with two DOF at the end of each leg. The leg has four Dynamixel Pro DC stepper motors for the four DOF, while the wheels' two DOF claimed are achieved through a DC stepper motor inside each wheel. The hardware has a decentralized configuration, each leg uses a dedicated microprocessor, that connects to the central mainboard via Universal Serial Bus (USB), and an E-Stop is used in case of undesirable emergencies. The robot is supplied through Lithium Polymer (LiPo) batteries, in order to be able to last up to 2 hours on a nominal basis, obviously depending on the tasks requested. The robot's base footprint has 0.8 x 0.7 m (L x W), it is considered relatively lightweight, weighing 58 kg, and the legs can extend up to a height of 0.72 m. The servos are controlled through the built-in low-level servo control commands, using PID controllers. The robot is semi-autonomously teleoperated, and the frequency of the data exchange robot $\rightarrow$ operator is 1 Hz, while operator $\rightarrow$ robot is 5 Hz.

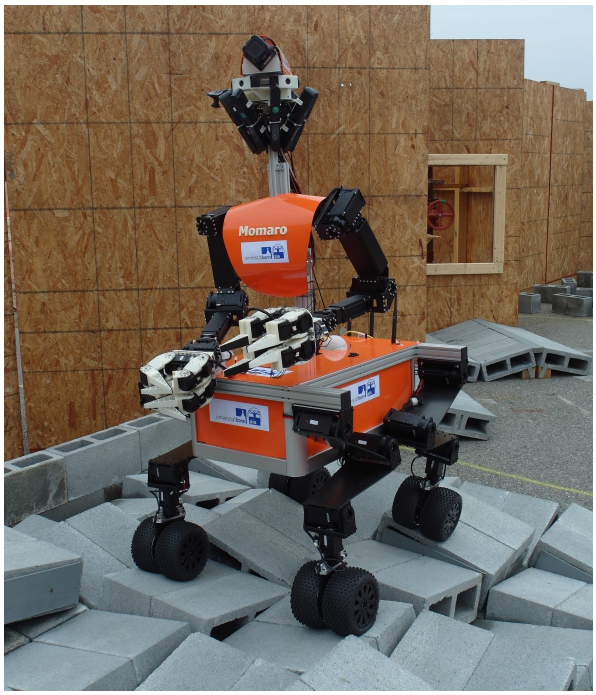


Figure 2.11: Momaro Robot (image kindly provided by Dr. Max Schwarz)

Jehanno et al. [75] have designed a hexapod called Creadapt dedicated to creative adaptation. In this specific context, the leg number is changed and the T-Resilience algorithm is used for the robot to learn how to walk again. The original vehicle has six legs, each one with six DOF, being one of them an actuated wheel on the extremity of the leg. To explore the design trade-offs, the authors developed an algorithm that searches for the best servo-motor/battery configuration to guarantee the best autonomy, weight, leg length, maximum slope and body shape. The resulting configuration is presented in Figure 2.12, uses 30 servo motors, with several pulley-toothed belt system to interconnect the axis of the same leg between them. The robot's final dimensions are 0.58 x 0.35 m (W x L), weight of 7 kg, payload of each leg of 3 kg, maximum wheeled and walking configurations speed of 0.25 and 0.3 m/s, respectively. The servos have in-built PID position controllers and use a communication bus set on RS-485, while the wheel servo firmware was changed to use a speed PID, instead of the original position one. The vehicle uses two Red-Green-Blue-Depth (RGB-D) cameras for 3D visual odometry and an Inertial Measurement Unit (IMU) to estimate its body orientation. The hardware control strategy is centralized, being the processor an embedded computer Intel Atom Z530. The power comes from a 14.8 V/ 8 Ah battery, having an autonomy of 30 minutes.

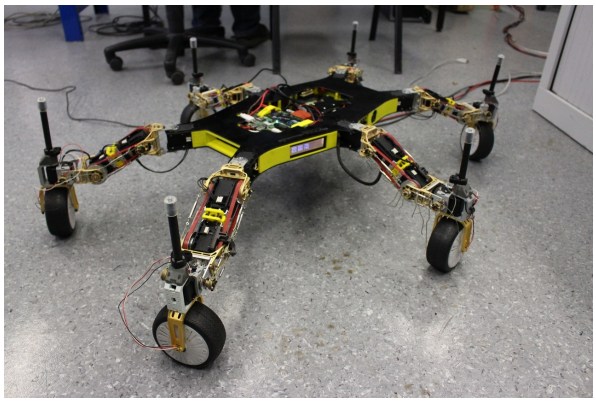


Figure 2.12: Creadapt Robot (image kindly provided by Dr. Jean-Baptiste Mouret)

CENTAURO [76] is a hybrid vehicle, developed in the Instituto Italiano di Tecnologia, with four legs and two arms within a body with 0.61 x 0.615 x 1.706 m (L x W x H). The authors developed their own series elastic actuators (SEA) using brushless DC motors, in order to achieve a modular performance, with a high power/strength density, while protecting the gearbox against impacts and providing torque measurement. The Range of Motion of the legs was designed for the robot to be able to operate in different formations, including inward and outward arrangements. Each leg has 5-DOF combining spider and mammal configurations. The robot weighs 92 kg and works at 48 V. The paper presents 2 studies, one at 0.1 m/s walking and another at 1.6 m/s moving on wheels, while the presented payload is 60 kg. The mentioned vehicle is presented in Figure 2.13.



Figure 2.13: CENTAURO Robot (image kindly provided by Dr. Navvab Kashiri)

ANYmal on Wheels [77,78] is the approach of the Robotic Systems Lab at ETH Zurich to a hybrid legged-wheeled robotic quadruped. The vehicle has four non-steerable, fully torque controlled legs with a total of 16 joints. As its older version - ANYmal [79] - the vehicle has links with approximately

0.25 m of length in the thigh and shank in a total weight of approximately 30 kg, with the size of a medium-sized dog. The onboard batteries of 650 Wh provide the vehicle with power for around 2 hours of autonomous operation. The robot has a simple mechanical topology, using three equal joints per leg, controlled through highly integrated brushless DC SEAs. Their control is done resorting to a compliant Position/Torque Controller, with friction compensation. The wheels are driven by integrated brushless DC motors. The vehicle's top speed while on wheels is 4 m/s and the cost of transport (COT) is considered 0.1 when driving at 2 m/s. Figure 2.14 pictures the current version of the ANYmal on Wheels.

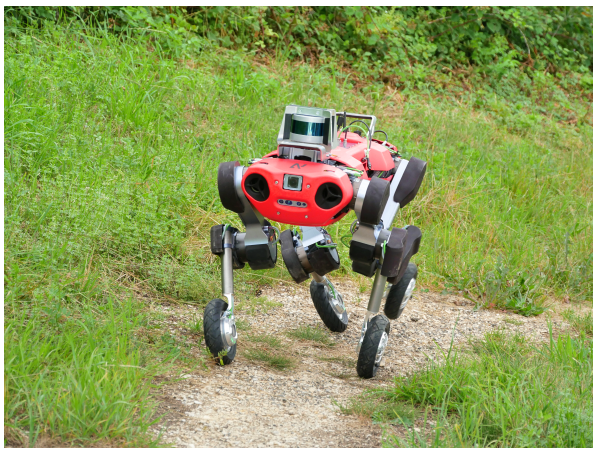


Figure 2.14: ANYmal Wheels Robot (image kindly provided by Dr. Marko Hutter)

The research team from State Key Laboratory of Robotics and System at the Harbin Institute of Technology has developed a biped hydraulic Wheeled-legged Robot, presented in Figure 2.15 - WLR [80] - and its locomotion system will be analyzed. Similar to its newer version, presented in Figure 2.16 - WLR II [81] - the vehicle's leg system with wheels on its extremity has the dimensions of approximately 0.44 x 0.9 m (W x H), the wheels are driven by a DC torque motor without gearbox and the joints use hydraulic driven units, weighing approximately 23 kg. Each of the latter have a flow servo valve, a force sensor and a displacement sensor. They use the provided state measurement (torque and position) and performance (torque and speed). The payload is demonstrated at 7 kg, overcoming steps of 0.1 m of height and slopes of 15 degrees.

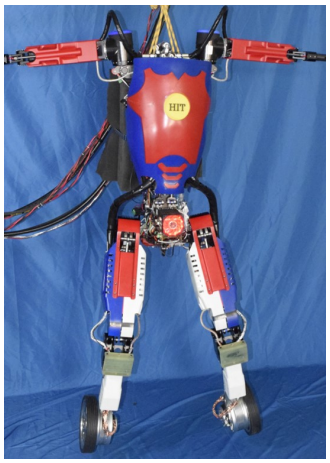


Figure 2.15: WLR Robot (image kindly provided by Dr. Xu Li)

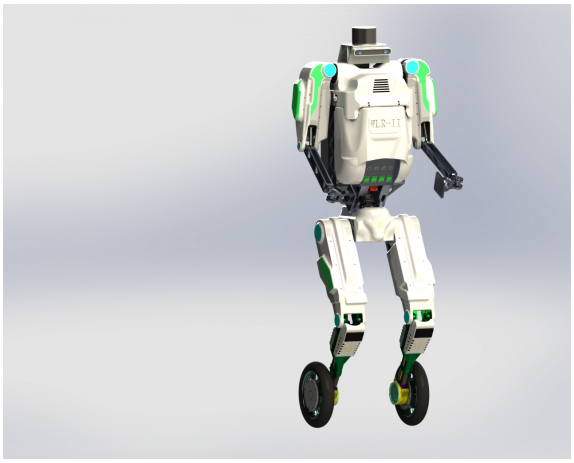


Figure 2.16: WLR-II Robot (image kindly provided by Dr. Xu Li)

Arthur Bouton et al., from the Institute of Intelligent Systems and Robotics at Université Pierre et Marie Curie has continued the work with Hylos and Hylos 2 [82] from Christophe Grand, introducing the newest improved Complios [83]. This vehicle uses a configuration with SEA DC servo motors, springs, steel cables and bilateral pulleys to actuate each one of the legs. In addition it has direct DC motors on the wheels for driving, presented in Figure 2.17. The maximum vehicle dimensions are approximately 0.328 x 0.19 x 0.34 (L x W x H), weighing 15 kg, and its wheels' diameter is 0.15 m. The electronic architecture uses a central single-board computer, running a posture control loop at 40 Hz in a real-time linux kernel. Each leg uses a dedicated microcontroller clocked at 72 Mhz in charge of collecting the sensory information and control the respective motors. Experimentally was

verified that the robot crossed obstacles up to 0.15 m (wheel diameter), overcome 30 degrees slopes, with a speed of 0.1 m/s. Additionally, the authors have tested the vehicle crossing and balancing through 50 degrees slopes with accentuated wheel slips.

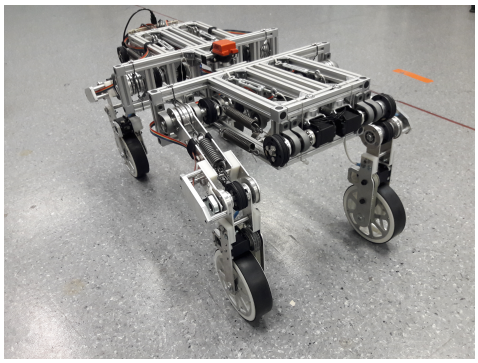


Figure 2.17: Complios Robot (image kindly provided by Dr. Arthur Bouton)

The work from Balchanowski and Gronowicz [84] in the design and simulation of a hybrid Wheeled-Legged Robot has culminated in the construction of a robot [85] that uses the first research work to self-level the chassis and navigate the vehicle through paths with steps. The work, developed at Wroclaw University of Technology, ended up with a vehicle using eight DC linear actuators, capable of moving the platform from the ground up to 0.26 m. The vehicle weighs around 100 kg and its dimensions are around 1.3 x 0.65 x 0.5 m (L x W x H), in a configuration presented in Figure 2.18. The wheels have integrated DC motors, and their rotational movement is achieved through another motor, attached to a belted mechanism.

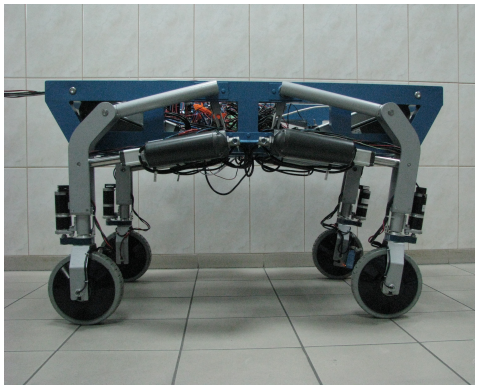


Figure 2.18: Wheel-Legged Mobile Robot (image kindly provided by Dr. Jacek Balchanowski)

2.4 Simulators

2.4.1 Rigid Bodies Realistic Simulators

Realistic simulators are tools capable of detecting rigid bodies interactions through physics engines. These are widely used, given their capability of simulating real-world situations without having to use physical systems. For the simulation, models of the systems are used, which can already be incorporated in the simulator, or defined by the user. The more realistic these models are, the closer to reality the simulation results become. It is noteworthy that this topic has a wide range of simulators for diversified uses. Thus, the focus will be on those that seem most interesting to the scope of the present work.

Gazebo

Gazebo is a Robot Simulator that allows the users to test algorithms, design robots or work with the Simulator Models, and also perform regression tests. It owns multi-robot capabilities and also indoor and outdoor environments.

It has a 3D Graphics Viewer, can work with different types of sensors and be sensitive to noise, and uses different physics engines, including Open Dynamics Engine (ODE), Simbody, Bullet and Dynamic Animation and Robotics Toolkit (DART).

The page dedicated to Gazebo has a set of tutorials to get started with the Simulation environment, as well as a Youtube Channel with videos from simulations *versus* real life tests. A list of the articles written related to the Gazebo platform is also provided.

The software has been developed since 2002 by Dr. Andrew Howard and his student Nate Koenig, at the University of Southern California. It is freeware and it can be runned in Windows, Ubuntu and MacOS.

SimTwo

SimTwo is a realistic robotic simulator, focused on the physic realism while it can be very configurable [86]. This simulator supports several types of robots, being its main purpose the simulation of mobile robots. The dynamic realism is achieved through the decomposition of the systems in rigid bodies, along with definable joints/wheels actuated by electric motors with tunable parameters.

It can be used to virtually simulate the already existing robots or to create a robot with an external design tool and incorporate it in the simulator. The simulator has a 3D Visualizer, the used physics engine is ODE and it is Freeware. It can be used by external programs through serial port and ethernet connections.

This software is being developed since 2008 and it runs on Windows or on Ubuntu (using Wine).

Webots

Webots provides an Integrated Development Environment (IDE) that can be used to model, program and simulate mobile robots. The software allows the user to change the properties of the objects, such as texture, mass, friction, among others. It uses the physics engine ODE, and so it can be used to realistic simulations. It has be co-developed by the Institute of Technology of Lausanne and has almost 20 years of existence.

This software is available for Windows, Linux and MacOS. It has a free limited version and a full paid version.

CoppeliaSim

CoppeliaSim (former V-REP) is a realistic simulator capable of using several physics engines, supports particles simulation, useful to simulate air or water jets, propellers, jet engines, among others.

It can work with different types of sensors, used for path/motion planning, data recording and visualization and supports special mesh edit modes.

There are several videos for helping users to use the tool and for showing several implemented examples.

The software has been developed since 2010. It has three versions and it can be runned in Windows, Ubuntu and MacOS.

2.4.2 Numerical Simulators

MATLAB® - Simulink®

Simulink® provides a multi domain environment for block diagram simulations, code generation and systems verification. It has an integrated graphical editor, supports custom libraries and it has a powerful mathematical solver to model and simulate dynamic systems. Since Simulink® is part of MATLAB®, it can export the results directly to it.

It do not have an integrated physics engine, but its strong point is the dynamic behavior systems simulation, being able to check the results as the simulation is running.

The software is paid, belonging to the MathWorks company.

LabView

LabView is an Integrated Environment specially developed to accelerate the process of development, creation and testing of a system. Since the system is built with blocks, and has a simple graphical-oriented interface, it makes it easier to create a model of our system to implement and to test. It has no associated physics engine or 3D Graphical Visualizer.

The software is paid, belonging to the National Instruments company.

2.5 Conclusion

Throughout this chapter, it was presented a literature review that addresses the multiple actuation areas of this thesis. Some research articles focused on legged robotic platforms were presented, trying to demonstrate the relevance of this type of platforms in mobile robotics. Furthermore, several robotic joints that exist in the literature were addressed, trying to highlight the important aspects mentioned by the article's authors. A slightly deeper study was carried out on legged-wheeled hybrid vehicles, considering that a relevant focus of the present work is researching and developing a hybrid locomotion system. Finally, it was performed a study about the characteristics of existing realistic simulators that allow the simulation of physical systems. Thus, it is demonstrated that the key points related to this thesis were addressed. After considering about the simulator that must be used, the chosen one was SimTwo, given its cross-platform capabilities, realistic systems modeling, previously developed packages and expandability possibilities.

CHAPTER 3

Theoretical Concepts

"No one wants to learn by mistakes, but we cannot learn enough from successes to go beyond the state of the art."
— Henry Petroski

Throughout this thesis, some methods and tools that are applied should be introduced. Thus, the following Chapter is focused on introducing the theoretical concepts that were used during the conducted research.

3.1 Parameter Estimation using Least Squares

Least Squares (LS) [87] is one of the oldest estimation processes, it is widely used because it is simple and does not require *a priori* information about the system.

Considering a function f_β and the system g, defined as:

$$f_\beta(x_t) = \boldsymbol{\phi}(x_t)^T \boldsymbol{\beta} \tag{3.1}$$

where x_t is the state variable.

A regression model can be considered linear when combining linear parameters, as described by Equation(3.2).

$$y(x_t) = \beta_1\phi_1(x_t) + \beta_2\phi_2(x_t) + \cdots + \beta_n\phi_n(x_t) \tag{3.2}$$

Considering matrices $\boldsymbol{\beta}$ and $\boldsymbol{\phi}$ as

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}, \quad \boldsymbol{\phi} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix} \tag{3.3}$$

where n is the system dimension, and that matrices $\mathbf{X}$ and $\mathbf{Y}$ are defined as:

$$\mathbf{X} = \begin{bmatrix} \boldsymbol{\phi}^T(x_1) \\ \boldsymbol{\phi}^T(x_2) \\ \vdots \\ \boldsymbol{\phi}^T(x_N) \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \tag{3.4}$$

where N is the number of samples.

Expanding the system g, Equation (3.5) can be obtained.

$$g(f_\beta(\mathbf{X}), \mathbf{Y}) = \mathbf{Y}^T\mathbf{Y} \text{ - } \mathbf{Y}^T\mathbf{X}\boldsymbol{\beta} - \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{Y} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} \tag{3.5}$$

which can be simplified to:

$$g(f_\beta(\mathbf{X}), \mathbf{Y}) = (\mathbf{Y} \text{ - } \mathbf{X}\boldsymbol{\beta})^T(\mathbf{Y} \text{ - } \mathbf{X}\boldsymbol{\beta}) \tag{3.6}$$

Since $\mathbf{X}^T\mathbf{X}$ is positive semi-defined, there is only one minimum that verifies:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{Y}) \tag{3.7}$$

The estimation can then be calculated with Equation (3.8).

$$\hat{y}_t = \boldsymbol{\phi}(x_t)^T\hat{\boldsymbol{\beta}} \tag{3.8}$$

3.2 Cubic Hermite Splines

Hermite splines consist on weighting four polynomial functions where, for each segment, the required start and end points and its derivatives are given, to obtain a continuous function. Carl de Boor [88] presents a practical guide to understand and implement several kinds of splines, among them the Hermite spline approach.

The most commonly used Hermite spline is the Cubic Hermite Spline. The Hermite Spline formula is applied to each interval separately, obtaining a spline that is continuous and has continuous first derivative.

The Cubic Hermite splines parametrization requires the start and end point, and also the initial and final slope need to be specified in order to obtain a smoothed function. Considering P as a third-degree polynomial, m_i the tangent for the defined point and p_i those defined points, it can be defined that:

$$\begin{aligned} P(0) &= P_0 \\ P(1) &= P_1 \\ m(0) &= m_0 = P'(0) \\ m(1) &= m_1 = P'(1) \\ t &\in [0,1] \end{aligned} \tag{3.9}$$

Considering the latter, the polynomial takes the following form:

$$P(t) = \underbrace{(2t^3 - 3t^2 + 1)}_{h_{00}(t)} P_0 + \underbrace{(t^3 - 2t^2 + t)}_{h_{10}(t)} m_0 + \underbrace{(-2t^3 + 3t^2)}_{h_{01}(t)} P_1 + \underbrace{(t^3 - t^2)}_{h_{11}(t)} m_1 \tag{3.10}$$

For $t' \in [0, t_f]$, $\Delta t = t_f - 0$ and considering the h_{xx} functions, new $P(t')$ become:

$$P(t') = h_{00}(t)P_0 + h_{10}(t)m_0 + h_{01}(t)P_f + h_{11}(t)m_f, \text{ where } t = \frac{t'}{\Delta t} \tag{3.11}$$

The h_{xx} functions can be represented as:

$$\begin{aligned} h_{00}(t) &= (2t^3 - 3t^2 + 1) \\ h_{10}(t) &= (t^3 - 2t^2 + t) \\ h_{01}(t) &= (-2t^3 + 3t^2) \\ h_{11}(t) &= (t^3 - t^2) \end{aligned} \tag{3.12}$$

their derivatives represented as:

$$\begin{aligned} \frac{dh_{00}(t)}{dt} &= (6t^2 - 6t) \\ \frac{dh_{10}(t)}{dt} &= (3t^2 - 4t + 1) \\ \frac{dh_{01}(t)}{dt} &= (-6t^2 + 6t) \\ \frac{dh_{11}(t)}{dt} &= (3t^2 - 2t) \end{aligned} \tag{3.13}$$

and their second derivatives represented as:

$$\begin{aligned}
\frac{d^2h_{00}(t)}{dt^2} &= (12t-6) \\
\frac{d^2h_{10}(t)}{dt^2} &= (6t-4) \\
\frac{d^2h_{01}(t)}{dt^2} &= (-12t+6) \\
\frac{d^2h_{11}(t)}{dt^2} &= (6t-2)
\end{aligned} \tag{3.14}$$

Now, considering that $t \in [0,\Delta]$, the equations for $\theta(t)$, $\omega(t)$ and $\alpha(t)$ becomes:

$$\theta(t) = P(t) = (2t^3-3t^2+1)P_0 + (t^3-2t^2+t)m_0\Delta + (-2t^3+3t^2)P_1 + (t^3-t^2)m_1\Delta \tag{3.15}$$

$$\omega(t) = \frac{dP(t)}{dt} = (6t^2-6t)P_0 + (3t^2-4t+1)m_0\Delta + (-6t^2+6t)P_1 + (3t^2-2t)m_1\Delta \tag{3.16}$$

$$\alpha(t) = \frac{d^2P(t)}{dt^2} = (12t-6)P_0 + (6t-4)m_0\Delta + (-12t+6)P_1 + (6t-2)m_1\Delta \tag{3.17}$$

where P_0, P_1, m_0 and m_1 are the initial and final points and slopes of the spline, respectively.

3.3 Bessel Prototype

There are several ways to choose the closed-loop pole locations. One of them is considering the step response of multiple prototype systems. A prototype that can be applied is Bessel's [89], being the Bessel polynomials known for the roots of the polynomials that guarantee a desirable step response. Considering a $B_k(s)$ the kth-order normalized Bessel polynomial, the transfer function of the system is $B_k(0)$ / $B_k(s)$. All the step responses to the Bessel polynomials present a settling time of 1 second. To obtain different settling times, the roots of the Bessel polynomials must be scaled, dividing them by the settling time.

Table 3.1 present the different polynomial roots for the first six Bessel polynomials.

Table 3.1: Roots of the Bessel polynomials for settling time of 1 second.

k	$B_k(s)$ **Pole location**
1	-4.6200
2	-4.0530 $\pm$ j 2.3400
3	-5.0093 , -3.9668 $\pm$ j3.7845
4	-4.0156 $\pm$ j5.0723, -5.5281 $\pm$ j1.6553
5	-6.4480 , -4.1104 $\pm$ j6.3142, -5.9268 $\pm$ j3.0813
6	-4.2169 $\pm$ j7.5300 , -6.2613 $\pm$ j4.4018, -7.1205 $\pm$ j1.4540

Figure 3.1 presents the step response of a system using the mentioned Bessel prototype.

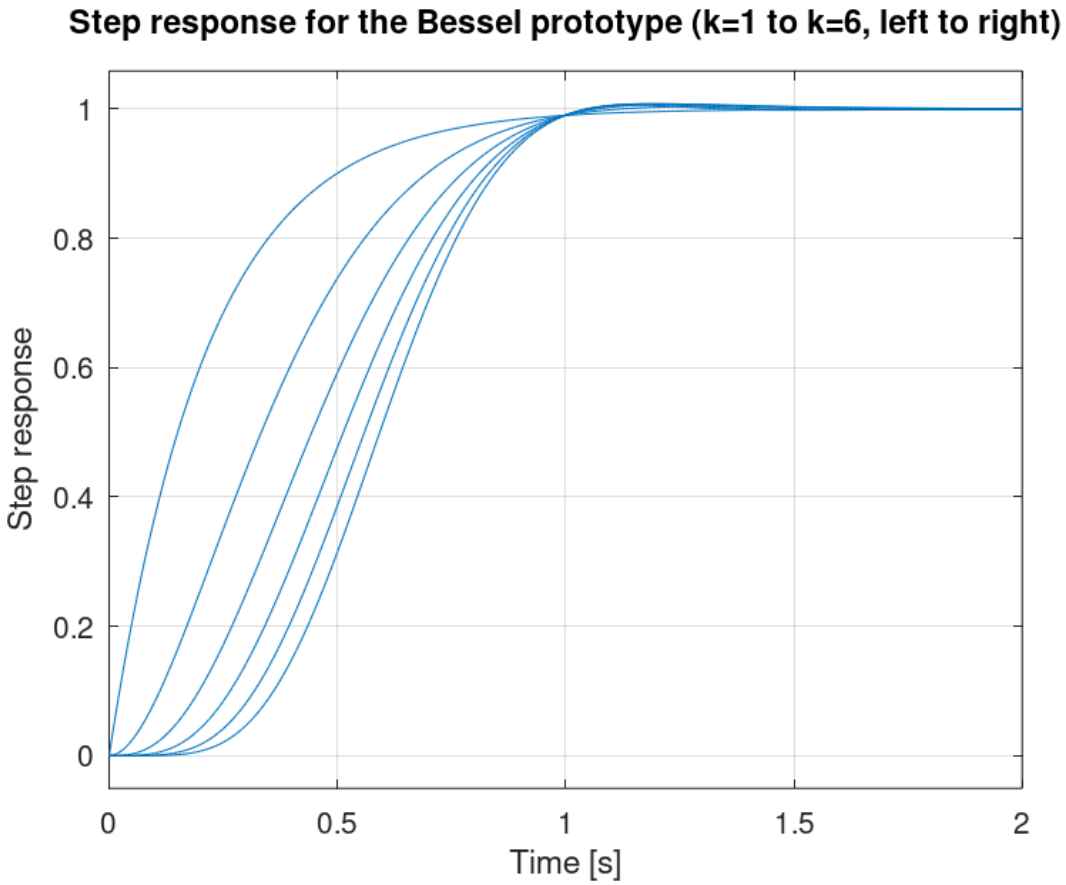


Figure 3.1: Step Response of a System with Bessel Prototype Roots.

3.4 PID Controllers

Classical control techniques have been used successfully for many years. One of the most common approaches is the PID. It calculates an error value $e(t)$ between the current and desired values for a variable, and applies a correction based on proportional, integral and derivative terms [90, 91].

Considering a reference value for a variable $r(t)$, as well as its current measured value $y(t)$, the error can be obtained through the difference between them:

$$e(t) = r(t) - y(t) \tag{3.18}$$

Then, a variable $u(t)$, which is the output of the PID controller is used as input of the system plant, that tries to decrease the error, until it reaches zero.

The controller consists on the weighted sum of the control terms P, I and D, calculated through the following formula:

$$u(t) = K_{\mathrm{p}} e(t) + K_{\mathrm{i}} \int_0^t e(\tau)\,\mathrm{d}\tau + K_{\mathrm{d}} \frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{3.19}$$

where K_p, K_i and K_d are the coefficients from the proportional, integral and derivative terms, respectively.

Figure 3.2 presents the block diagram for a generic PID controller.

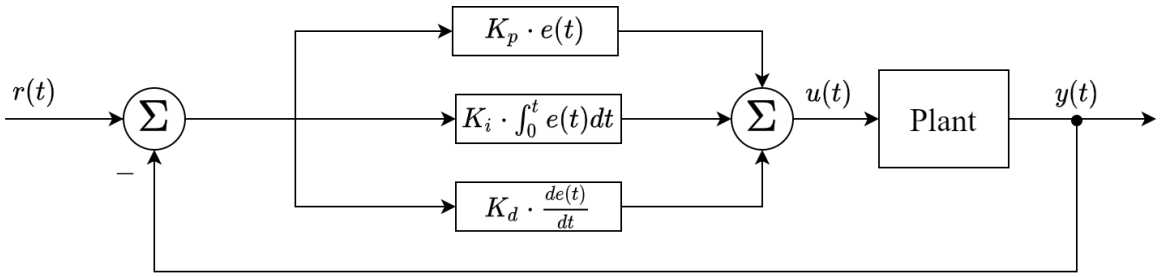


Figure 3.2: Generic PID Controller Block Diagram.

PID Controllers use an approach related to the system's errors, without using the current state. This technique is used by several authors [92, 93], as well as several adaptations [94–96].

Obtaining a realistic model of the system is very useful, allowing several tests to be performed, such as the system limitations, multiple controller matching, speed profiles, among others. Several controllers can be used, from classical PID controllers [97], to more complex approaches [98].

3.4.1 Anti-windup Implementation

In PID controllers, one common issue is the integral windup. When there is a significant change in the reference, the accumulating integral error becomes high. If large enough, at a certain point, it can saturate the system output. Moreover, if the system does not have enough time to reduce the error, the integral term continues to increase. This way, the system overshoots, since the accumulated error is unwound. Anti-windup techniques try to address this problem [99].

Some of the most common used techniques are:

- Limiting the integral term to a minimum/maximum bound;
- Increasing/decreasing the reference point in a suitable ramp;
- Disabling the integral function until the controlled variable reaches a controllable region threshold;
- Feeding back the difference between the saturated and unsaturated control signals.

3.4.2 Feed-forward Block Implementation

The feed-forward control technique is useful where the process variables are known, providing an effective improvement to the classical controllers, anticipating their actions when a change exists in the process variable [100]. The generic feed-forward block *adds* a signal to the result of the PID controller, as illustrated by the block diagram present in Figure 3.3.

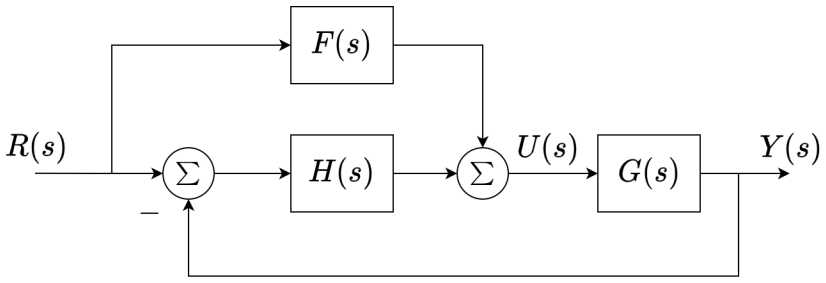


Figure 3.3: Generic Feed-Forward Block Diagram.

Considering the diagram, the following transfer function is obtained:

$$Y(s) = \left(\frac{G(s)H(s)}{1+G(s)H(s)} + \frac{F(s)G(s)}{1+G(s)H(s)} \right) R(s) \tag{3.20}$$

Considering that the optimal is to obtain zero error, Equations (3.21) and (3.22) are deduced:

$$F(s)G(s) = 1 \tag{3.21}$$

$$F(s) = \frac{1}{G(s)} \tag{3.22}$$

As an interesting note, considering a first-order system, in which $G(s) = \frac{k}{\tau s+1}$, then Equation (3.22) becomes:

$$F(s) = \frac{1}{k}(\tau s + 1) \tag{3.23}$$

Now, considering a second-order system, in which $G(s) = \frac{k}{s(\tau s+1)}$, then Equation (3.22) becomes:

$$F(s) = \frac{1}{k}(\tau s^2 + s) \tag{3.24}$$

3.5 State-Space Control Approach

Linear, time invariant systems can be described through differential equations or transfer functions. However, these can also be discretized and described through state-space models. State-space representation presents an alternative methodology to describe a dynamic system, using a compact standard notation. When considering multiple input/output systems, or high complexity systems, state space control approaches are more adequate. This control technique is commonly used for robot control [101–103], and must use an accurate model of the system to be controlled. The concepts explored in this section were extensively covered by Vaccaro [98], in his book dedicated to Digital Control using a state-space approach. The input is represented by function *u(t)* and the output with *y(t)*. The state vector $\mathbf{x}(t)$ (3.25) is a set of n continuous-time functions

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{bmatrix} \tag{3.25}$$

where t represents the time instant of the continuous-time function. On the other hand, $\mathbf{x}[k]$ (3.26) is the same vector, but in discrete-time

$$\mathbf{x}[k] = \begin{bmatrix} x_1[k] \\ \vdots \\ x_n[k] \end{bmatrix} \tag{3.26}$$

where k represents the n-th sample of the discrete-time function.

The state-space model of a continuous-time system uses the first-order differential equations for $\mathbf{x}(t)$, as presented in Equation 3.27, obtaining an output function *y(t)*, presented in Equation 3.28.

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), u(t), t) \tag{3.27}$$

$$y(t) = g(\mathbf{x}(t), u(t), t) \tag{3.28}$$

The function $\mathbf{f}$ is considered linear when $\dot{x}_i(t)$ is expressed through a combination of linear functions of all the state variables and the corresponding input, as Equation (3.29) demonstrates.

$$\dot{x}_i(t) = a_{i\,1}x_1(t) + a_{i\,2}x_2(t) + \cdots + a_{i\,n}x_n(t) + b_i u(t) \tag{3.29}$$

Integrating Equation (3.29), it becomes

$$x_i(t) = x_i(t_0) + \sum_{j=1}^{n} a_{ij} \int_{t_0}^{t} x_j(\tau)\mathrm{d}\tau + b_j \int_{t_0}^{t} u(\tau)\mathrm{d}\tau \tag{3.30}$$

Using Equation (3.30), considering that $x_i(t_0) = 0$, $x_i(t)$ can be obtained through a sum of integrals of state variables and the system input. So, using linear, time-invariant systems, the previous equation can be written in the matrix form as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\,\mathbf{x}(t) + \mathbf{b}\,u(t) \tag{3.31}$$

where

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{3.32}$$

The same principle can be applied to Equation (3.28), obtaining the general form for a nth-order state-space

$$\dot{\mathbf{x}}(t) = \mathbf{A}\,\mathbf{x}(t) + \mathbf{b}\,u(t)$$
$$y(t) = \mathbf{c}\,\mathbf{x}(t) \tag{3.33}$$

where $\mathbf{A}$ represents the $n \times n$ system matrix, $\mathbf{b}$ the $n \times 1$ input vector and $\mathbf{c}$ the $1 \times n$ the output vector.

3.5.1 Obtaining the Transfer Function from the State-Space Model

If it is considered an nth-order system with the transfer function $H(s)$, Equation (3.34) presents the generic function.

$$H(s) = \frac{b_0 s^n + b_1 s^{n-1} + \ldots b_n}{s^n + a_1 s^{n-1} + \cdots + a_n} \tag{3.34}$$

Using Equation (3.33), and assuming zero initial conditions, if a Laplace transform is applied, the continuous-time transfer function can be obtained through:

$$H(s) = \frac{Y(s)}{U(s)} = \mathbf{c}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{b} \tag{3.35}$$

Through Equation (3.35), it is possible to obtain the transfer function of the system only using its state-space representation. Now, since the inverse of a matrix is equal to its adjoint divided by its determinant, $H(s)$ can be re-written as

$$H(s) = \frac{\mathbf{c}\,\text{adj}(s\mathbf{I} - \mathbf{A})\mathbf{b}}{\det(s\mathbf{I} - \mathbf{A})} \tag{3.36}$$

So, it can be deduced that the poles of the transfer function $H(s)$ are equal to the roots of the determinant $\det(s\mathbf{I} - \mathbf{A})$, which means these are equal to the eigenvalues of matrix $\mathbf{A}$.

3.5.2 Discretization of the State-Space Model

Using the state-space model in Equation (3.33) for the plant $G(s)$, Figure 3.4 presents the block diagram

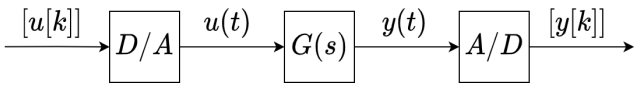


Figure 3.4: Discrete-Time State-Space Model Block Diagram.

To obtain the discrete time model, using a Zero Order Hold (ZOH) equivalent to discretize the system, as proposed by Vaccaro [98], if the value of $x(t)$ is known at some time t_0:

$$\mathbf{x}(t) = e^{\mathbf{A}(t-t_0)}\mathbf{x}(t_0) + \int_{t_0}^{t} e^{\mathbf{A}(t-\tau)}\mathbf{b}u(\tau)\,\mathrm{d}\tau \tag{3.37}$$

where $e^{\mathbf{A}t}$ and $\mathbf{b}$ are defined in Equation (3.32). Equation (3.37) uses $t_0 = kT$ and $t = kT + T$ for the discretization, where T is considered the sampling period, the update equation becomes:

$$\mathbf{x}(kT+T) = e^{\mathbf{A}(kT+T-kT)}\mathbf{x}(kT) + \int_{kT}^{kT+T} e^{\mathbf{A}(kT+T-\tau)}\,\mathbf{b}u(\tau)\,\mathrm{d}\tau \tag{3.38}$$

Since during the interval of integration the function $u(t)$ is equal to $u[k]$, due to the ZOH being constant, Equation (3.38) becomes:

$$\mathbf{x}(kT+T) = \underbrace{e^{\mathbf{A}T}}_{\mathbf{\Phi}}\mathbf{x}(kT) + \underbrace{\left[\int_{kT}^{kT+T} e^{\mathbf{A}(kT+T-\tau)}\mathbf{b}\,\mathrm{d}\tau\right]}_{\mathbf{\Gamma}} u[k] \tag{3.39}$$

$$\mathbf{\Phi} = e^{AT} \tag{3.40}$$

$$\mathbf{\Gamma} = \int_0^T e^{\mathbf{A}\tau}\mathbf{b}\,\mathrm{d}\tau \tag{3.41}$$

Thus, defining a discrete-time state vector $\mathbf{x}[k] = \mathbf{x}(kT)$ and considering the new nomenclature $\mathbf{\Phi}$ and $\mathbf{\Gamma}$, the discrete-time state-space equation becomes:

$$\mathbf{x}[k+1] = \mathbf{\Phi}\mathbf{x}[k] + \mathbf{\Gamma}u[k] \tag{3.42}$$

Using the Van Loan method to compute integrals involving the matrix exponential [104], a matrix $\boldsymbol{M}$ can be used to obtain the $\mathbf{\Phi}$ and $\mathbf{\Gamma}$ ones by computing the obtained $e^{\mathbf{M}T}$ matrix:

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ 0 & 0 \end{bmatrix} \tag{3.43}$$

$$e^{\mathbf{M}T} = \begin{bmatrix} \mathbf{\Phi} & \mathbf{\Gamma} \\ 0 & 1 \end{bmatrix} \tag{3.44}$$

3.6 Controllability of a State-Space Control System

For controlling systems, recurring to the state-space approach, the system must be controllable. This means that it is possible to transfer the system from an arbitrary initial state, to an arbitrary final state in finite time. Considering that the solution to the discrete-time state equations is present in Equation (3.45)

$$\mathbf{x}[j] = \mathbf{\Phi}^j \mathbf{x}[0] + \sum_{i=0}^{j-1} \mathbf{\Phi}^{j-i-1} \mathbf{\Gamma} u[i] \tag{3.45}$$

which expresses $\mathbf{x}[j]$ in terms of $\mathbf{x}[0]$ and the input sequence. Equation (3.45) can be seen as a weighted sum of columns. Thus, the system can be represented in matrix form, shown in Equation (3.46)

$$\begin{bmatrix} x_1[n] \\ \vdots \\ x_n[n] \end{bmatrix} - \mathbf{\Phi}^n \begin{bmatrix} x_1[0] \\ \vdots \\ x_n[0] \end{bmatrix} = \begin{bmatrix} \mathbf{\Gamma} & \mathbf{\Phi\Gamma} & \dots & \mathbf{\Phi}^{n-1}\mathbf{\Gamma} \end{bmatrix} \begin{bmatrix} u[n-1] \\ \vdots \\ u[0] \end{bmatrix} \tag{3.46}$$

where the matrix

$$\mathbf{W}_c = \begin{bmatrix} \mathbf{\Gamma} & \mathbf{\Phi\Gamma} & \dots & \mathbf{\Phi}^{n-1}\mathbf{\Gamma} \end{bmatrix} \tag{3.47}$$

is the controllability matrix for $(\mathbf{\Phi}, \mathbf{\Gamma})$.

3.7 State Control Feedback

The state control feedback is a technique used to define the poles of the closed-loop controller, through the definition of a desired characteristic polynomial. To calculate the state feedback matrix $\mathbf{K}$, the Ackermann's formula [105] can be used. Using the Ackermann's formula, this feedback vector can be calculated using Equation (3.48)

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & \dots & 0 & 1 \end{bmatrix} \mathbf{W}_c^{-1} \Delta_{new}(\boldsymbol{\phi}) \tag{3.48}$$

where $\Delta_{new}(z)$ represents the desired characteristic polynomial, that can be calculated through Equation (3.49), and $\mathbf{W}_c$ the system's controllability matrix, which can be calculated through Equation (3.47).

$$\Delta_{new}(z) = \det(s\mathbf{I} - (\boldsymbol{\phi} - \mathbf{bK})) \tag{3.49}$$

3.8 State Observer

If one does not have access to the system variables, it is impossible to use state-space control. Considering this, a state observer can be used. This is a system that estimates the values of the plant's variables state. When estimating all the variables in the system, the observer is called full-order. If only some state variables are being estimated, is called a reduced-order observer.

A ZOH model of the system's plant can be used to generate an estimation $\hat{\mathbf{x}}[k]$ of the state vector $\mathbf{x}[k]$. However, since in multiple examples the state vector cannot be measured, the plant output $y[k] = \mathbf{c}\,\mathbf{x}[k]$ must be used. Using the same linear combination of the estimated state variables, the following error signal can be generated: $e[k] = \mathbf{c}(\mathbf{x}[k] - \hat{\mathbf{x}}[k])$. To verify if $e[k]$ can be used as a feedback signal, a closed-loop observer can be tested in a plant under digital control, as present in Figure 3.5. In the figure, it can be noted that the input $e[k]$ is multiplied by a $n \times 1$ vector $\mathbf{L}$ with the observer gains.

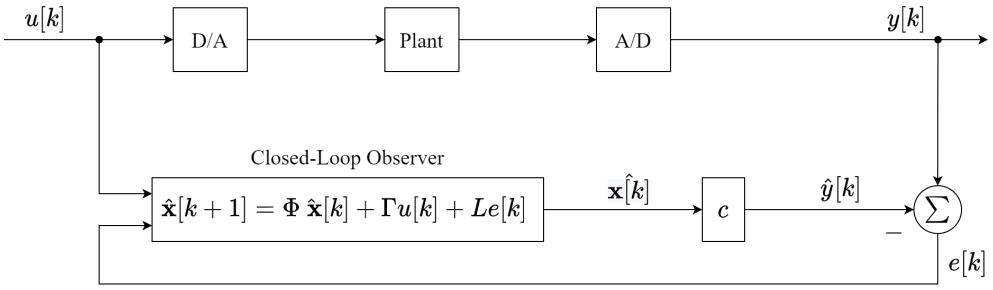


Figure 3.5: Generic Plant with Digital Control and a Closed-Loop Observer.

Figure 3.6 presents an analysis model, obtained through a ZOH equivalent of the generic model illustrated in the block diagram of Figure 3.5.

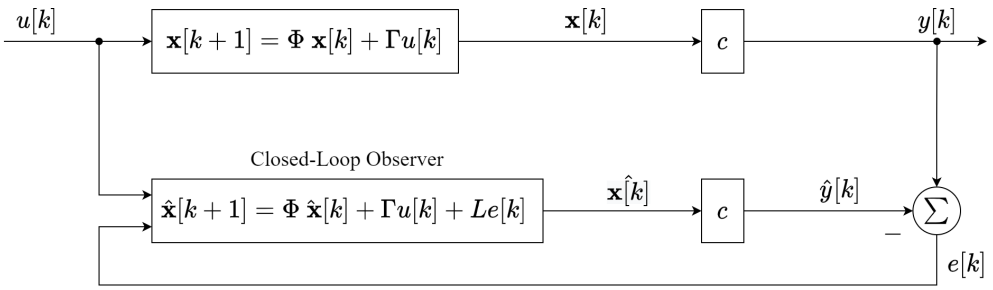


Figure 3.6: State Observer ZOH Equivalent Block Diagram.

The ZOH model of the plant present in Figure 3.6 is defined as:

$$\mathbf{x}[k+1] = \mathbf{\Phi}\mathbf{x}[k] + \mathbf{\Gamma}u[k] \tag{3.50}$$

while the ZOH model of the closed-loop observer is obtained through:

$$\hat{\mathbf{x}}[k+1] = \mathbf{\Phi}\hat{\mathbf{x}}[k] + \mathbf{\Gamma}u[k] + \mathbf{L}\,\mathbf{c}(\mathbf{x}[k] - \hat{\mathbf{x}}[k]) \tag{3.51}$$

To obtain the error, Equations (3.51) and (3.50) can be subtracted, obtaining the error vector:

$$\mathbf{e}[k] = \mathbf{x}[k] - \hat{\mathbf{x}}[k] \tag{3.52}$$

Assuming that $e[0] \neq 0$, one can write a difference equation for $e[k+1]$.

$$e[k+1] = (\mathbf{\Phi} - \mathbf{L}\mathbf{c})^k\, e[k] \tag{3.53}$$

Rewritting Equation (3.51), it becomes a state update equation for the observer with two inputs ($u[k]$ and $y[k]$).

$$\hat{\mathbf{x}}[k+1] = (\mathbf{\Phi} - \mathbf{L}\,\mathbf{c})\hat{\mathbf{x}}[k] + \mathbf{\Gamma}u[k] + \mathbf{L}y[k] \tag{3.54}$$

Considering that $\hat{\mathbf{x}}[0]$ and $\mathbf{x}[0]$ are not estimated or measured, the output for the plant is given by Equation (3.55)

$$y[k] = \mathbf{c}\mathbf{x}[k] \tag{3.55}$$

where $\mathbf{c}$ is a $1 \times n$ coefficient matrix, and $\mathbf{x}[k]$ is the n unknowns state vector. The minimum-norm solution at $k = 0$ is presented in Equation (3.56), while the observer initialization vector is given by Equation (3.57)

$$x[0] = \mathbf{c}^T(\mathbf{c}\mathbf{c}^T)^{-1}y[0] \tag{3.56}$$

$$\hat{\mathbf{x}}[0] = \mathbf{c}^T(\mathbf{c}\mathbf{c}^T)^{-1}y[0] \tag{3.57}$$

In order to find the observer gains, the vector $\mathbf{L}$ needs to be calculated such that the eigenvalues of $(\mathbf{\Phi} - \mathbf{L}\,\mathbf{c})$ are in the specified locations.

$$(\mathbf{\Phi} - \mathbf{L}\,\mathbf{c})^T = \mathbf{\Phi}^T - \mathbf{c}^T\mathbf{L}^T \tag{3.58}$$

Using $\mathbf{\Phi}^T$ and $\mathbf{c}^T$ in the pole placement formula, and solving for the vector $\mathbf{L}^T$, it results in

$$\mathbf{L}^T = \begin{bmatrix}(p_1 - a_1) & \dots & (p_n - a_n)\end{bmatrix} \overline{\mathbf{W}}_c \begin{bmatrix}\mathbf{c}^T & \mathbf{\Phi}^T\mathbf{c}^T & \dots & (\mathbf{\Phi}^T)^{n-1}\mathbf{c}^T\end{bmatrix}^{-1} \tag{3.59}$$

The a_i coefficients are the result of the characteristic polynomial of $\mathbf{\Phi}$, identical to the one of $\mathbf{\Phi}^T$. The p_i coefficients specify the polynomal whose roots are the desired eigenvalues of $(\mathbf{\Phi} - \mathbf{L}\,\mathbf{c})$, the observer poles. $\overline{\mathbf{W}}_c$ is the controllability matrix, and can be calculated using $\mathbf{\Phi}$ and $\mathbf{\Gamma}$. Transposing Equation (3.59), a more useful formula can be obtained to calculate the observer gains vector — $\mathbf{L}$. This is presented in Equation (3.66):

$$\mathbf{L} = \begin{bmatrix}\mathbf{c} \\ \mathbf{c\Phi} \\ \vdots \\ \mathbf{c\Phi}^{n-1}\end{bmatrix}^{-1} \overline{\mathbf{W}}_c^T \begin{bmatrix}p_1 - a_1 \\ p_2 - a_2 \\ \vdots \\ p_n - a_n\end{bmatrix} \tag{3.60}$$

The $\mathbf{L}$ vector can only be calculated if the observability matrix (Equation (3.61)) is invertible. Generally, a n-th order system is observable if the observability matrix $\mathbf{W}_o$ is of rank n.

$$\mathbf{W}_o = \begin{bmatrix}\mathbf{c} \\ \mathbf{c\Phi} \\ \vdots \\ \mathbf{c\Phi}^{n-1}\end{bmatrix} \tag{3.61}$$

However, if $y[k]$ has dimension 1, the Ackermann's formula [105] can also be used to calculate the observer gain matrix $\mathbf{L}$. Using the Ackermann's formula, this vector can be calculated using Equation (3.62)

$$\mathbf{L} = \Delta_{new}(\mathbf{\Phi})\mathbf{W}_o^{-1} \begin{bmatrix}0 \\ 0 \\ \vdots \\ 0 \\ 1\end{bmatrix} \tag{3.62}$$

where $\Delta_{new}(z)$ represents the desired characteristic polynomial, that can be calculated through Equation (3.63), and $\mathbf{W}_o$ the system's observabillity matrix.

$$\Delta_{new}(z) = \det(s\mathbf{I} - (\mathbf{\Phi} - \mathbf{cL})) \tag{3.63}$$

3.8.1 State Observer Algorithm Design

The process of designing the observer with specific pole locations is presented below.

1. Calculate the observability matrix

Considering matrices $\mathbf{c}, \mathbf{\Phi}$ and $\mathbf{\Gamma}$, calculate the observability matrix, using Equation (3.61).

2. Calculate the characteristic polynomial of $\mathbf{\Phi}$ ($\Delta_{new}(z)$)

The roots of the polynomial $\Delta_{new}(z)$ can be obtained using the ZOH pole-mapping formula of the continuous-time plant.

3. Construct the following matrices

Form the matrices presented in Equation(3.64):

$$\overline{\mathbf{\Phi}} = \begin{bmatrix} -a_1 & -a_2 & \dots & -a_n \\ 1 & 0 & \dots & 0 \\ 0 & \vdots & \ddots & \vdots \\ 0 & \dots & 1 & 0 \end{bmatrix} \quad \text{and} \quad \overline{\mathbf{\Gamma}} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{3.64}$$

4. Form the following matrix

Form the matrix present in Equation(3.65):

$$\overline{\mathbf{W}_c} = \begin{bmatrix} \overline{\mathbf{\Gamma}} & \overline{\mathbf{\Phi}}\overline{\mathbf{\Gamma}} & \dots & \overline{\mathbf{\Phi}}^{n-1}\overline{\mathbf{\Gamma}} \end{bmatrix} \tag{3.65}$$

5. Calculate matrix L of the observer gains

Calculate the $\mathbf{L}$ matrix through the formula:

$$\mathbf{L} = \mathbf{W}_o^{-1}\overline{\mathbf{W}}_c^T \begin{bmatrix} p_1 - a_1 \\ \vdots \\ p_n - a_n \end{bmatrix} \tag{3.66}$$

6. Initialize the observer

Using Equation(3.57), initialize the observer.

3.9 Tracking System Design

Tracking systems are important, since it is desired to add a reference input that the system must follow. A tracking system is constantly verifying the output and updating the controller input(s) to *track* the reference signal.

In the present subsection, a tracking system using state-feedback, in a plant consisting on a ZOH discretization of the system to be controlled will be described. This additional block modifies the plant to have a specified zero-input trajectory, and is introduced as part of the compensator. Therefore, it allows to design a system with zero steady-state error. The presence of an additional dynamics is optional, but important, so it will be described.

Defining a composite state vector, that already includes the state vector of the additional dynamics

$$\mathbf{x}_d[k] = \begin{bmatrix} \mathbf{x}[k] \\ \mathbf{x}_a[k] \end{bmatrix} \tag{3.67}$$

since two cascade state-space systems will be used, the description for the plant will be

$$\mathbf{\Phi}_d = \begin{bmatrix} \mathbf{\Phi} & 0 \\ \mathbf{\Gamma}_a\, c & \mathbf{\Phi}_a \end{bmatrix}, \qquad \mathbf{\Gamma}_d = \begin{bmatrix} \mathbf{\Gamma} \\ 0 \end{bmatrix} \tag{3.68}$$

Then, a regulator can be designed and the feedback gains can be obtained as mentioned in Section 3.7, obtaining:

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_1 & \mathbf{K}_2 \end{bmatrix} \tag{3.69}$$

The design plant with the added regulation of the feedback vector is present in Figure 3.7.

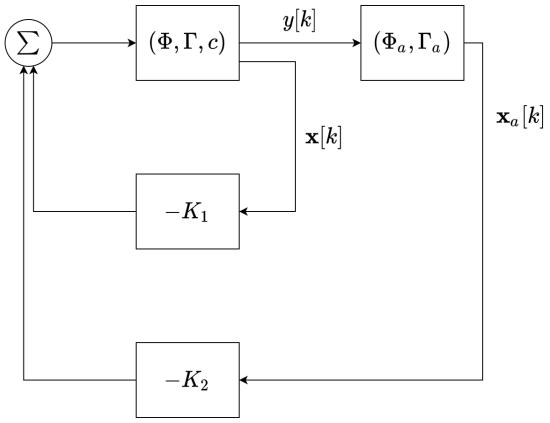


Figure 3.7: Design Plant with Added Feedback Vector.

Considering the obtained plant, the logical way of inputting the reference signals is by comparing them with the plant's output, thus, the diagram in Figure 3.8 can be drawn.

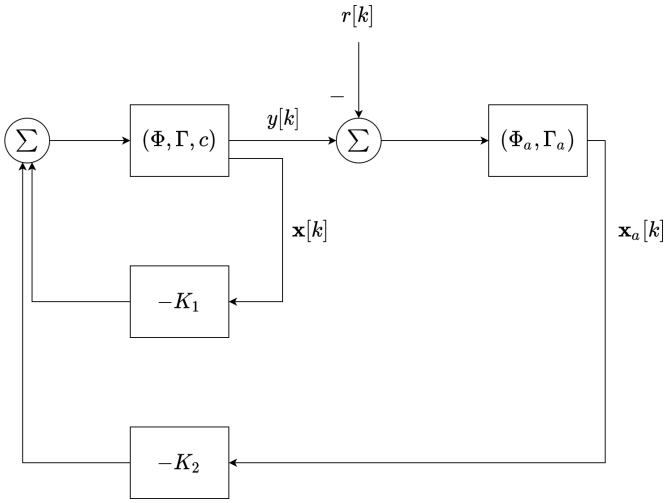


Figure 3.8: Design Plant with Added Feedback Vector and Reference Input.

Using the mentioned plant, by applying some rearrangements, the plant shown in Figure 3.9 can be obtained. It is noteworthy that the system is now more likely the systems studied in classical control theory, due to the negative unity feedback. These systems are known to track the reference with zero steady-state error if the poles of the z-transform of the reference function are contained in the path of the control system. Analyzing the diagram in Figure 3.9, the poles of the plant are modified by $\mathbf{K}_1$ vector. Thus, the poles of the reference appear as the eigenvalues of the additional dynamics. Furthermore, in line with classical control theory, this plant provides robustness to disturbances and

model inaccuracies.

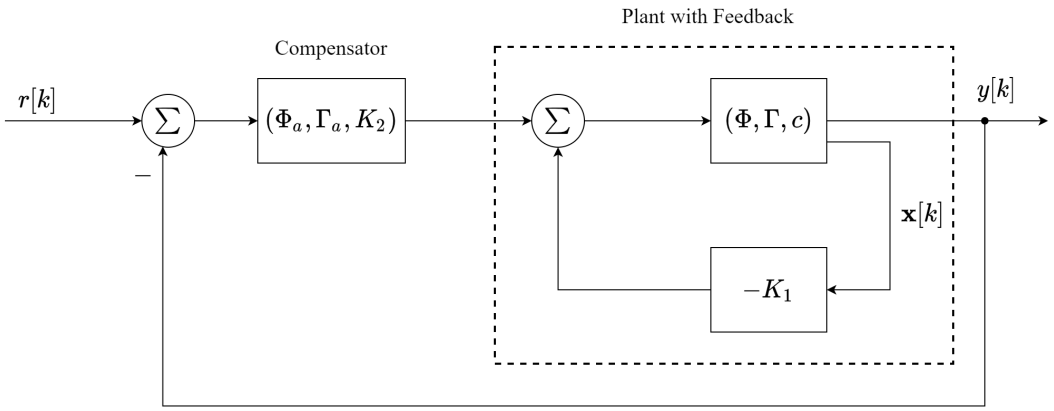


Figure 3.9: Design Plant with Added Feedback Vector and Reference Input - Rearrangement.

The design procedure that will be presented below is a discretization of the method presented by Davison and adapted by Vaccaro [98]. Based on the state-space models presented in Equations (3.70) to (3.73), the design algorithm for the tracking system is presented. The state-space model for the plant is given by

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\,\mathbf{x}(t) + \mathbf{b}\,u(t) \\ y(t) &= \mathbf{c}\,\mathbf{x}(t) + w(t) \end{aligned} \tag{3.70}$$

where $w(t)$ is a disturbance signal than can affect the input or output equations.

The disturbance is assumed to be the output of a homogeneous state-space model

$$\begin{aligned} \dot{\mathbf{z}_d}(t) &= A_d(t) + \mathbf{z}_d(t) \\ w(t) &= \mathbf{c}_d(t) + \mathbf{z}_d(t) \end{aligned} \tag{3.71}$$

For the reference input, it is also assumed to be the output of a homogeneous state-space model

$$\begin{aligned} \dot{\mathbf{z}_r}(t) &= A_r(t) + \mathbf{z}_r(t) \\ r(t) &= \mathbf{c}_r(t) + \mathbf{z}_r(t) \end{aligned} \tag{3.72}$$

The available measurements are considered to be a combination of plant state variables, possibly affected by disturbances

$$\mathbf{y}_m(t) = \mathbf{C}_m\, \mathbf{x}_m(t) + \mathbf{f}_m\, w(t) \tag{3.73}$$

It is considered that the disturbance cannot be measured directly, since otherwise its effects could be compensated. If $\mathbf{f}_m \neq 0$, the disturbance affects the system, so it is assumed that exists a row vector $\mathbf{c}_0$ that satisfies $\mathbf{c}_0\mathbf{C}_m = \mathbf{c}$ and $\mathbf{c}_0\mathbf{f}_m = f$.

3.9.1 Tracking System Design Algorithm

Then, the algorithm for designing the tracking system consists on the procedures explained below:

1. Calculate the necessary additional dynamics

First, it is required to map the eigenvalues sets from A_r and A_d, and set A as the higher multiplicity set of the eigenvalues common to both sets. Then, if A contains eigenvalues λ_i with multiplicity m_i, if their total number is $s = \sum m_i$, then the denominator polynomial of the additional dynamics can be obtained through Equation (3.74), which can be discretized to Equation (3.75).

$$\delta(z) = \prod_i (z - e^{\lambda_i T})^{m_i} \tag{3.74}$$

$$\delta(z) = z^s + \delta_1 z^{s-1} + \cdots + \delta_s \tag{3.75}$$

Defining the state-space matrices for the additional dynamics as

$$\mathbf{\Phi}_a = \begin{bmatrix} -\delta_1 & 1 & 0 & \dots & 0 \\ -\delta_2 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\delta_{s-1} & 0 & 0 & \dots & 1 \\ -\delta_s & 0 & 0 & \dots & 0 \end{bmatrix}, \qquad \mathbf{\Gamma}_a = \begin{bmatrix} -\delta_1 \\ -\delta_2 \\ \vdots \\ -\delta_{s-1} \\ -\delta_s \end{bmatrix} \tag{3.76}$$

2. Calculate a feedback vector for the designed model

The proposed design model consists of a cascade combination of the ZOH of the plant with an additional dynamics, as illustrated by the block diagram in Figure 3.9. Using the state-space models presented in Equations (3.67) and (3.68), the matrices for the combined model are

$$\mathbf{\Phi}_d = \begin{bmatrix} \mathbf{\Phi} & 0 \\ \mathbf{\Gamma}_a\, c & \mathbf{\Phi}_a \end{bmatrix}, \qquad \mathbf{\Gamma}_d = \begin{bmatrix} \mathbf{\Gamma} \\ 0 \end{bmatrix}, \qquad \mathbf{x}_d[k] = \begin{bmatrix} \mathbf{x}[k] \\ \mathbf{x}_a[k] \end{bmatrix} \tag{3.77}$$

with a feedback vector **K**, previously defined in Equation(3.69).

3. Introduce the reference input and implement the additional dynamics as part of the compensator

In this step, only the digital control system is implemented, with the possible use of a state observer, if the full state vector cannot be measured.

3.10 Conclusion

The Least Squares method can be used to estimate the unknown parameters for several models, considering its simplicity and effectiveness. Cubic Hermite Splines can be applied to the motion trajectory planner, taking advantage of its characteristic of creating curves knowing only the intended starting and ending points and their derivatives. The Bessel prototype can help on analyzing the system response to a step input and defining the desired roots for the polynomials. PID controllers are one of the most common control approaches, given their implementation and obtained results. In the context of this thesis, it will be used as a position and speed controllers, using an anti-windup technique and feed-forward block. The state-space approach is very useful for controlling multiple input/output systems, or those with high complexity. However, the model of the system is required *a priori*, as well as all the state variables must be known during the operation. When these are not known, an observer can be used to estimate the unknown state variables. The state-space approach can be very useful to control the non-rigid joints that are used in the developed system, complemented by the use of a state observer.

CHAPTER 4

Design, Modeling and Control of Robotic Joint Systems

"The history of engineering is really the history of breakages, and of learning from those breakages. I was taught at college 'the engineer learns most on the scrapheap'."

— C.A. Claremont, Spanning Space

Along this Chapter some relevant concepts will be detailed, specifically about modeling and control of robotic joint systems. First, a brief introduction on generic modeling and control techniques is presented. After that, an introduction on the specific Geared DC Motor and Rotary Joints are described. Finally, the realistic models of the elements used in this thesis are presented.

4.1 Model of a Physical System

Obtaining a realistic model of the system to be controlled is an advantage when defining and tuning the controller. In addition to being more prompt to decide which control approach is more appropriate, it also serves to obtain the system response to multiple inputs. A more realistic model allows to know, closest to reality, how the system will behave around a certain operating point [106]. Furthermore, these models can be used to simulate realistically the physical system, which can help to reduce controller and software development time, and to avoid hardware damage in early development stages. After estimating the model parameters, multiple tests can be used to validate these models.

Estimation techniques are used to estimate the model parameters. These techniques can be used online or offline, as mentioned by Dupuis et al. [107]. Online parameter estimation can be done using several techniques. Least Squares (LS) is one of the techniques that are widely used [108, 109],

since they are considered efficent and can estimate well in the presence of noise. Recursive Least Squares (RLS) methods are also used [110, 111], when an improved estimation accuracy is required, with an increased processing cost. Other examples of online parameter identification are present in [112, 113], from evolutionary algorithms, such as Montazeri et al. use in [114] to recursive methods as Gonzalez et al. [115] use for their third-order modeling. On offline methods, some interesting examples are [116–118]. Multiple techniques can be applied through the use of neural networks, as explained in [119] or even experimentally, as the authors of [120] and [121] present.

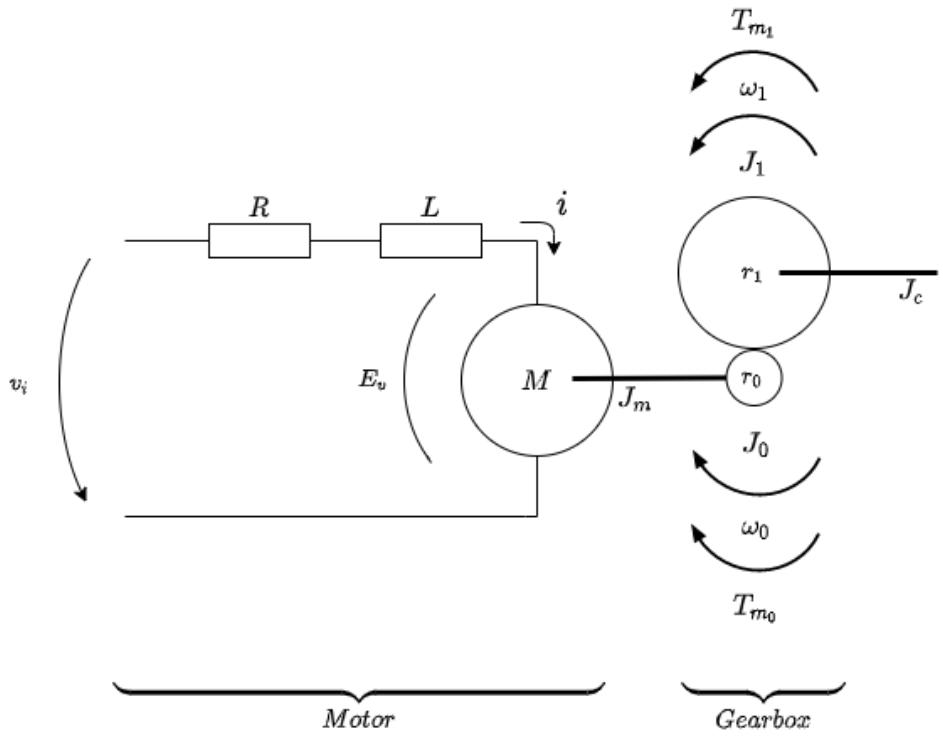
However, online estimation methods are complex and require a significant amount of computational performance. Considering that it is not predictable that either the motor or joints operating conditions change when they are actuated, as well as their essential characteristics are not expected to suffer structural changes, the estimation of its dynamic parameters will be carried out through real hardware tests. For the applied motor model, the considered assumptions were the same ones mentioned by the authors of [117].

4.2 Model of a Geared DC Motor

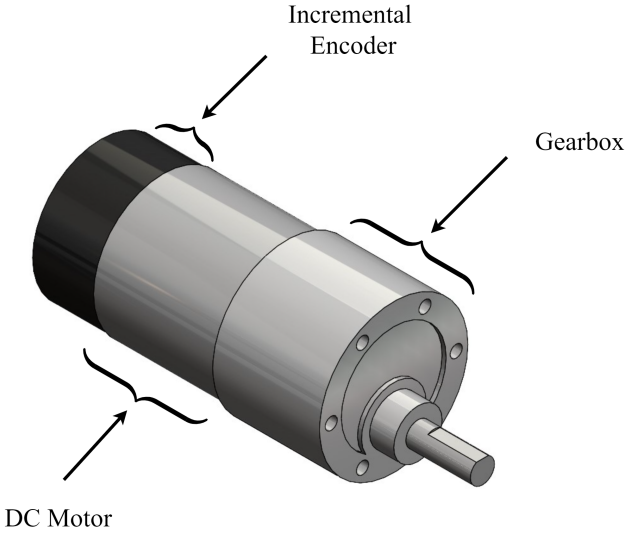
One of the nuclear elements to be used in the designed system is the DC motor. Therefore, it will be described and modeled. An important note is that it was considered that the motor model is equivalent to the motor+gearbox model. It was previously validated that the models are extremely similar.

4.2.1 Non-Linear DC Motor Model

The conceptual representation of the DC Motor is present in Figure 4.1a, while Figure 4.1b illustrates its realistic 3D CAD model. Through this, the general model of a DC motor can be described using Equations (4.1) to (4.4):



(a) Motor — Conceptual Representation.



(b) Motor — 3D CAD Representation.

Figure 4.1: Motor Representation.

$$v_i(t) = R \cdot i(t) + L \cdot \frac{di}{dt} + E_v(t) \tag{4.1}$$

$$E_v(t) = k \cdot \omega_1(t) \tag{4.2}$$

$$\frac{d\omega_1(t)}{dt} = \frac{1}{J} \cdot (T_{m_1}(t) - B_v \cdot \omega_1(t) - T_q) \tag{4.3}$$

$$T_{m_1}(t) = k \cdot i(t) \tag{4.4}$$

where v_i is the voltage applied to the system (V), i is the electrical current (A), R the resistance (Ω), L the inductance (H), and E_v the Back Electromotive Force (V). This force is obtained through Equation (4.2), where k is the motor constant, and ω_1 the angular speed (rad/s). In Equation (4.3), J represents the moment of inertia at the motor's shaft (kg·m^2), which is obtained through Equation (4.5), T_{m_1} the motor torque (N·m), obtained by the fourth equation, B_v the viscous friction coefficient (N·m·s) and T_q (N·m) the static friction torque.

$$J = J_l + J_1 + N^2(J_m + J_0) \tag{4.5}$$

where J_m is the internal moment of inertia from the motor, J_0 and J_1 the gearbox gears input and output moments, J_l the load moment of inertia and N the gearbox reduction, defined as $N = r_1/r_0$, where r_0 and r_1 are the gearbox ratio terms.

The model used for the static friction T_q assumes that the T_q = T_m while $T_m < T_{q_max}$. This explains the existence of the motor dead zone, which is a non-linear element of the model.

4.2.2 Model Linearization

To linearize the system in the motor's deadzone, a Hammerstein block is used. The Hammerstein model is used to represent the nonlinear dead zone, to decrease the complexity of modeling this practical application. This model consists on a nonlinear block, followed by a linear plant block, as represented in Figure 4.2. The use of the Hammerstein block for identification and estimation of nonlinear systems is deeply explored by Eskinat et al. [122].

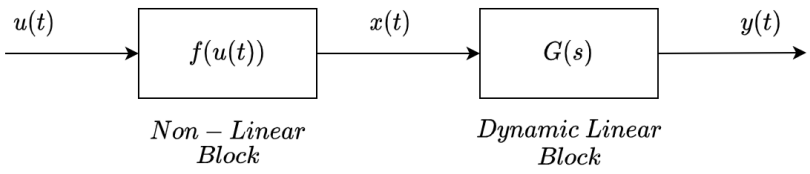


Figure 4.2: Hammerstein Model Block Representation.

As the figure shows, there is a non-linear block, which scales the input $u(t)$, transforming it to $x(t)$. Furthermore, the linear discrete function $G(s)$ models the dynamics, outputing a function $y(t)$. The non-linearity is modeled as an input-dependent gain effect, linearizing at a certain operating point as an instantaneous gain of the system. Some further details on the implementation of this block can be found in Section 4.4.

4.2.3 Linearized DC Motor Model

The final model is used in the Laplace domain, and the system can be represented as:

$$V_I(s) = R \cdot i(s) + s \cdot L \cdot I(s) + k \cdot \omega(s) \tag{4.6}$$

$$J \cdot s \cdot \omega(s) = k \cdot I(s) - B_v \cdot \omega(s) \tag{4.7}$$

The transfer function of the system will then be:

$$\frac{\omega(s)}{V_I(s)} = \frac{k}{(s \cdot J + B_v) \cdot (R + s \cdot L) + k^2} \tag{4.8}$$

To facilitate its later use in the estimation of some parameters, the previous equation becomes:

$$\frac{\omega(s)}{V_I(s)} = \frac{\frac{k}{J \cdot L}}{s^2 + s\left(\frac{R}{L} + \frac{B_v}{J}\right) + \left(\frac{R \cdot B_v + k^2}{J \cdot L}\right)} \tag{4.9}$$

Since the pole associated with the impedance L is a fast pole, its effect can be neglected. Therefore, the value of L can be set to 0, as shown in Equation (4.10). So, Equations (4.8) and (4.9) can be simplified to Equations (4.11) and (4.12).

$$\frac{\omega(s)}{V_I(s)} = \frac{k}{(s \cdot J + B_v) \cdot (R + \cancel{s \cdot L}) + k^2} \tag{4.10}$$

$$\frac{\omega(s)}{V_I(s)} = \frac{k}{(s \cdot J + B_v) \cdot R + k^2} \tag{4.11}$$

$$\frac{\omega(s)}{V_I(s)} = \frac{k}{s \cdot J \cdot R + B_v \cdot R + k^2} \tag{4.12}$$

Therefore, Equation (4.1) can be simplified to Equation (4.13), further arranged to Equation (4.14). These will be used, along with some graphics obtained experimentally to withdraw some conclusions about the motor parameters.

$$u(t) = R \cdot i(t) + k\omega(t) \tag{4.13}$$

$$i(t) = \frac{u(t) - k\omega(t)}{R} \tag{4.14}$$

4.2.4 Geared DC Motor

With the goal of increasing the vehicle's autonomy, using SEA, it was decided to use DC motors with gearbox. After some tests, which were published in article [13], preceded by [14], it was found to be more convenient to use in-line gearboxes, since the use of self-locking gearboxes increases the shaft's static friction. The used motors were those from CQRobot, being decided to use different gearboxes. Concerning the model parameter estimation, it is only relevant to define the one used for the non-rigid joint — CQRGL20180525-52GB — with ratio of 210:1 and top speed of 52 RPM (≈5.45 rad/s). The motor is present in Figure 4.3. As previously mentioned, the model mostly differs by a scale factor, introduced by the gearbox.

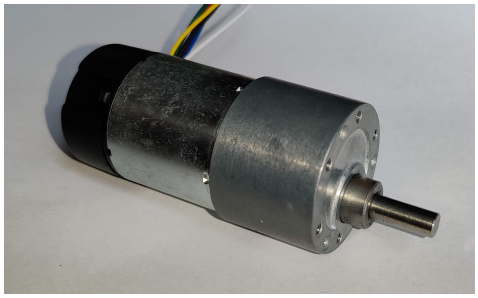


Figure 4.3: Used Geared DC Motor — Real Version.

4.2.5 DC Motor Model Parameters Estimation

To obtain the parameters referring to the used motor, it is necessary to collect the data and group them in a graphic form. In order to obtain the value of the motor constant k, and its resistance R, the graphs to be evaluated are the angular speed over voltage, as well as $\frac{current}{angular\ speed}$ over $\frac{voltage}{angular\ speed}$. From the graphic of ($k\times$current) over angular speed, the values of the motor viscous friction coefficient and the static friction torque can be obtained.

The physical test that allows to estimate some unknown motor parameters is described:

Powering the motor with multiple pre-known values of supply voltage, allowing it to reach the steady-state - This test consists in powering the motor with several different known supply voltages,

from 0.5 V to 12 V and measure the steady-state angular speed ω. With this information, several graphical representations can be used to extract the motor parameters.

Figures 4.4 to 4.6 present the collected data, which allow the estimation of the motor model parameters. The performed tests were carried out with the planetary reduction gear coupled to the motor shaft. This way, the gear effects are incorporated in the motor model, instead of the joint model.

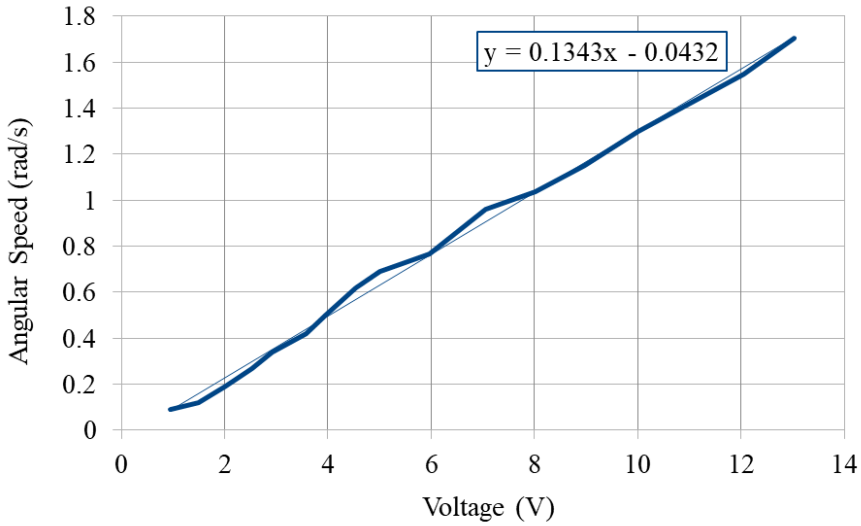


Figure 4.4: Motor Angular Speed vs. Voltage.

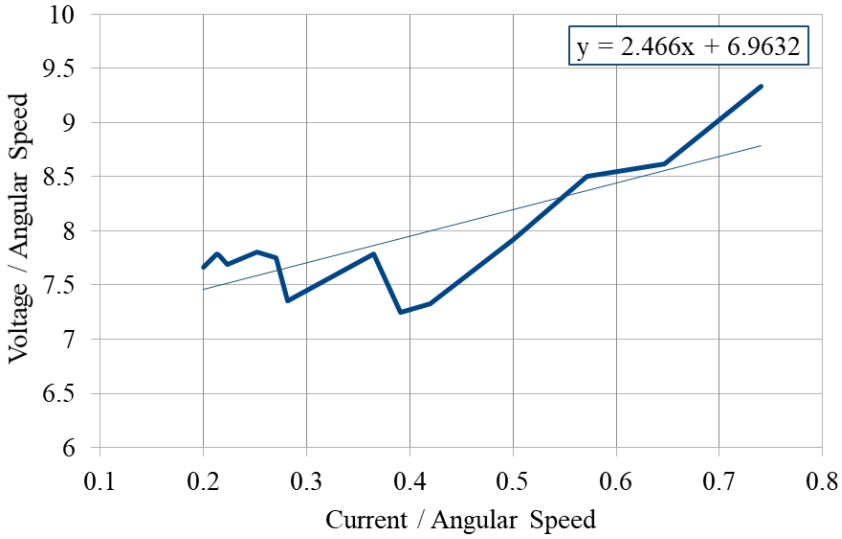


Figure 4.5: Motor $\frac{Voltage}{AngularSpeed}$ vs. $\frac{Current}{AngularSpeed}$.

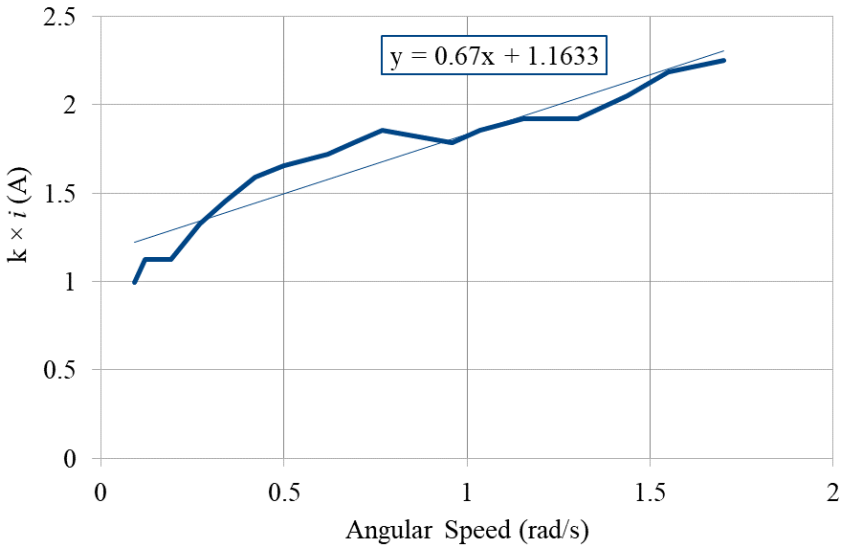


Figure 4.6: Motor Angular Speed vs. $k \times$ Current.

The obtained estimated parameters for the motor are presented in Table 4.1.

Table 4.1: Estimated Motor Parameters.

Parameters	
R	2.47 Ω
k	6.62
T_q	1.16 N·m
B_v	0.67 N·m·s

4.3 Design and Model Robotic Rotary Joints

Another core element of the designed system is the rotary joint. There are two configurations used in the system — rigid and non-rigid. Their design, models and respective parameter estimation will be presented below.

4.3.1 Rigid Joint

The rigid joint immediately transmits the rotational force from its input to the output. The proposed model for the rigid and non-rigid joints is not the same. This subsection presents the details about the rigid joint.

4.3.1.1 Model

For the rigid joint, the conceptual representation shown in Figure 4.7 is applied, and originates Equation (4.15).

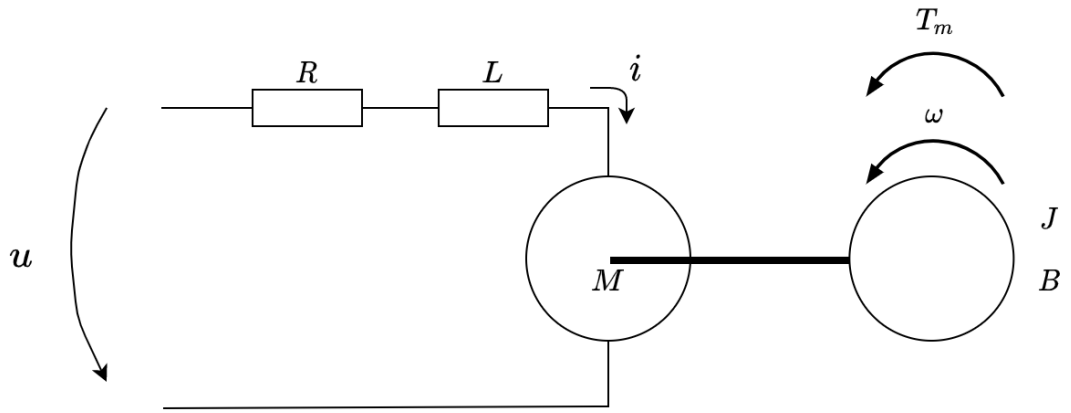


Figure 4.7: Rigid Joint Conceptual Representation.

$$J\dot{\omega} = ki - B\omega \tag{4.15}$$

Considering i as defined in Equation (4.14), and rearranging Equation (4.15) to (4.16):

$$J\dot{\omega} = \frac{k}{R}u - \frac{k^2}{R}\omega - B\omega \tag{4.16}$$

The generic model for this joint is presented below, having considered the above equations to develop the matrix system.

$$\dot{\theta} = \omega \tag{4.17}$$

$$\dot{\omega} = \frac{1}{J}\left(\frac{k}{R}u - \left(B + \frac{k^2}{R}\right)\omega\right) \tag{4.18}$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \omega \end{bmatrix} \tag{4.19}$$

$$\mathbf{X} = \begin{bmatrix} \theta \\ \omega \end{bmatrix} \tag{4.20}$$

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{\left(B+\frac{k^2}{R}\right)}{J} \end{bmatrix} \mathbf{X} + \begin{bmatrix} 0 \\ \frac{k}{JR} \end{bmatrix} u \tag{4.21}$$

4.3.1.2 Prototype Design

One of the purposes of the applied leg configuration is that it can rotate at least $\pm$ 90º, allowing purely linear lateral movement. In addition, the wheel's support point should be aligned with the rotation axis of the hip joint. Otherwise, when the leg rotates, the wheel would drag on the ground. This joint was also suffering from torsional moment, and could be damaged for this reason. This is due to being the only exactly vertical axis, therefore more prone to impacts. Furthermore, this is also the joint responsible for steering the vehicle when it is moving, so it is extremely important that its positioning is as accurate as possible. To improve the mechanical robustness of this joint, as well as to decrease the gearbox backlash, it was added a reduction gear. This has a ratio of 4:1, connected through a geared belt, that improves both specifications. Figure 4.8 represents the mechanical explosion of the same joint. The obtained rigid hip joint is the one present in Figure 4.9.

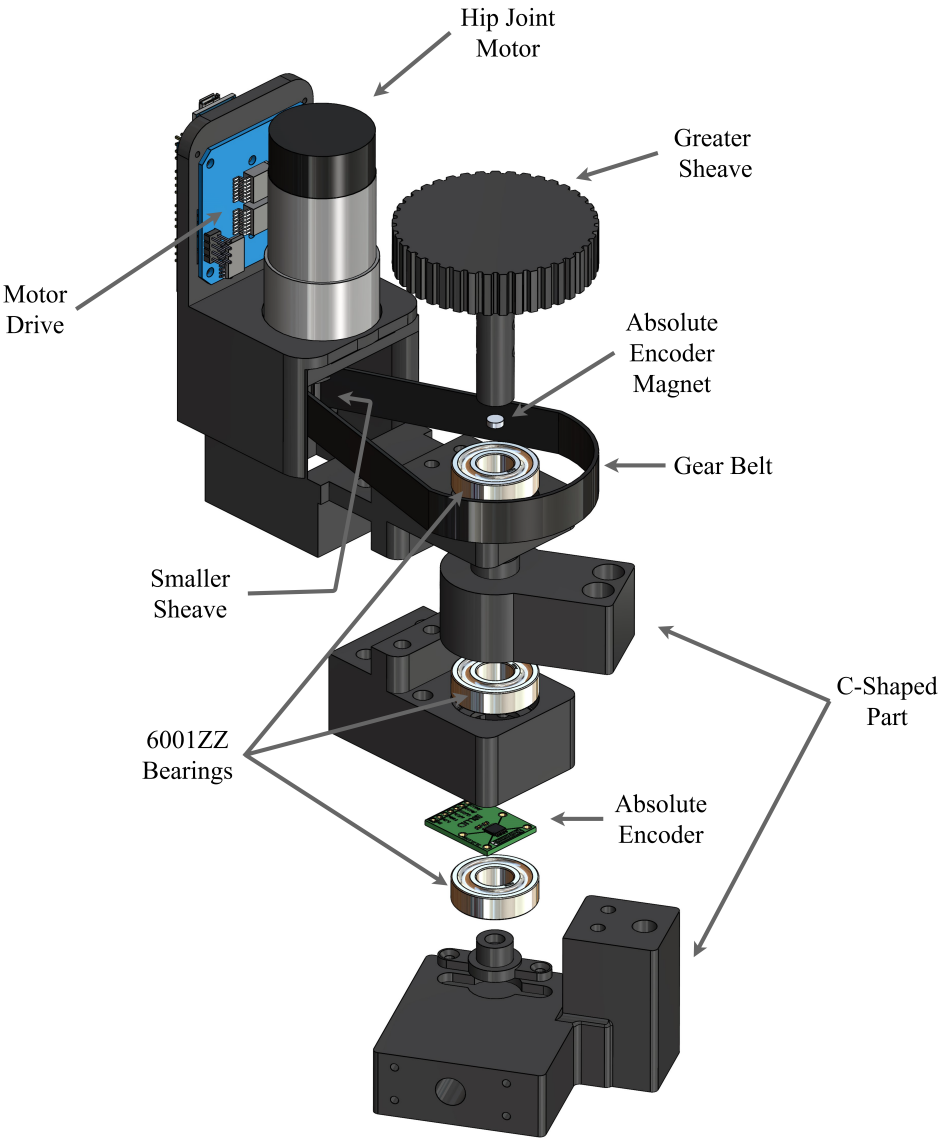


Figure 4.8: Improved Hip Rigid Joint (mechanical explosion).

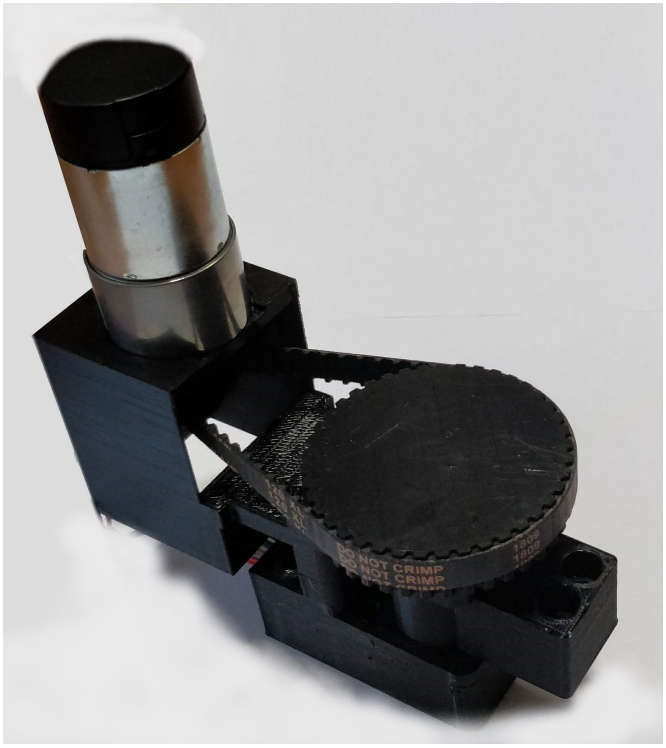


Figure 4.9: Hip Rigid Joint.

4.3.1.3 Prototype Model Parameter Estimation

Taking Equations (4.20) and (4.21) and considering a centered finite difference discretization:

$$\dot{\omega} \approx \frac{\omega_{k+1} - \omega_{k-1}}{2\Delta t} \tag{4.22}$$

$$\dot{\omega} = \underbrace{\left(\frac{B + \frac{k^2}{R}}{J}\right)}_{\beta_1} \omega + \underbrace{\frac{k}{R}}_{\beta_2} u \tag{4.23}$$

The matrices $\mathbf{Y}$, $\mathbf{X}$ and $\boldsymbol{\beta}$ can be obtained through the definitions in Equation (4.24). These use $\dot{\omega}_i$, ω_i, and u_i representing the corresponding variable's i-th sample. To estimate the joint's model

parameters, N samples are used.

$$\mathbf{Y} = \begin{bmatrix} \dot{\omega}_1 \\ \vdots \\ \dot{\omega}_N \end{bmatrix} \quad , \quad \mathbf{X} = \begin{bmatrix} \omega_1 & u_1 \\ \vdots & \vdots \\ \omega_N & u_N \end{bmatrix} \quad \text{, and} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} \tag{4.24}$$

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} \tag{4.25}$$

$$\begin{bmatrix} \dot{\omega}_1 \\ \vdots \\ \dot{\omega}_N \end{bmatrix} = \begin{bmatrix} \omega_1 & u_1 \\ \vdots & \vdots \\ \omega_N & u_N \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} \tag{4.26}$$

Therefore, to estimate matrix $\boldsymbol{\beta}$, the following model can be used, previously mentioned in Equation (3.7):

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{Y}) \tag{4.27}$$

To estimate the parameters regarding the joint model, which allows to obtain parameters J, k and B, physical tests must be performed. Considering that the joint is rigid, the performed test was:

Applying a sweep sinusoidal signal - A sinusoidal signal is applied with a constant amplitude but with a variable frequency and maximum time. This option makes it possible to excite the system in different regimes, improving the model realism. The whole process is parameterized to allow an excitation of the system in multiple modes.

For this estimation, it was used a sinusoidal sweep with an amplitude of 4.8 V, starting frequency of 0.2 Hz, final frequency of 2 Hz, and a total time of 20 s, as ilustrated in Figure 4.10.

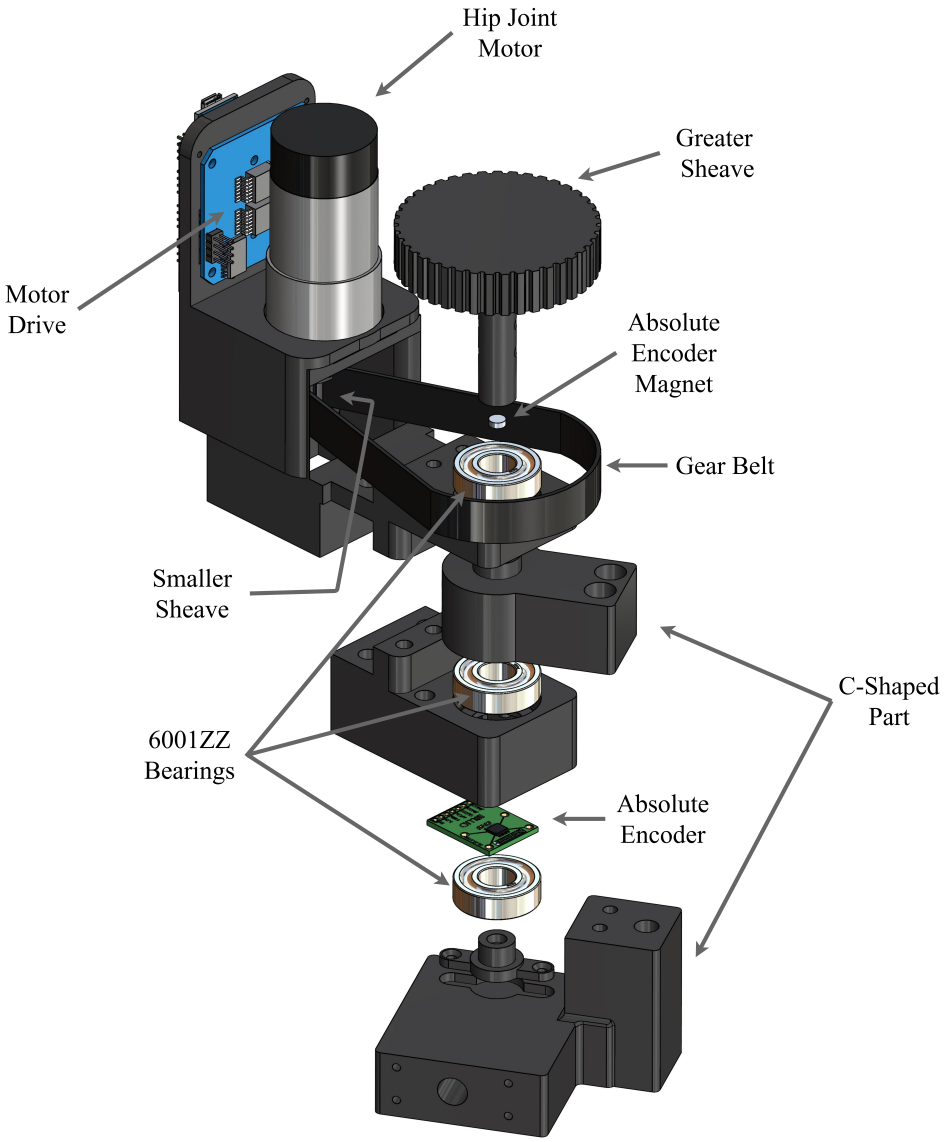


Figure 4.10: Sinusoidal Sweep Signal.

The parameters obtained after the estimation are the ones presented in Equation 4.28.

$$\begin{bmatrix} \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1.0000 \\ 0 & -16.5273 \end{bmatrix} \begin{bmatrix} \theta \\ \omega \end{bmatrix} + \begin{bmatrix} 0 \\ 8.3570 \end{bmatrix} u \tag{4.28}$$

Given this model, the obtained transfer function is then presented.

$$G(s) = \frac{8.357}{s^2 + 16.53s + 7.5} \tag{4.29}$$

The sampling period used to obtain the values presented in the matrices of Equation 4.28 was 10 ms.

The presented model originated the following result, which can be compared to the data obtained in the test through the graphic on Figure 4.11.

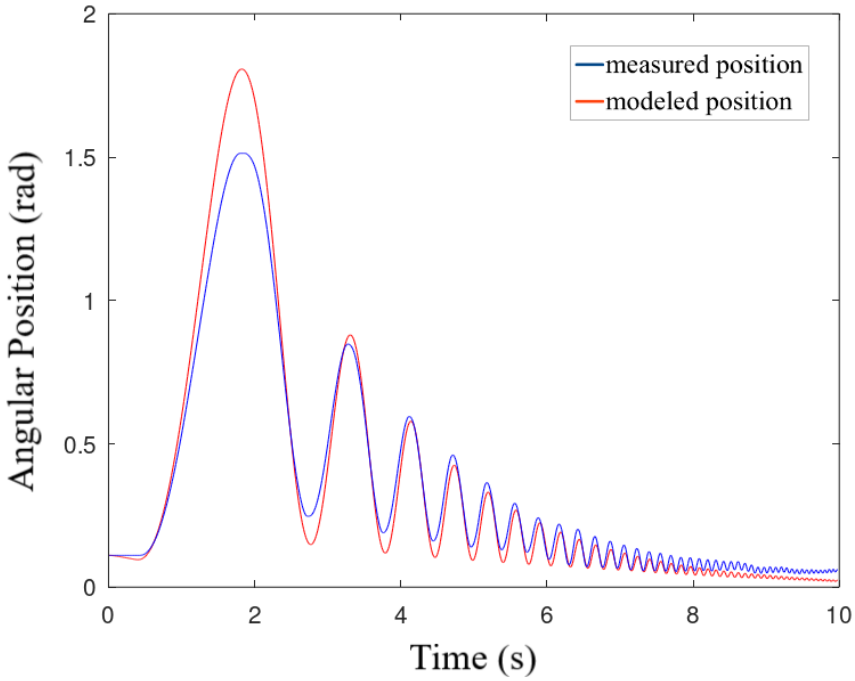


Figure 4.11: Rigid Joint Model vs. Real Angular Positions.

This modeling allowed to develop controllers so that its performance can be enhanced. There were performed three tests, with angular position differences of 0.5, 1 and 1.5 rad, obtaining a step response of this reference variation. The result of the controller applied to the physical joint is illustrated in Figures 4.12, 4.13 and 4.14:

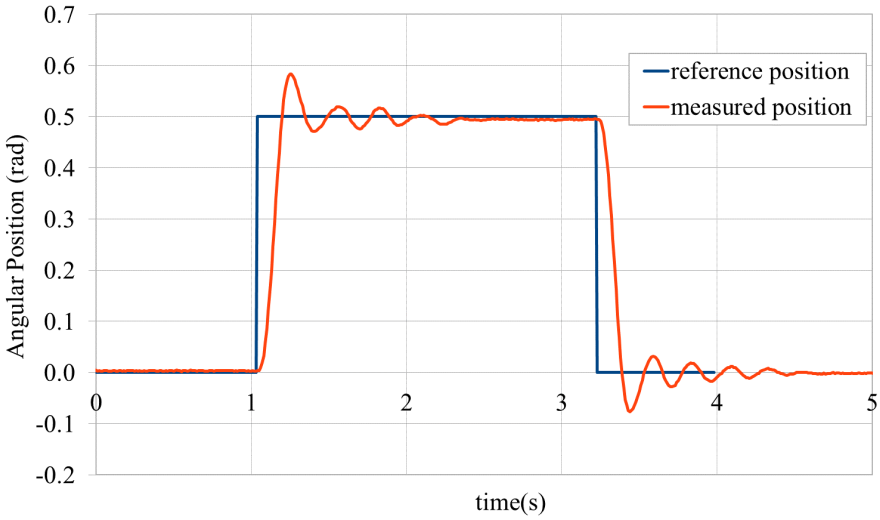


Figure 4.12: Step response Rigid Joint (θ_{ref} = 0.5 rad).

Considering a reference of 0.5 rad, and with an initial position of 0 rad, the obtained overshoot was 14.8%, the rise time 0.09 s and the settling time 0.51 s. In the downward direction, the overshoot was 14.2%, the rise time was 0.09 s and the settling time was 0.41 s.

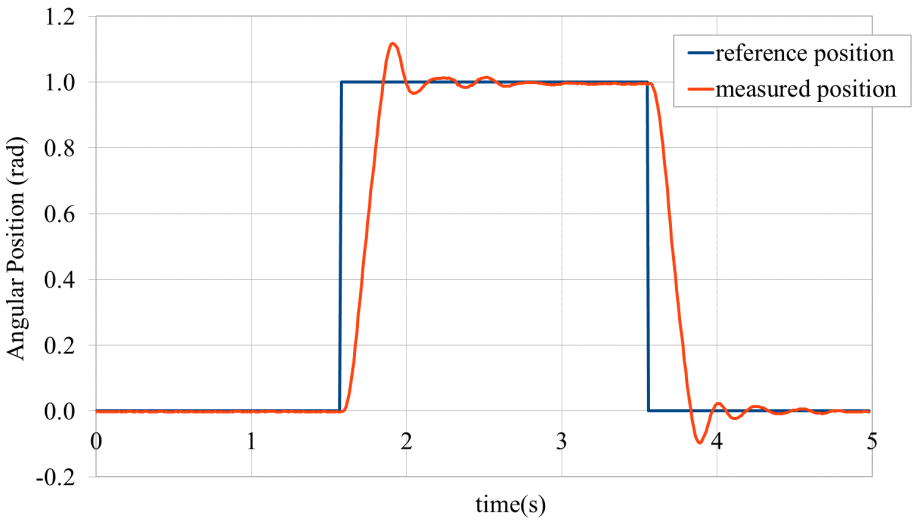


Figure 4.13: Step response Rigid Joint (θ_{ref} = 1 rad).

Considering a reference of 1 rad, and with an initial position of 0 rad, the obtained overshoot was 10.3%, the rise time 0.18 s and the settling time 0.52 s. In the downward direction, the overshoot was 9.5%, the rise time was 0.18 s and the settling time was 0.39 s.

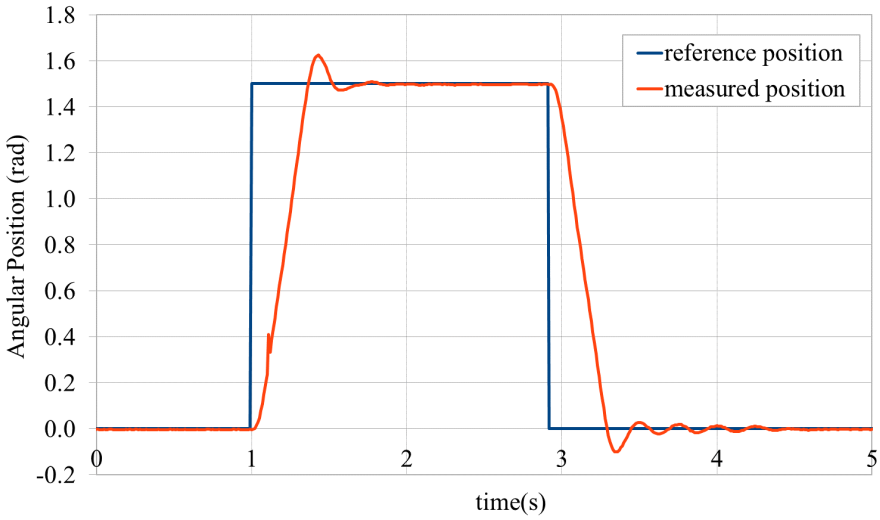


Figure 4.14: Step response Rigid Joint (θ_{ref} = 1.5 rad).

Considering a reference of 1.5 rad, and with an initial position of 0 rad, the obtained overshoot was 7.3%, the rise time 0.25 s and the settling time 0.50 s. In the downward direction, the overshoot was 6.8%, the rise time was 0.27 s and the settling time was 0.71 s.

It is transverse the presence of an oscillation around the reference. It could be detected that this effect is due to the deadzone of the gearbox coupled to the motor, which causes this effect.

4.3.2 Non-Rigid Joint

The non-rigid joint uses a spring-damper system to transmit the rotational force from its input to the output. The models are different from those presented in the previous subsection, since now these includes non-linear elements. The following subsection presents the details about the non-rigid joint.

4.3.2.1 Model

Regarding the non-rigid joint, the respective conceptual representation is presented in Figure 4.15. The linear differential equations that represent it are presented below.

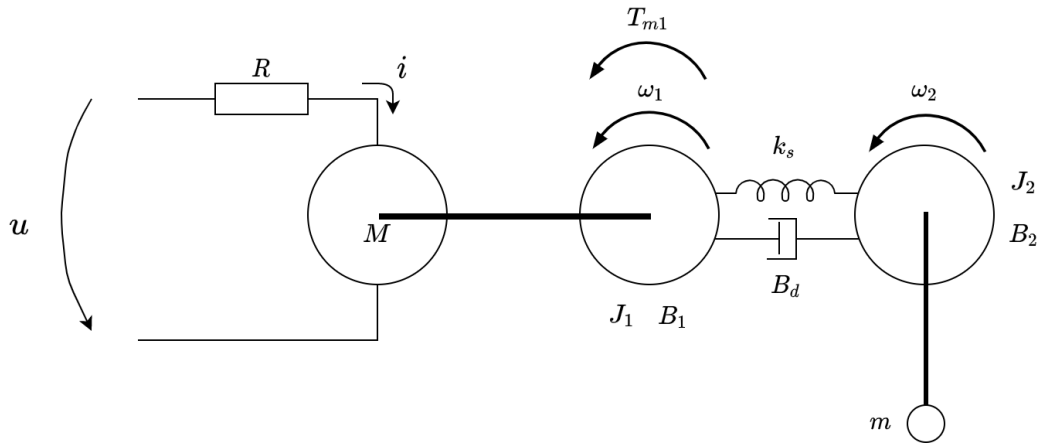


Figure 4.15: Non-Rigid Joint Conceptual Representation.

The spring induced torque can be modeled by the spring constant (k_s), multiplied by the difference between the input (θ_1) and output (θ_2) angular positions.

$$\tau_K = k_s \cdot (\theta_2 - \theta_1) \tag{4.30}$$

The viscous induced torque can be represented as the damping coefficient (B_d), multiplied by the difference between the input and output angular speeds.

$$\tau_B = B_d \cdot \left(\frac{d\theta_2(t)}{dt} - \frac{d\theta_1(t)}{dt} \right) \tag{4.31}$$

During the performed experiments, the input angle is fixed, θ_1 and ω_1 are equal to zero. So, the system angular acceleration over time can be obtained through:

$$\frac{d\omega_2(t)}{dt} = \frac{1}{J_2}(-k_s\theta_2 - B_d\omega_2) \tag{4.32}$$

Using only the derivatives of θ_2, the previous equation becomes:

$$J_2 \cdot \frac{d^2\theta_2(t)}{dt} + B_d \cdot \frac{d\theta_2(t)}{dt} + k_s \cdot \theta_2(t) = 0 \tag{4.33}$$

In the Laplace domain, the system can be represented as:

$$J_2 \cdot (s^2\theta_2(s) - s\theta_2(t=0) - \frac{d\theta_2}{dt}(t=0)) + B_d \cdot (s\theta_2(s) - \theta_2(t=0)) + k_s \cdot \theta_2(s) = 0 \tag{4.34}$$

Given that the initial conditions might not be null, the next variables take the initial values

$$\theta_2\ (t=0) = \theta_0 \tag{4.35}$$

$$\frac{d\theta_2\ (t=0)}{dt} = \omega_2\ (t=0) = 0 \tag{4.36}$$

Thus, Equation (4.34) will be updated to

$$J_2 s^2 \theta_2(s) - J_2 s\theta_0 + B_d s\theta_2(s) - B_d\theta_0 + k_s\theta_2(s) = 0 \tag{4.37}$$

Simplifying and manipulating Equation (4.37), the following is obtained:

$$\theta_2(s) = \frac{\theta_0 \cdot (J_2 s + B_d)}{J_2 s^2 + B_d s + k_s} \tag{4.38}$$

$$\theta_2(s) = \theta_0 \cdot \frac{\left(s + \frac{B_d}{J_2}\right)}{s^2 + \left(\frac{B_d}{J_2}\right) s + \left(\frac{k_s}{J_2}\right)} \tag{4.39}$$

Regarding the physical tests, in addition to the the test mentioned in section 4.3.1.3, another can be performed to help model the non-rigid joint:

Pulling the link attached to the non-rigid joint and releasing it, obtaining the angular speed and the angle until it stabilizes - The link attached to the non-rigid joint is pulled manually and, after reaching a pre-defined angle, is released and left to stabilize until it reaches the rest angle. The angular speed, torque and the angle are sampled with a period of 10 ms.

The test to obtain the data to estimate some important joint parameters is performed while θ_1 is fixed. So, the model presented in Equation (4.39) is used.

However, this joint will be used with rotating θ_1, so a model in which both angles are not fixed is presented below. Again, using Figure 4.15, the non-rigid joint model can be obtained, presented in Equations (4.40) and (4.41):

$$J_1\dot{\omega}_1 = ki - B_1\omega_1 - k_s(\theta_1 - \theta_2) - B_d(\omega_1 - \omega_2) \tag{4.40}$$

$$J_2\dot{\omega}_2 = -B_2\omega_2 + k_s(\theta_1 - \theta_2) + B_d(\omega_1 - \omega_2) - dmg\sin\theta_2 \tag{4.41}$$

where J_1 and J_2 are the moment of inertia (kg·m^2), ω_1 and ω_2 the respective angular speeds (rad/s), B_1 and B_2 the viscous friction coefficients (N·m·s), θ_1 and θ_2 the angular positions (rad), k_s the springs constant (N/m), B_d the damping coefficient ($N \cdot s/m$), d the distance (m) between the joint and a mass m (kg) and finally g the gravity constant (m/s^2).

Now, using the rearranged Equation (4.14), it is possible to obtain the following equations, rearranged as well.

$$J_1\dot{\omega}_1 = \frac{k}{R}u - \frac{k^2}{R}\omega_1 - B_1\omega_1 - k_s\theta_1 + k_s\theta_2 - B_d\omega_1 + B_d\omega_2 \tag{4.42}$$

$$J_2\dot{\omega}_2 = -B_2\omega_2 + k_s\theta_1 - k_s\theta_2 + B_d\omega_1 - B_d\omega_2 - dmg\sin\theta_2 \tag{4.43}$$

Considering that $\dot{\theta}_1 = \omega_1$ and $\dot{\theta}_2 = \omega_2$, Equations (4.42) and (4.43) become:

$$\dot{\omega}_1 = \frac{1}{J_1}\left(\frac{k}{R}u - \left(\frac{k^2}{R} + B_1 + B_d\right)\omega_1 + B_d\omega_2 - k_s\theta_1 + k_s\theta_2\right) \tag{4.44}$$

$$\dot{\omega}_2 = \frac{1}{J_2}\left(B_d\omega_1 - (B_2 + B_d)\omega_2 + k_s\theta_1 - k_s\theta_2 - dmg\sin\theta_2\right) \tag{4.45}$$

Considering that the angular variations in θ_2 would be small, Equation (4.47) is simplified, replacing the $\sin\theta_2$ with θ_2. The system then becomes:

$$\dot{\omega}_1 = \frac{1}{J_1}\left(\frac{k}{R}u - \left(\frac{k^2}{R} + B_1 + B_d\right)\omega_1 + B_d\omega_2 - k_s\theta_1 + k_s\theta_2\right) \tag{4.46}$$

$$\dot{\omega}_2 = \frac{1}{J_2}\left(B_d\omega_1 - (B_2 + B_d)\omega_2 + k_s\theta_1 - k_s\theta_2 - dmg\theta_2\right) \tag{4.47}$$

Using Equations (4.46) and (4.47) and transforming them into a state-space model in the form of Equation (4.48), it becomes:

$$\dot{\mathbf{X}} = \mathbf{A}\mathbf{X} + \mathbf{B}u \tag{4.48}$$

$$\mathbf{X} = \begin{bmatrix} \theta_1 \\ \omega_1 \\ \theta_2 \\ \omega_2 \end{bmatrix} \tag{4.49}$$

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\omega}_1 \\ \dot{\theta}_2 \\ \dot{\omega}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_s}{J_1} & -\left(\frac{k^2}{RJ_1} + \frac{B_1+B_d}{J_1}\right) & \frac{k_s}{J_1} & \frac{B_d}{J_1} \\ 0 & 0 & 0 & 1 \\ \frac{k_s}{J_2} & \frac{B_d}{J_2} & -\frac{k_s+dmg}{J_2} & \frac{-B_2+B_d}{J_2} \end{bmatrix} \begin{bmatrix} \theta_1 \\ \omega_1 \\ \theta_2 \\ \omega_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{k}{RJ_1} \\ 0 \\ 0 \end{bmatrix} u \tag{4.50}$$

$$y = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \omega_1 \\ \theta_2 \\ \omega_2 \end{bmatrix} \tag{4.51}$$

The function used to find the system's transfer function is presented in Equation (4.52).

$$G(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} \tag{4.52}$$

4.3.2.2 Prototype Design

Regarding the non-rigid joints, there were several versions between the first and the last one presented here. Throughout the research, it was discovered that the torsional moment required from the motors, specially on the rigid joint was very high. This can be explained with the rotational movements that are sometimes sudden. A way to solve this problem would be to use motor with gearboxes built with more resistant materials, which can transmit and support larger torques. However, this solution increases the weight and the cost of the vehicle, which is not compatible with the cost-effective paradigm. The solution that avoids these increased mechanical efforts, while still being complying with the construction paradigm, is the use of the two lower joints from each leg in a non-rigid configuration.

Using this configuration on the two lower joints increases the number of situations in which its properties can be useful. It is clear that this configuration is capable of handling horizontal and vertical impacts, as well as ground irregularities. The only configuration where these joints do not exhibit its vertical absorption capability is when it is completely stretched.

For the non-rigid joint construction, a passive system was designed and developed. Applying eight springs, attached to a central circular support part, as further represented in Figure 4.17. A rotational force is applied in the support's rotational center, transmitting this to the next link through the springs, providing elasticity and cushioning to the rotational movement.

However, there are few DC motors available with higher reduction gears, that are capable of producing the torques required by this joint. This torque is used for the positioning over the z_r-axis. To increase the applied torque, an involute spur gear was added immediately after the motor shaft. This reduction gear increases the rotational torque to the desired values, while helping reducing the joint backlash. This additional gear set has a 3.2:1 ratio, maintains the system modularity, and solves the reported problem.

A solution was reached that is mechanically stable, extremely robust and modular. It is easy to assemble and disassemble, its stiffness can easily be changed by simply replacing the springs, as well as the desired damping coefficient. Concerning the latter, an off-the-shelf rotary damper (ACE FRT-K2-103) was coupled to the shaft of the joint's central part. Fig. 4.16 presents the obtained configuration.

Having finally managed to develop a solution that meets the necessary requirements and solves all problems found in the previous versions, Figure 4.17 presents the CAD explosion of the mentioned non-rigid joint.

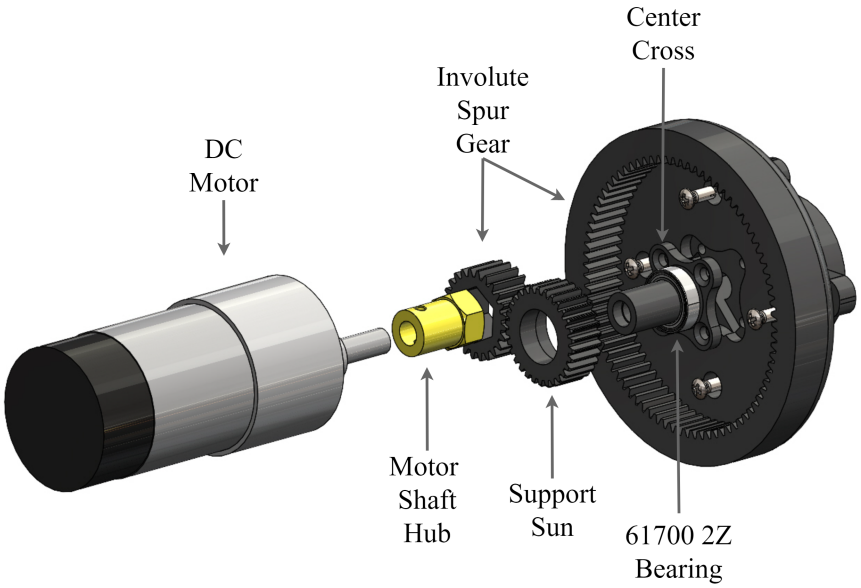


Figure 4.16: Motor and Involute Spur Gear Assembly Explosion.

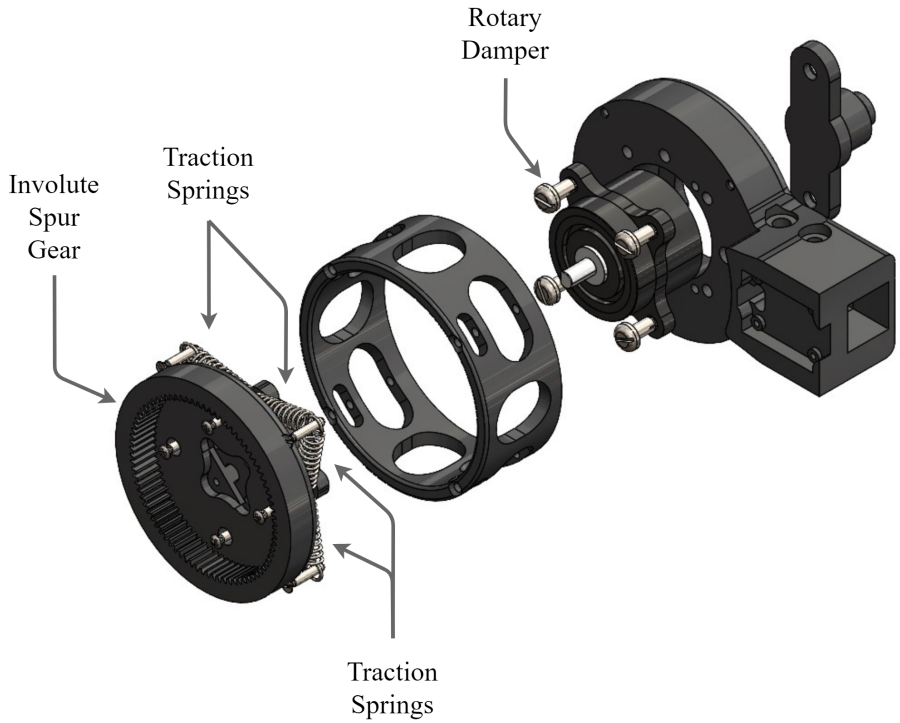


Figure 4.17: Non-Rigid Joint Explosion Assembly.

Regarding the support parts and alignment of the joints, it was decided to use the internal shaft motor bearing to support one side of the joint. To support and reduce friction on the other side of the joint, another bearing is used. Considering that, a rotating shaft with a larger diameter adds strength and slightly reduces the possibility of clearances, it was decided to use a bearing with a 20 mm internal diameter (6004 2Z). With the introduction of the inline spur gear, it was necessary to add an extra bearing to reduce the friction between the center cross alignment part and the sun (61700 2Z). Another was used to achieve the alignment between the center cross part and the motor support part (61900 2Z). Figure 4.18 presents the mechanical explosion for the assembly where these bearings are placed, and the remaining parts that were custom designed for the system.

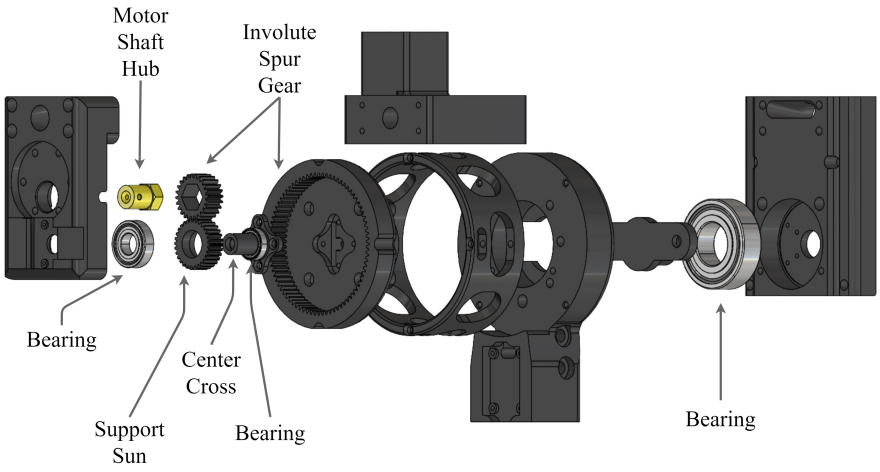


Figure 4.18: Mechanical Explosion Assembly Non-Rigid Joint.

Figure 4.19 presents the non-rigid joint mechanical explosion with all the electronic components attached to it, in their designated positions.

To receive data from the sensors that each joint has, as well as to control the drives associated with each motor, the Arduino Nano 33 IOT board was used. These microcontrollers present a powerful solution, at a relatively low cost.

To facilitate the connections to the respective peripherals, an I/O Shield was used for each microcontroller board, in which the connections are mainly performed through Jumper Wires.

To collect the data from the motor movement, an absolute encoder was added, connected directly to the motor shaft (defined as Absolute Encoder In, from here on). Using this information, it is possible to calculate the springs extension, and avoid an extension greater than the allowed one, which would cause a damage to the springs. To obtain this measurement, AS5048B encoders were used, which communicate through Inter-Integrated Circuit (I2C) protocol. In addition to this, and taking into account that these are non-rigid joints, it was considered important to use an absolute encoder attached to the joints' output. For this purpose, another high resolution (14-bit) rotational position sensor —AMS AS5048A— was used, which communicates through the Serial Peripheral Interface (SPI) protocol to transmit the readings. Figure 4.20 helps understand the encoders' locations and the communication between the sensors and the microcontroller boards.

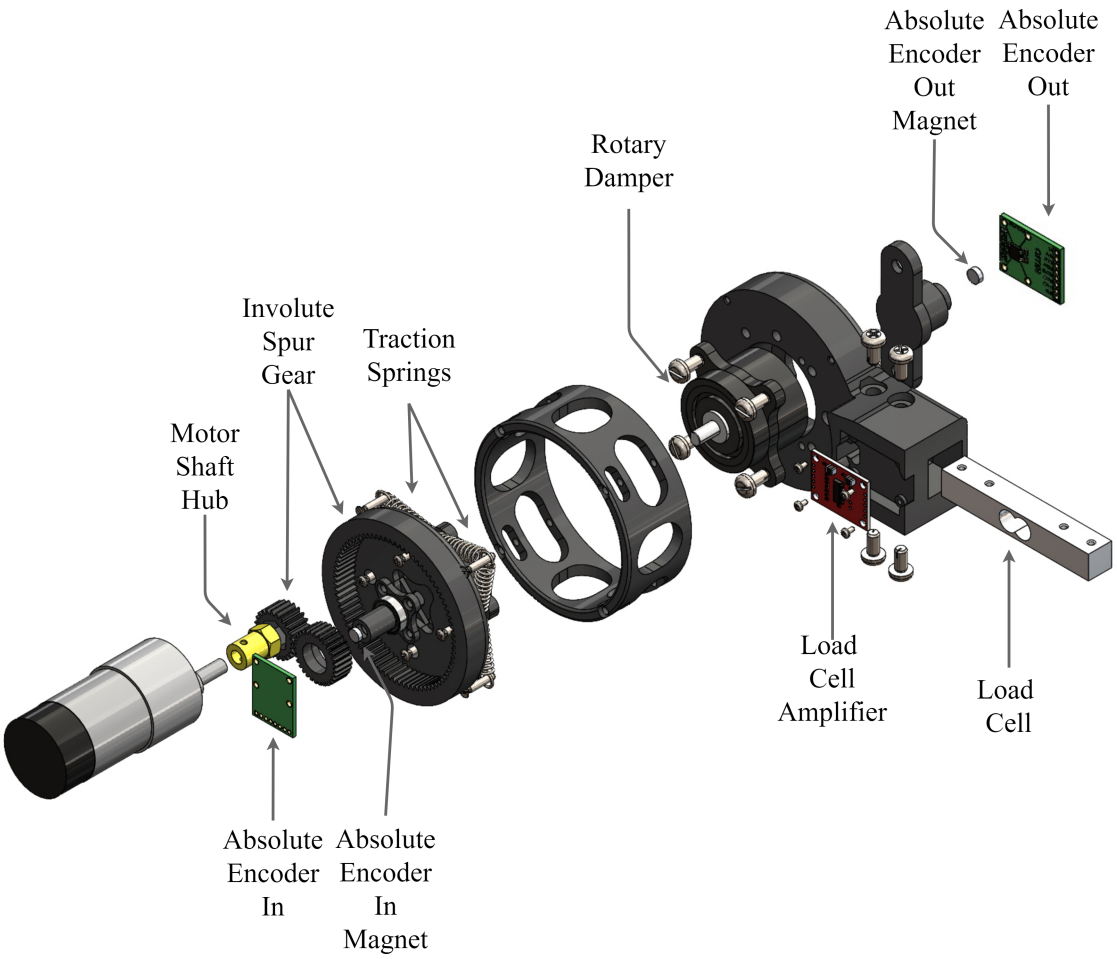


Figure 4.19: Mechanical Explosion Assembly Non-Rigid Joint with Electronic Components.

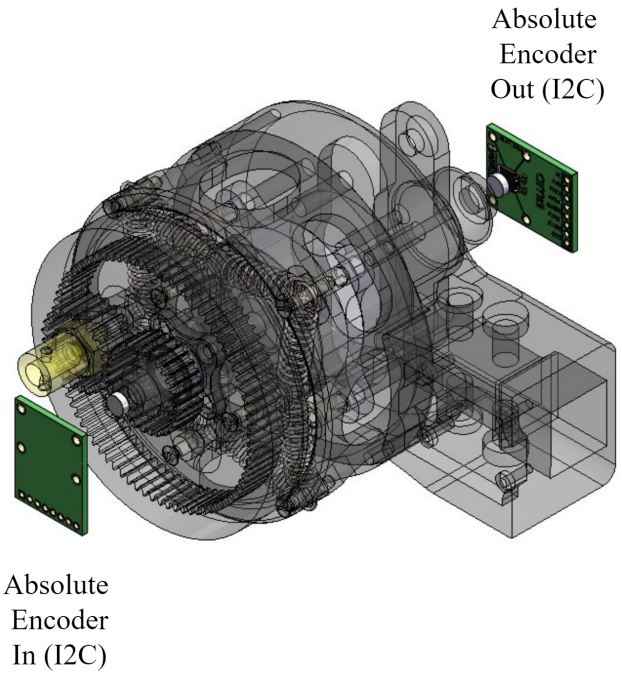


Figure 4.20: Non-Rigid Joint Absolute Encoders Connections.

4.3.2.3 Prototype Model Parameter Estimation

Taking Equations (4.49) and (4.50) and considering a centered finite difference discretization:

$$\dot{\omega}_1 \approx \frac{\omega_{1k+1} - \omega_{1k-1}}{2\Delta t} \tag{4.53}$$

$$\dot{\omega}_2 \approx \frac{\omega_{2k+1} - \omega_{2k-1}}{2\Delta t} \tag{4.54}$$

$$\dot{\omega}_1 = \underbrace{-\frac{k_s}{J_1}}_{\beta_1}\theta_1 - \underbrace{\left(\frac{k^2}{RJ_1} + \frac{B_1 + B_d}{J_1}\right)}_{\beta_2}\omega_1 + \underbrace{\frac{k_s}{J_1}}_{\beta_3}\theta_2 + \underbrace{\frac{B_d}{J_1}}_{\beta_4}\omega_2 + \underbrace{\frac{k}{J_1 R}}_{\beta_5}u \tag{4.55}$$

$$\dot{\omega}_2 = \underbrace{\frac{k_s}{J_2}}_{\beta_6}\theta_1 + \underbrace{\frac{B_d}{J_2}}_{\beta_7}\omega_1 - \underbrace{\frac{k_s + dMg}{J_2}}_{\beta_8}\theta_2 - \underbrace{\frac{-B_2 + B_d}{J_2}}_{\beta_9}\omega_2 \tag{4.56}$$

The matrices $\mathbf{Y}$, $\mathbf{X}$ and $\boldsymbol{\beta}$ can be obtained through the definitions in Equation (4.57). These use $\dot{\omega}_{1,i}$, $\dot{\omega}_{2,i}$, $\theta_{1,i}$, $\theta_{2,i}$, $\omega_{1,i}$, $\omega_{2,i}$, and u_i representing the corresponding variable's i-th sample. Again, to perform the joint's model parameters estimation, N samples are used (N = 2000).

$$\mathbf{Y} = \begin{bmatrix} \dot{\omega}_{1,1} \\ \vdots \\ \dot{\omega}_{1,N} \\ \dot{\omega}_{2,1} \\ \vdots \\ \dot{\omega}_{2,N} \end{bmatrix}, \mathbf{X} = \begin{bmatrix} \theta_{1,1} & \omega_{1,1} & \theta_{2,1} & \omega_{2,1} & u_1 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \theta_{1,N} & \omega_{1,N} & \theta_{2,N} & \omega_{2,N} & u_N & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \theta_{1,1} & \omega_{1,1} & \theta_{2,1} & \omega_{2,1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \theta_{1,N} & \omega_{1,N} & \theta_{2,N} & \omega_{2,N} \end{bmatrix} \text{ and } \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \\ \beta_6 \\ \beta_7 \\ \beta_8 \\ \beta_9 \end{bmatrix} \tag{4.57}$$

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} \tag{4.58}$$

$$\begin{bmatrix} \dot{\omega}_{1,1} \\ \vdots \\ \dot{\omega}_{1,N} \\ \dot{\omega}_{2,1} \\ \vdots \\ \dot{\omega}_{2,N} \end{bmatrix} = \begin{bmatrix} \theta_{1,1} & \omega_{1,1} & \theta_{2,1} & \omega_{2,1} & u_1 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \theta_{1,N} & \omega_{1,N} & \theta_{2,N} & \omega_{2,N} & u_N & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \theta_{1,1} & \omega_{1,1} & \theta_{2,1} & \omega_{2,1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \theta_{1,N} & \omega_{1,N} & \theta_{2,N} & \omega_{2,N} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \\ \beta_5 \\ \beta_6 \\ \beta_7 \\ \beta_8 \\ \beta_9 \end{bmatrix} \tag{4.59}$$

Once again, to estimate parameters of the $\boldsymbol{\beta}$ matrix, the model presented in Equation (3.7) can be used:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \tag{4.60}$$

Regarding the physical tests, in addition to the the test mentioned in Section 4.3.2.1, another can be performed to help model the non-rigid joint:

Pulling the link attached to the non-rigid joint to a certain angle - In this test, the link attached to the non-rigid joint is pulled manually for a certain set of increasing angles. For those angles, the

torque is measured;

The collected data are graphically captured and introduced in Figure 4.21 in blue in the form of an angle versus time, while the optimization result of the model is shown in orange. After the optimization process had been completed, the following data were entered for J and B_d values of 0.007 kg·m^2 and 1.218 N·s/m were obtained. Figure 4.22 graphically shows the absolute error of the model compared to the real position and the numerical error values are summarized in Table 4.2.

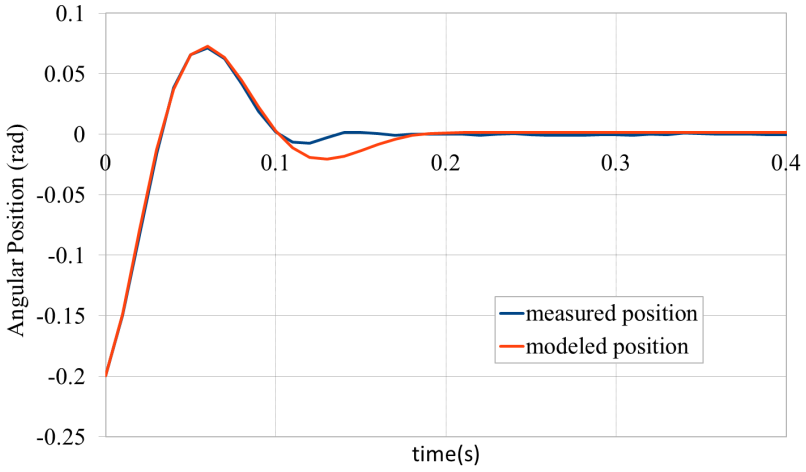


Figure 4.21: Angular Position (rad) over Time (s).

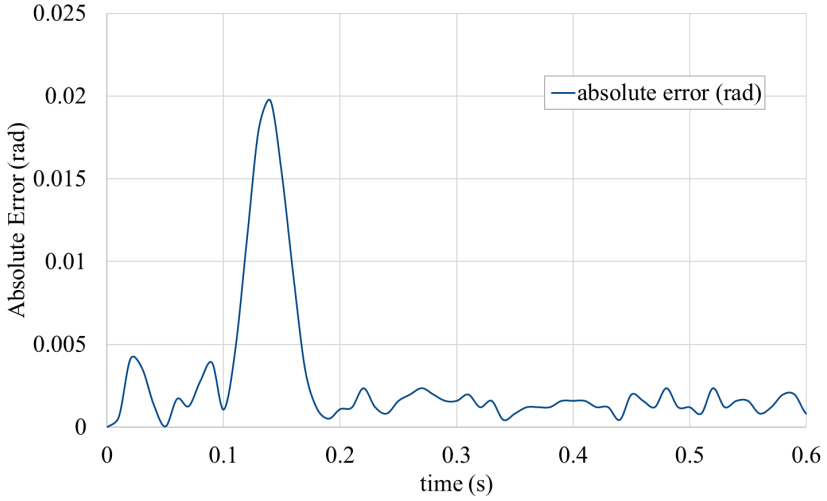


Figure 4.22: Absolute Error (rad) over Time (s).

Table 4.2: Absolute Error Analysis.

Absolute Error	
Maximum	0.0197 rad
Average	0.0026 rad

The estimated parameters are shown in Table 4.3.

Table 4.3: Estimated Joint Parameters.

Parameters	
k_s	16.92 N/m
B_d	1.218 N·s/m
J	0.007 kg·m^2

For the remaining non-rigid joint model parameters, a wave with an amplitude of 7.2 V was applied, with a starting frequency of 0.1 Hz, final frequency of 2.5 Hz, and a total time of 20 s. The following Equation (4.61) presents the values obtained for the beta parameters introduced in Equations (4.55) and (4.56). It is noticeable that these parameters seem well estimated since, for example, the values of β_1 and β_3 are practically symmetrical.

$$\boldsymbol{\beta}^T = \begin{bmatrix} -86.09 & -19.62 & 85.92 & 0.2081 & 9.086 & 56.31 & 2.24 & -79.46 & -2.182 \end{bmatrix} \tag{4.61}$$

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\omega}_1 \\ \dot{\theta}_2 \\ \dot{\omega}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -86.09 & -19.62 & 85.92 & 0.2081 \\ 0 & 0 & 0 & 1 \\ 56.31 & 2.24 & -79.46 & -2.182 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \omega_1 \\ \theta_2 \\ \omega_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 9.086 \\ 0 \\ 0 \end{bmatrix} u \tag{4.62}$$

This model originated the following result, which can be compared to the data obtained in the test through the graphic on Figure 4.23. Given this model, the obtained transfer function is presented in Equation (4.63).

$$G(s) = \frac{20.35s + 511.6}{s^4 + 21.8s^3 + 207.9s^2 + 1543s + 2003} \tag{4.63}$$

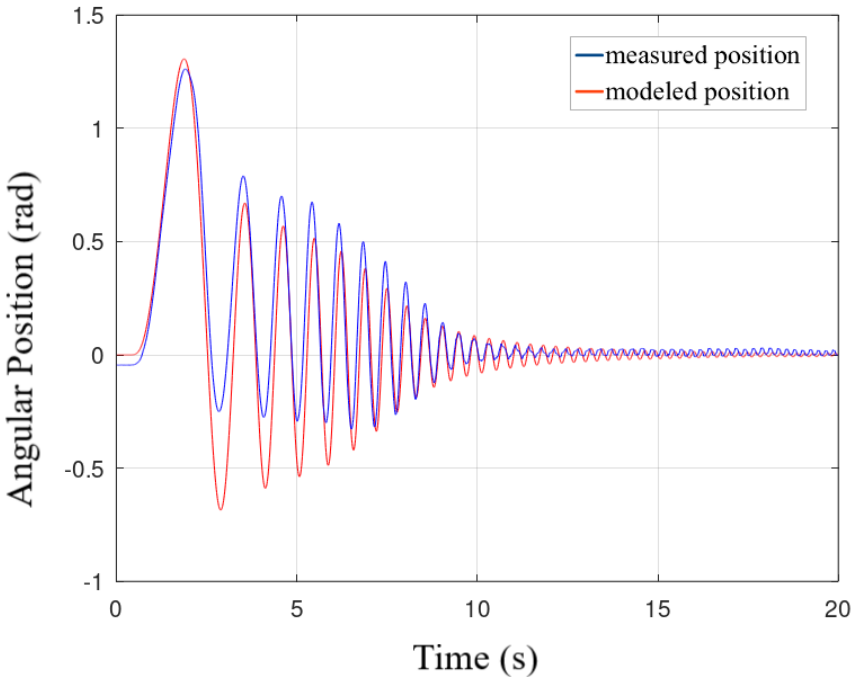


Figure 4.23: Non-Rigid Joint Model vs. Real Angular Positions.

4.3.2.4 Prototype Model including Non-Linear Elements

Implementing non-linear elements on a controller design for a physical prototype is not trivial. However, concerning realistic simulation, all these non-linearities contribute to enhance and strengthen the used model. Thus, some non-linear elements will be added to the previously presented non-rigid joint model.

For this operation, it is necessary to use the equations in the time domain to optimize the collected data and obtain the desired parameters. For this, Equation (4.39) and the inverse Laplace transform are used, obtaining the following equations:

$$\theta_2(t) = \theta_0 \cdot \sqrt{\frac{(\frac{\alpha}{\omega_n} - \xi \cdot \omega_n)^2}{1-\xi^2} + 1} \cdot e^{-\xi \omega_n t} sin(\omega_n \cdot \sqrt{1-\xi^2} \cdot t + \phi) \tag{4.64}$$

$$\phi = tan^{-1}\left(\frac{\omega_n \sqrt{1-\xi^2}}{\alpha - \xi \omega_n}\right) \tag{4.65}$$

The model presented in Equations (4.46) and (4.47) is correct and will be applied to the physical non-rigid joint. However, after performing some tests, and taking into account the non-linearity of the

joint's dead zone, it was decided to include this non-linearity in the realistic simulation model, by first changing Equation (4.32):

$$\frac{d\omega(t)}{dt} = \frac{1}{J}(-d(\theta(t)) - B(\omega(t))) \tag{4.66}$$

where $d(\theta)$ is introduced to model the dead-zone and $B(\omega)$ to model the viscous friction. $d(\theta)$ is defined by Equation (4.67), where $-\theta_d$ and θ_d are the dead zone left and right limits, as illustrated in Figure 4.24, while $B(\omega)$ is defined by Equation (4.68).

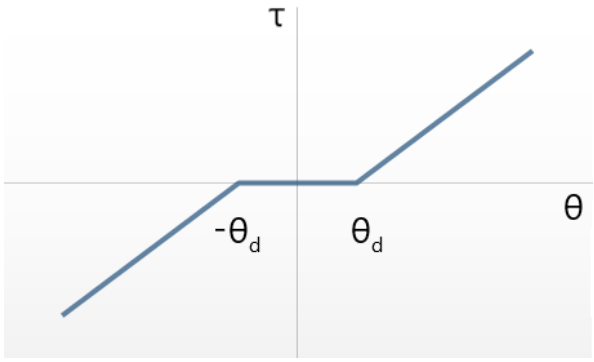


Figure 4.24: Dead-Zone Specification.

$$d(\theta) = \begin{cases} (\theta - \theta_d) \cdot K & \text{if } \theta > \theta_d \\ 0 & \text{if } -\theta_d < \theta < \theta_d \\ (\theta + \theta_d) \cdot K & \text{if } \theta < -\theta_d \end{cases} \tag{4.67}$$

$$B(\omega) = \begin{cases} B \cdot \omega + G \cdot (\omega - G_d) & \text{if } \omega > G_d \\ B \cdot \omega & \text{if } -G_d < \omega < G_d \\ B \cdot \omega + G \cdot (\omega + G_d) & \text{if } \omega < -G_d \end{cases} \tag{4.68}$$

For the discretization, it was considered that $\theta[-2] = \theta[-1] = \theta[0]$. Thus,

$$\omega(t) = \frac{d\theta(t)}{dt} \simeq \frac{\Delta\theta}{\Delta t} \tag{4.69}$$

$$\omega[k] = \frac{\Delta\theta}{\Delta t} = \frac{\theta[k] - \theta[k-1]}{\Delta t} \tag{4.70}$$

A finite difference approximation method was used to represent the referred model, more specifically a second-order central. This is presented in Equation (4.71) and manipulated to obtain the model presented in Equation (4.73).

$$\frac{d\omega(t)}{dt} = \frac{d^2\theta(t)}{dt^2} \simeq \frac{\theta[k+1] - 2\theta[k] + \theta[k-1]}{\Delta t^2} \tag{4.71}$$

$$\frac{\theta[k+1]-2\theta[k]+\theta[k-1]}{\Delta t^2} = \frac{1}{J}\left(-d(\theta[k])-B(\omega[k])\right) \tag{4.72}$$

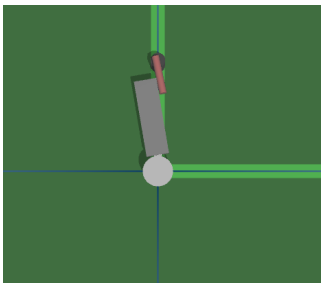
$$\theta[k+1] = \frac{\Delta t^2}{J}\left(-d(\theta[k])-B(\omega[k])\right)+2\theta[k]-\theta[k-1] \tag{4.73}$$

4.3.2.5 Realistic Simulation including Non-Linear Elements

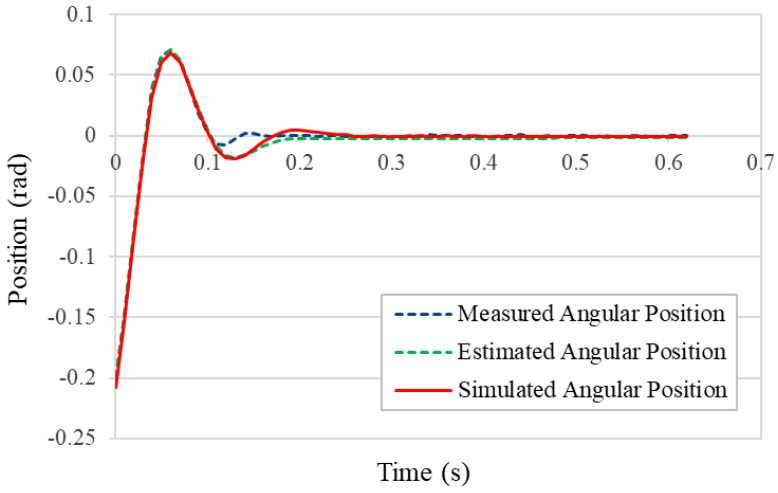
SimTwo [86] provides a 3D environment with incorporated physics engine. It can simulate realistically different types of robots, namely manipulators, wheeled and legged mobile robots. The robotic system is decomposed into a combination of rigid bodies and joints, which are optionally powered with motors. Considering all the features of the SimTwo simulator, it seemed feasible to create a realistic simulation model of the vehicle through this software.

As mentioned before, the non-linear model is useful to produce a realistic simulation, using the chosen simulator. It was created a system that simulates the spring-damper-mass system, to obtain simulation data to compare with the real data. With these data, it is possible to validate the model, as well as obtain the error between them.

For the presented non-rigid joint, the used parameters were the ones presented in Table 4.3, as well as the parameters obtained in Equation (4.61). Figure 4.25a present the simulated setup, while Figure 4.25b presents the obtained results.



(a)



(b)

Figure 4.25: (a) Simulator Setup for Observation of the Oscillatory Response of the Non-Rigid Joint. (b) Angular Position (Real vs. Estimated vs. Simulated.

The settling time of the system is approximately 200 ms (considering 2% range) and the 98% rise time around 30 ms. The data collected in the simulation (shown in green) was compared against the results of the test performed on the real non-rigid joint and its modeled response. The mean absolute errors of the position in simulation compared to the estimated and real positions are, respectively, 0.0025 and 0.0023 rad, while the maximum absolute errors are 0.0084 and 0.0181 rad. The reason the obtained response in simulation is closer to the modeled one instead of the real one is related to the uncertainties that derive from data acquisition in real settings, which might not be modeled in a simulated environment (e.g., load cell noise). The joint simulation results were submitted and accepted for publication in the CLAWAR international conference [123].

4.4 Control for Robotic Joint Systems

The issue of the sudden changes in speed can shorten the life of the motor. Furthermore, this also introduces several undesirable torsional moment components. A non-rigid joint should increase the ability of the robotic system to operate in highly dynamic environments. In addition it should also increase its tolerance to impacts, i.e., small force impacts with objects or people.

Although several types of strategies can be used to control the joint, one mode of operation that was intended for this platform was compliant control. The main controllers are the position controller and the speed controller, as both have PI controllers, with a feedforward block added to the speed. Then the force controller is used in cascade with the two previous ones, the result of them being the force controller setpoint, using only a P controller with feedforward, since the introduction of the smallest integral component seemed to make the system unstable. Finally, a saturation block is applied to define the limits of the admissible force. This method corresponds to the premise of compliant control.

4.4.1 Angular Positioning Control

For lower-level control, position and force controllers were tested independently of each other. Figure 4.26 presents the first controller block diagram, for pure position control. Figure 4.27 presents the response to a step in the reference of the position.

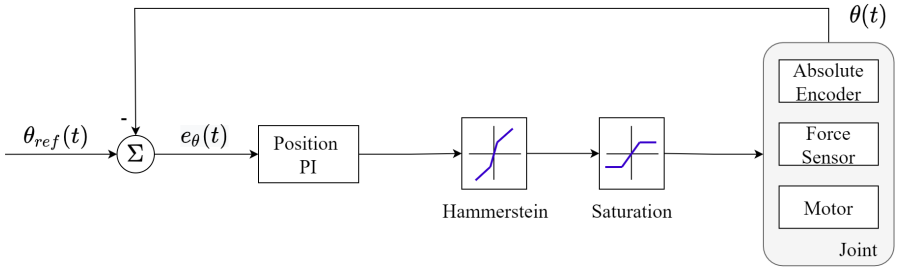


Figure 4.26: Position Controller Block Diagram.

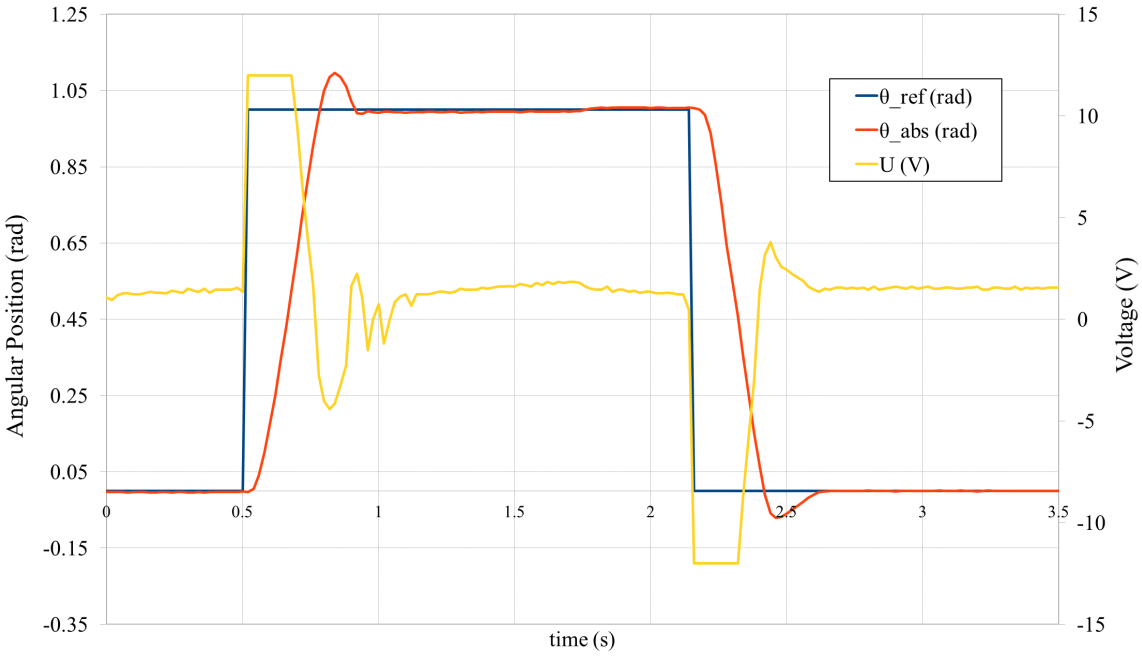


Figure 4.27: Response to a Step in the Position Reference.

4.4.2 Backdrivability—Force Controller, with Zero Reference

For the pure force controller, Figure 4.28 shows the controller block diagram. Figure 4.29 presents the obtained data for the experiment.

An important aspect of this controller is that it allows implementation of a backdrivability function to the joint, as shown in Figure 4.29. The torque limit was defined as 0 N·m, therefore the control followed the movement of the operator after registering an absolute maximum error of 0.5 N·m in rapid movements, and 0.27 N·m in slower ones. As can be observed, the movement was performed between about 1.7 and -1.7 radian with several variations over about 24 s. Although this function is currently not useful for the work under development, it can be very interesting for use in a robot manipulator.

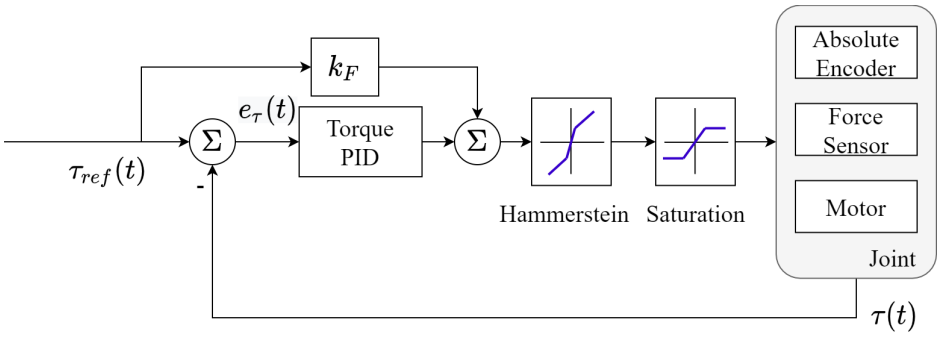


Figure 4.28: Force Controller Block Diagram.

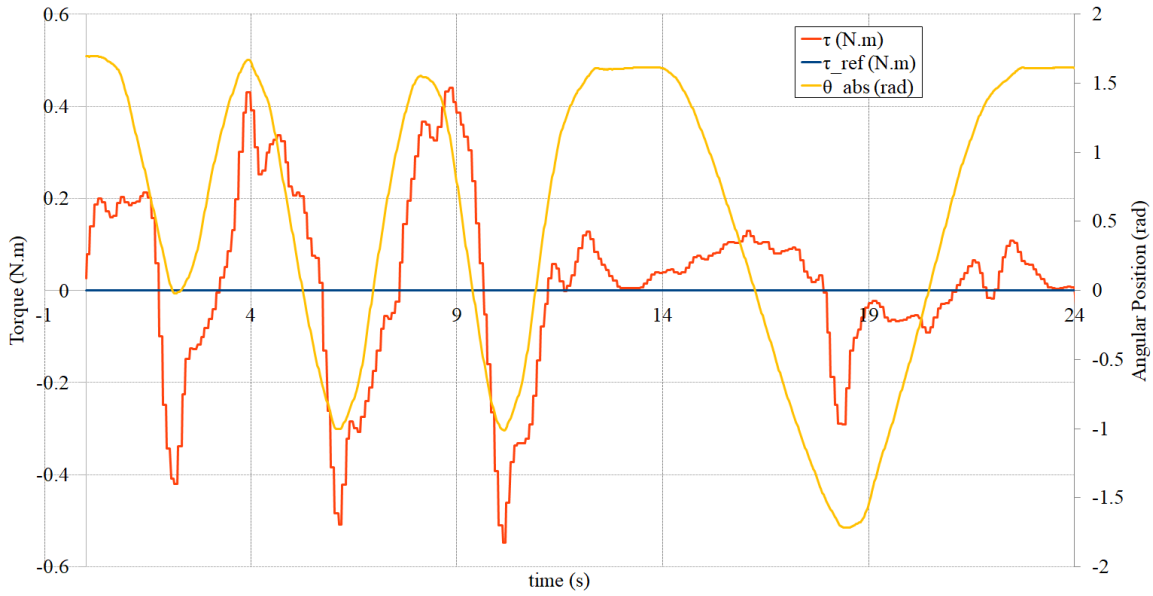


Figure 4.29: Response to Changes in the Position, Maintaining Force Reference.

4.4.3 Compliant Position Controller

After the low-level controllers were individually tuned, the cascade controller was developed with a combination of position, speed and force control. The controller block diagram is presented in Figure 4.30.

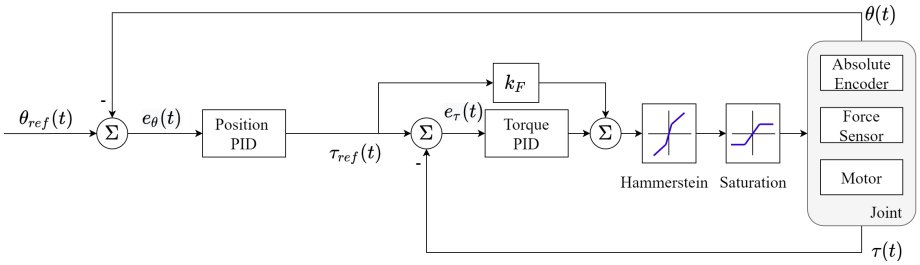


Figure 4.30: Compliant Position and Force Controller Block Diagram.

4.4.3.1 Without Obstacle in the Trajectory

The next test was to establish a reference in position and a limit for the force controller. The diagram in Figure 4.31 represents a movement up to an angle of 1 radian and back to 0 with no obstacles in the way.

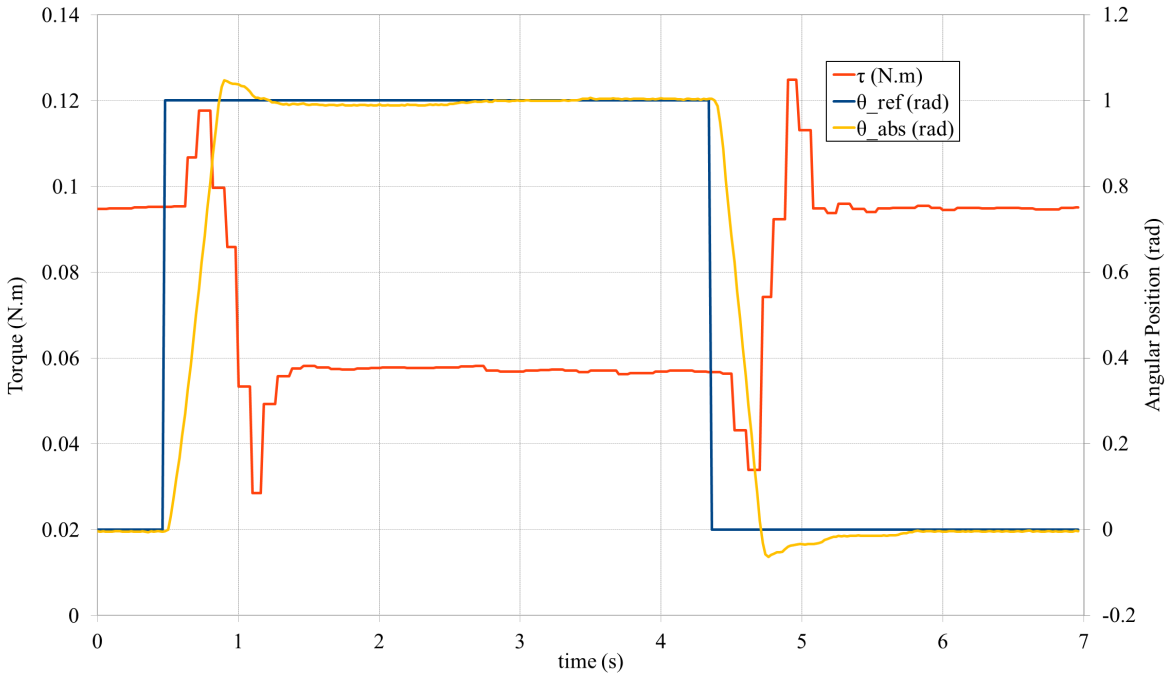


Figure 4.31: Change in Position Reference with Compliant Controller.

4.4.3.2 With Obstacle in the Trajectory

Figure 4.32 shows the data relating to a positioning attempt with an obstacle, where the allowable absolute torque limit was defined as 0.7 N·m. As can be seen, the reference is not reached, but the force limit was reached and the position remains in the closer possible position.

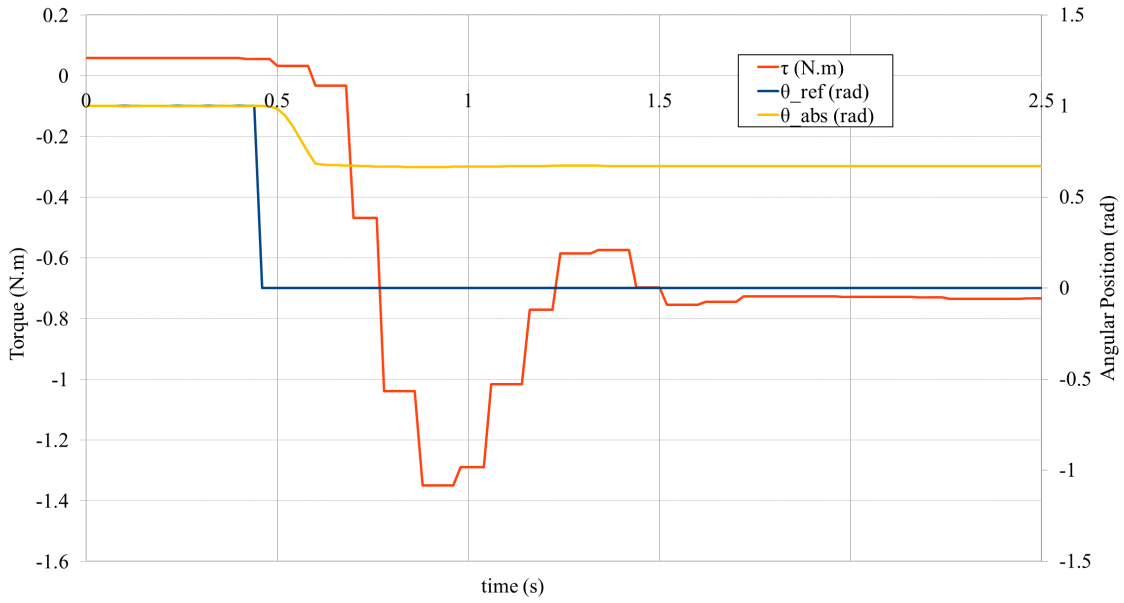


Figure 4.32: Change in Position Reference with Compliant Controller with Obstacle in the Way.

4.4.4 Hermite Spline for Smooth Profile Generation

It was decided to use smoothed profiles for the speed and position of the motor controller. The current approach allows the trajectory to be recalculated if the user decides to change the point or the final slope of the trajectory. For this purpose, the Hermite Spline was used to obtain the trajectory functions. This function is very useful when the end point and/or the slope change during the trajectory performance. When this happens, it is only necessary to input new end references to the trajectory generator, obtaining the initial values from data collected from the current position and velocity.

4.4.5 Trajectory Following

When the trajectory generator mentioned in the previous section was introduced into the control system (as can be seen in Figure 4.33), the compliant control system with force limitation and trajectory tracking was implemented and tested.

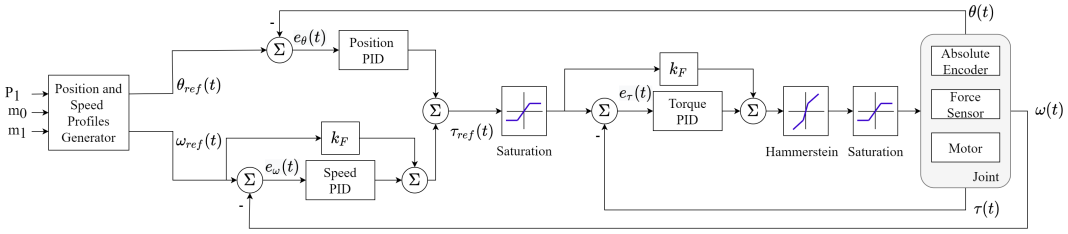


Figure 4.33: Full compliant controller block diagram.

The diagram in Figure 4.34 shows a movement to an angle of 1 radian (up), without obstacles, and the corresponding speed behavior (down).

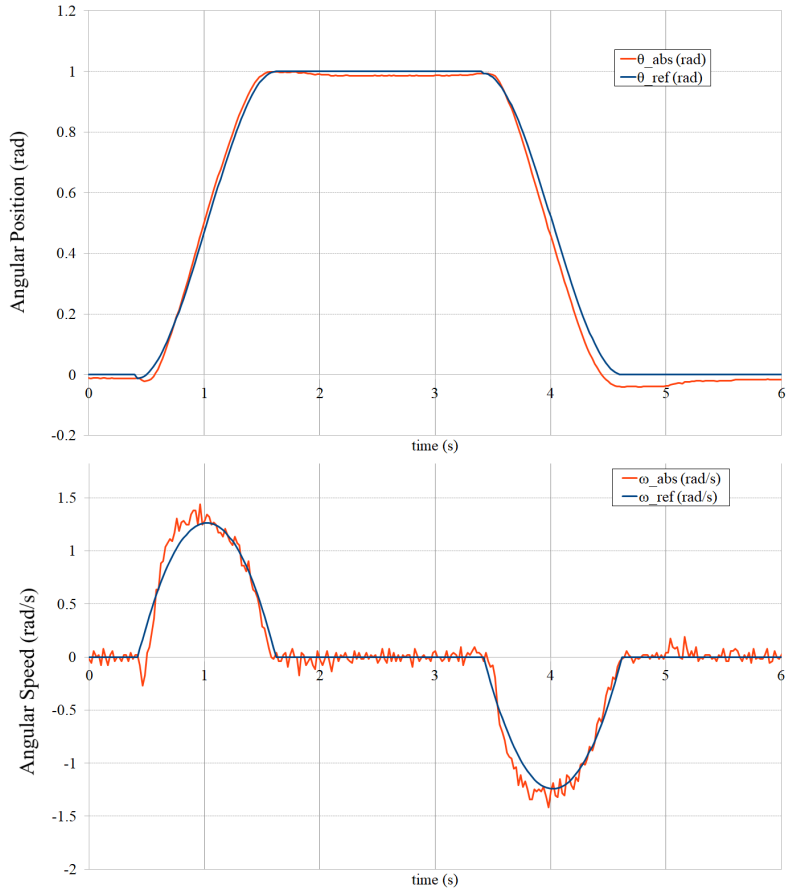


Figure 4.34: Compliant controller trajectory following in angular position (up) and speed (down).

4.4.6 Weight Placement

Finally, a test for the placement of unknown disturbances was carried out by introducing loads

into the spoon-shaped part presented in Figure 4.35.

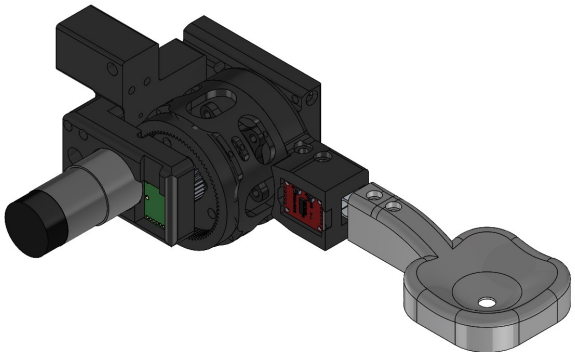


Figure 4.35: Spoon-Shaped Designed Part.

4.4.6.1 Without Reaching the Strength Limit

First, a load was placed that did not exceed the defined force limit, so that it was able to hold the load in the defined position (0 rad). The data associated with this test are presented in Figure 4.36.

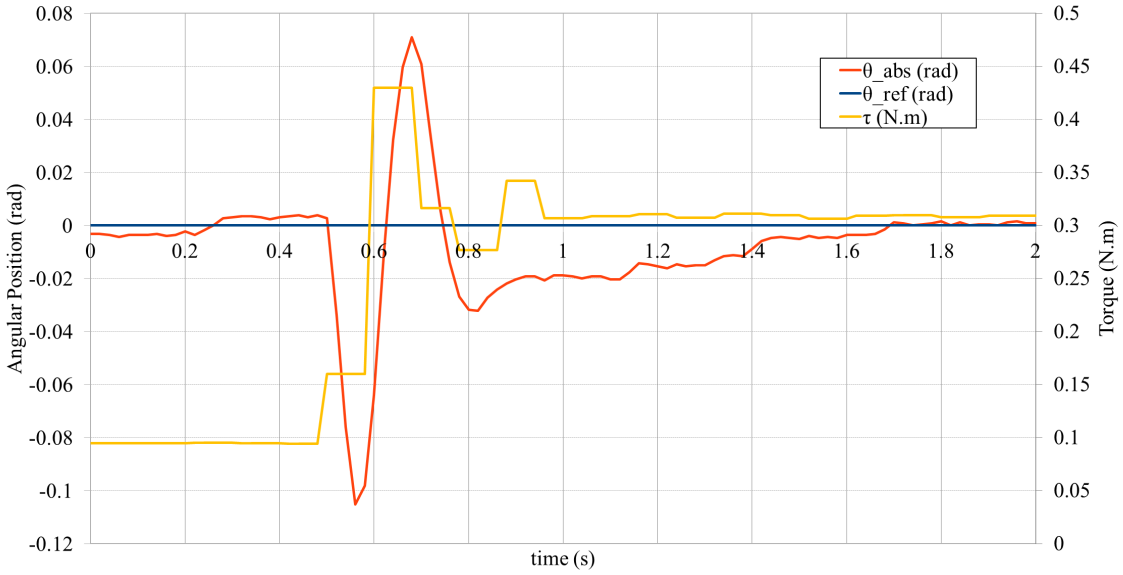


Figure 4.36: Compliant Controller Disturbance Rejection with Position reference.

4.4.6.2 Exceeding the Strength Limit

In the second iteration of the test, two loads were placed, with placement of the second one exceeding the limit, causing the joint to give way and drop the loads. Figure 4.37 represents the behavior of the joint during this situation.

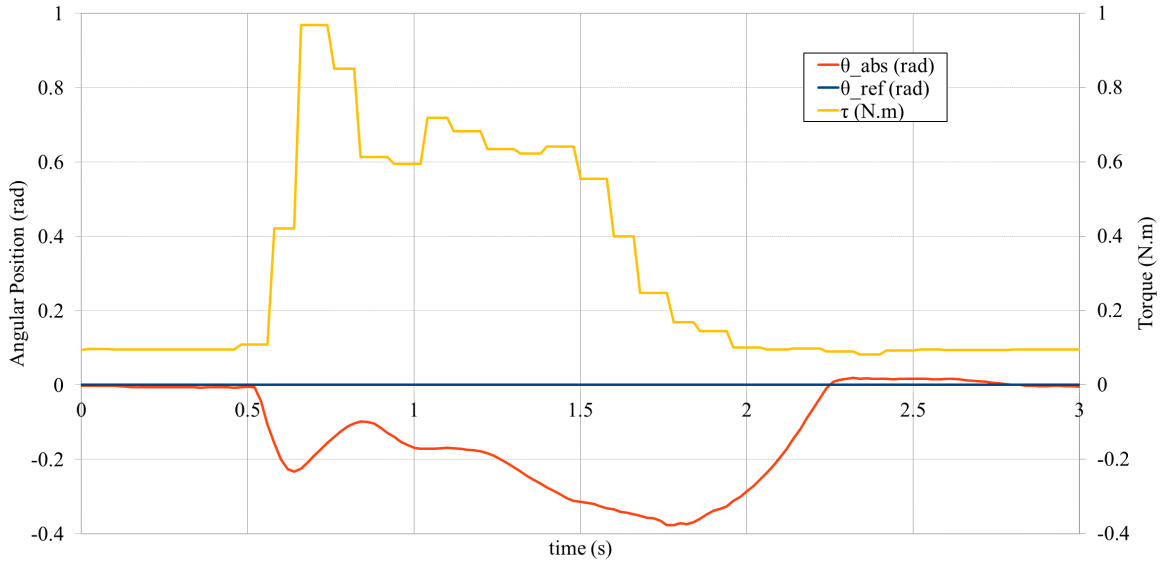


Figure 4.37: Compliant Controller Disturbance Rejection with Position reference and Force limit exceeded.

4.4.7 State-Space Controller

As mentioned through chapter 4.3.2, the non-rigid joint presents a fourth order transfer function. Thus, it requires more advanced control techniques. It was decided to use a state space control approach, since this is a complex, multiple-input system. This approach was also used to the rigid joint.

4.4.7.1 Rigid Joint

Although the rigid joint can use a more traditional control, using simpler controllers, it was decided to also use a state space approach. This is due to the fact that this joint, despite being rigid, may suffer more oscillations, due to its connection to the non-rigid joint. Furthermore, with this controller the joint presents a more suited response, being able to deal with step disturbances. Unlike the non-rigid joint, the considered reference signal is the relative angular position, in order to reduce the noise from the absolute encoder. The block diagram for the implemented controller is presented in Figure 4.38.

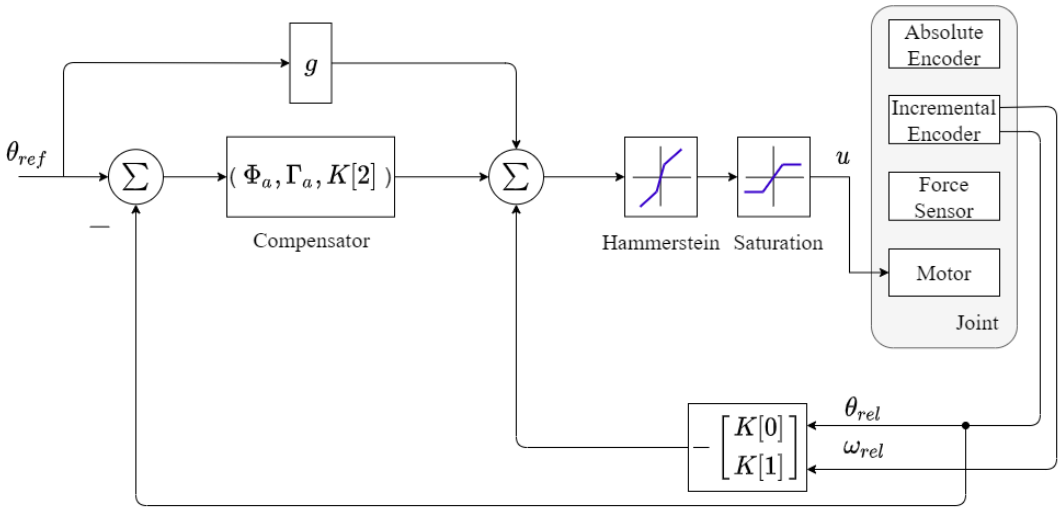


Figure 4.38: State-space Controller Block Diagram (Rigid Joint).

To obtain the following matrices $\mathbf{\Phi}$ and $\mathbf{\Gamma}$, Equations (3.43) and (3.44).

$$\mathbf{\Phi} = \begin{bmatrix} 0.9996 & 0.0092 & 0 \\ -0.0691 & 0.8473 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \mathbf{\Gamma} = \begin{bmatrix} 0.000396 \\ 0.077020 \\ 0 \end{bmatrix} \tag{4.74}$$

The additional dynamics matrices are equal to 1 to implement the controller's ability to deal with unit step disturbances.

$$\mathbf{\Phi}_a = [1] \, , \, \mathbf{\Gamma}_a = [1] \tag{4.75}$$

The matrix containing the calculated controller gains is now presented.

$$\mathbf{K}_s = \begin{bmatrix} 70.6230 \\ 2.8978 \\ 3.5952 \end{bmatrix} \tag{4.76}$$

As can be validated by the results presented in Figures 4.39, 4.40 and 4.41, the controller provides satisfactory results, that will be evaluated individually. The overshoot is not significat, thus it can be neglected.

Considering a reference of 0.5 rad, and with an initial position of 0 rad, the obtained rise time was 0.47 s and the settling time 0.78 s. In the downward direction, the rise time was 0.48 s and the settling time was 0.79 s.

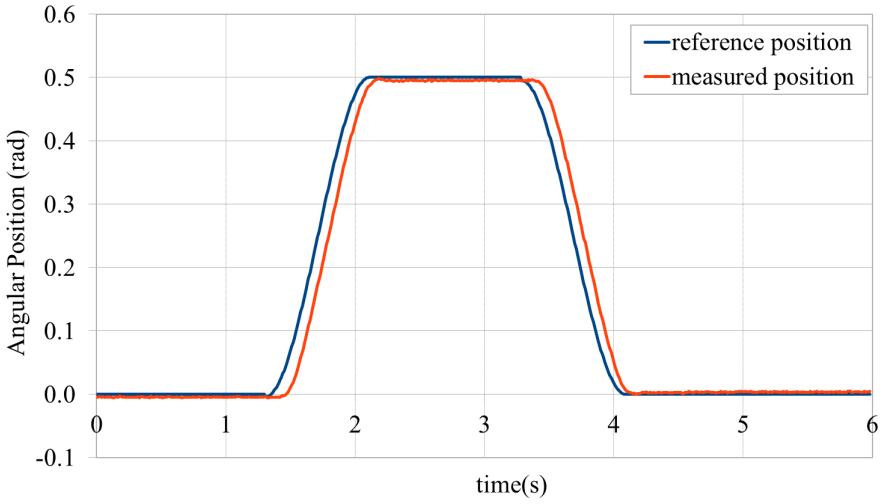


Figure 4.39: Rigid Joint Controller (θ_{ref} = 0.5 rad).

Considering a reference of 1 rad, and with an initial position of 0 rad, the obtained rise time was 0.47 s and the settling time 0.75 s. In the downward direction, the rise time was 0.47 s and the settling time was 0.76 s.

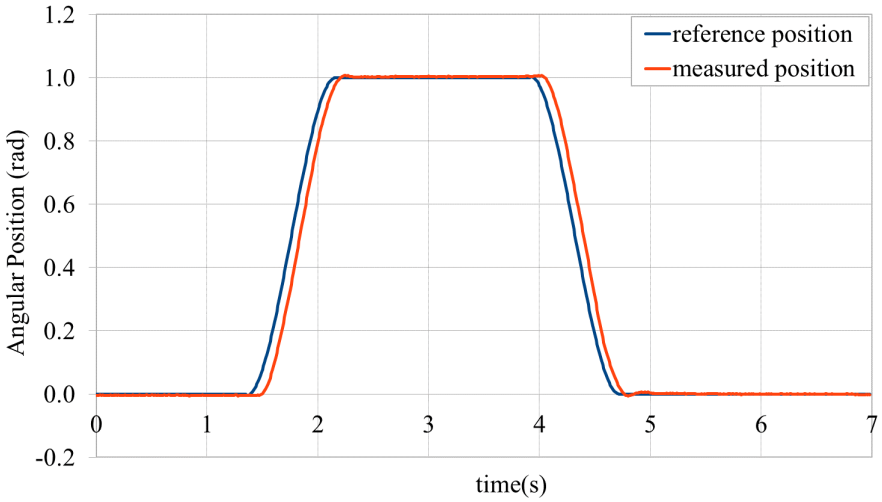


Figure 4.40: Rigid Joint Controller (θ_{ref} = 1 rad).

Finally, considering a reference of 1.5 rad, and with an initial position of 0 rad, the obtained rise time was 0.46 s and the settling time 0.77 s. In the downward direction, the rise time was 0.47 s and the settling time was 0.74 s.

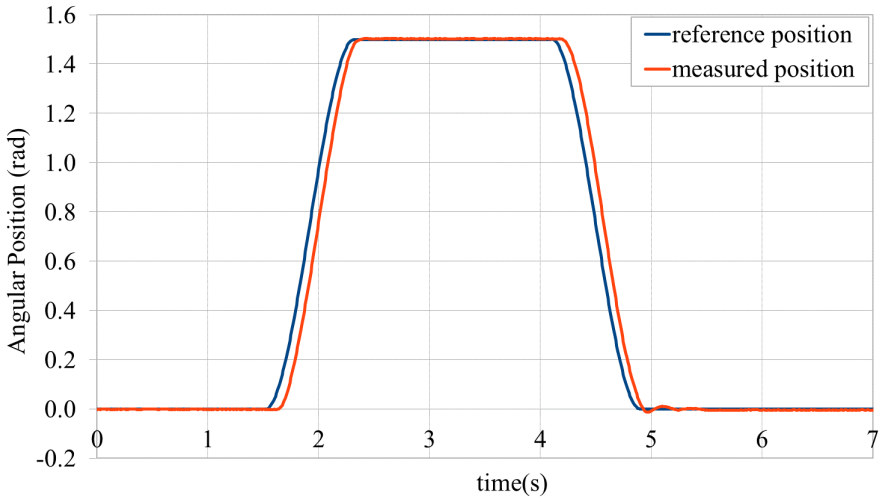


Figure 4.41: Rigid Joint Controller (θ_{ref} = 1.5 rad).

4.4.7.2 Non-Rigid Joint

The attempts to control the system with classic techniques were not succeeding, becoming easily unstable. The state space aproach is better suited since the non-rigid joint has two different angular measurements, one from the absolute encoder of the motor shaft and another from the absolute encoder of the joint output. Thus, this technique uses both absolute positions to calculate the controller's output. Additionally, to reduce the noise from the sensor readings and to improve the estimation of the variables that are not known directly, a state observer was implemented, which improves the system's performance. By applying the observer, the settling time can be decreased, as well as the trajectory and positioning errors reduced. For this joint, the considered reference signal is the absolute position of the joint output. The block diagram for the implemented controller is presented in Figure 4.42.

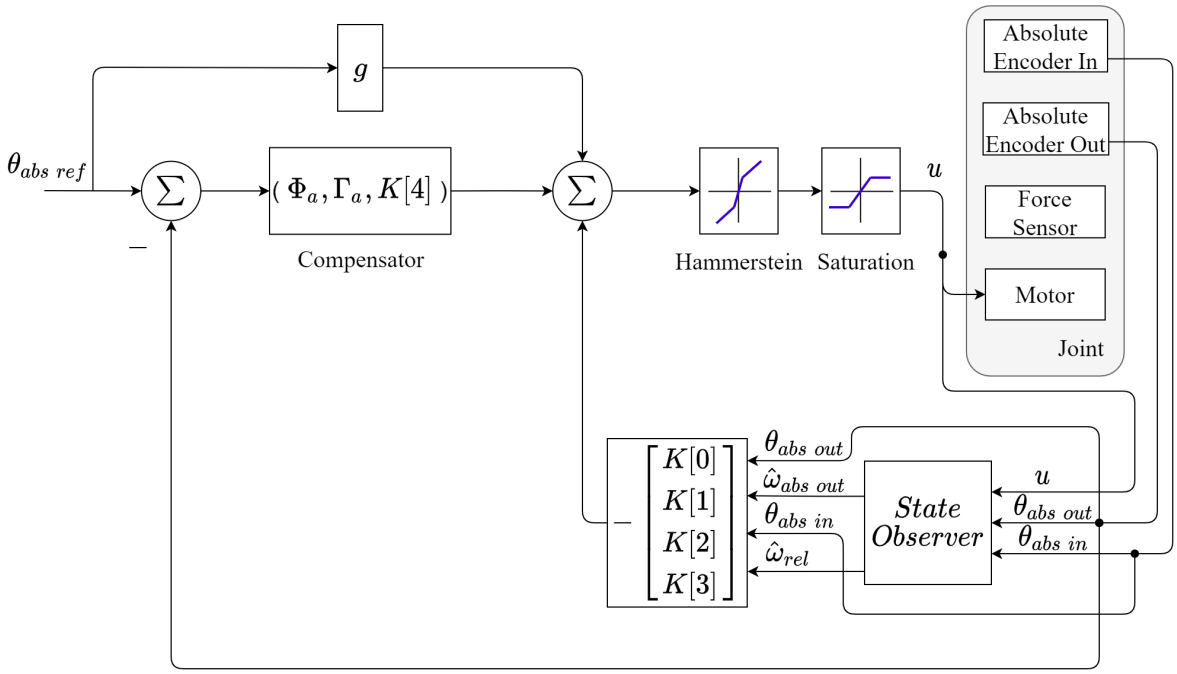


Figure 4.42: State-space Controller Block Diagram (Non-Rigid Joint).

The designed controller is capable of following a step input, so it was decided that an additional dynamic will be introduced in the system to enable this feature. Another plus is that this additional dynamic also makes the system capable of dealing with disturbances of the same kind—unitary step disturbances. To add this to the state-space model, $\mathbf{\Phi}_a$ and $\mathbf{\Gamma}_a$ matrices must be defined as:

$$\mathbf{\Phi}_a = \begin{bmatrix} -\delta_1 & 1 & 0 & \dots & 0 \\ -\delta_2 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\delta_{s-1} & 0 & 0 & \dots & 1 \\ -\delta_s & 0 & 0 & \dots & 0 \end{bmatrix}, \mathbf{\Gamma}_a = \begin{bmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \delta_{s-1} \\ \delta_s \end{bmatrix} \tag{4.77}$$

$$\mathbf{\Phi}_a = [1]\ ,\ \mathbf{\Gamma}_a = [1] \tag{4.78}$$

Now, a Bessel polynomial with a stabilizing time T_s will be used, as presented in Equations (4.79)–(4.81).

$$p_{12} = \frac{(a \pm jb)}{T_s} \tag{4.79}$$

$$p_{34} = \frac{(c \pm jd)}{T_s} \tag{4.80}$$

$$p_5 = \frac{e}{T_s} \tag{4.81}$$

$$p_{12} = \frac{(-5.9268 \pm j3.0813)}{T_s} \tag{4.82}$$

$$p_{34} = \frac{(-4.1104 \pm j6.3142)}{T_s} \tag{4.83}$$

$$p_5 = \frac{-6.4880}{T_s} \tag{4.84}$$

Using the Ackermann's formula [105], the controller constant vector $\mathbf{K}_s$ can be found.

$$\Delta(s) = (s^2 - s2a + a^2 + b^2)(s^2 - s2c + c^2 + d^2)(s - e) \tag{4.85}$$

$$\Delta(\mathbf{\Phi}) = (\mathbf{\Phi}^2 - \mathbf{\Phi}2a + a^2 + b^2)(\mathbf{\Phi}^2 - \mathbf{\Phi}2c + c^2 + d^2)(\mathbf{\Phi} - e) \tag{4.86}$$

Finally, the calculated controller gains are obtained and introduced in the system. These gains are presented below in a matrix form.

$$\mathbf{K}_s = \begin{bmatrix} 10.7110 \\ 0.5160 \\ -2.3263 \\ 0.4410 \\ 0.3118 \end{bmatrix} \tag{4.87}$$

As can be validated by the results presented in Figures 4.43, 4.44 and 4.45, the controller provides satisfactory results, that will be evaluated individually. The overshoot is not significat, thus it can be neglected.

Considering a reference of 0.5 rad, and with an initial position of 0 rad, the obtained rise time was 0.35 s and the settling time 0.64 s. In the downward direction, the rise time was 0.34 s and the settling time was 0.60 s.

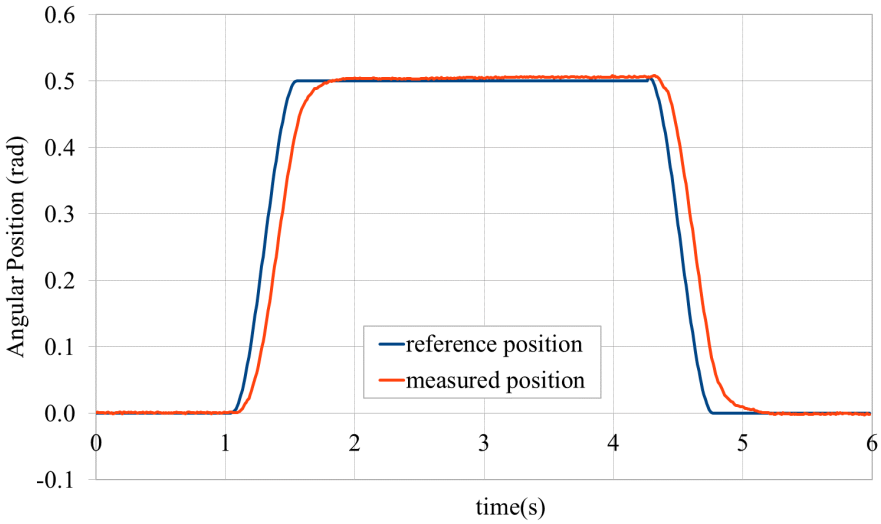


Figure 4.43: Non-Rigid Joint Controller (θ_{ref} = 0.5 rad).

Considering a reference of 1 rad, and with an initial position of 0 rad, the obtained rise time was 0.47 s and the settling time 0.82 s. In the downward direction, the rise time was 0.47 s and the settling time was 0.80 s.

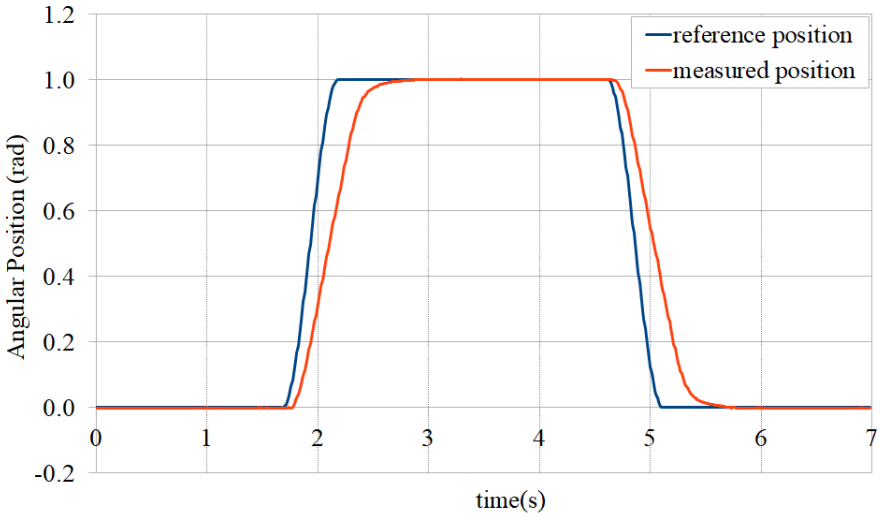


Figure 4.44: Non-Rigid Joint Controller (θ_{ref} = 1 rad).

Finally, considering a reference of 1.5 rad, and with an initial position of 0 rad, the obtained rise time was 0.68 s and the settling time 1.02 s. In the downward direction, the rise time was 0.67 s and the settling time was 1.01 s.

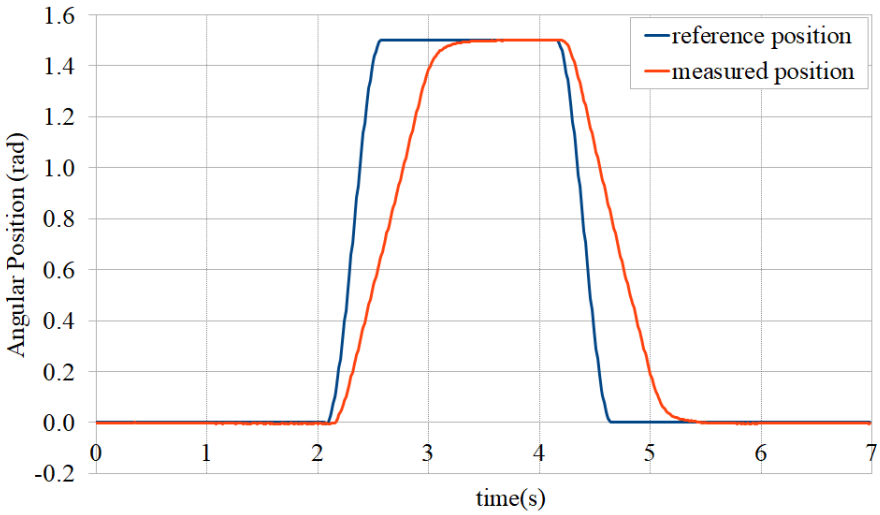


Figure 4.45: Non-Rigid Joint Controller (θ_{ref} = 1.5 rad).

4.5 Conclusion

Throughout chapter 4, all the core components used to build the system were described. The off-the-shelf elements that were acquired were briefly presented, while the elements that were custom designed for final solution were detailed. Namely, both linear and non-linear models of a geared DC motor were introduced and developed. Concerning the rigid and non-rigid robotic joints, models with and without non-linear elements were deduced and presented. Linear (or linearized) models were used for the design and implementation of the studied controllers, while non-linear models were used in the realistic simulation. The parameters for the motor, rigid and non-rigid joints were estimated. Furthermore, those models were compared with physical experiments, obtaining successful results. Several control approaches were used on the non-rigid, namely classical position PID and compliant controllers. The performed tests were detailed and the obtained data were presented. The parameters for the state-space controllers were calculated, and the control was also successfully tested through physical experiments. The obtained results confirmed the feasibility of the proposed approach.

Our research contributions regarding all the mentioned fields were published in multiple articles.

The following three publications are specially focused on detailing and analyzing the motor model, in both linear and nonlinear versions. Moreover, these present the design and implementation of dedicated controllers, including the results of carried out tests.

Pinto, V. H., José A. Gonçalves, and Paulo Costa. ”Introduction to DC motors for engineering

students based on laboratory experiments.” 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT). IEEE, 2019.

Pinto, V. H., José Gonçalves, and Paulo Costa. ”Model of a DC motor with worm gearbox.” Portuguese Conference on Automatic Control. Springer, Cham, 2020.

Pinto, V.H.; Gonçalves, J.; Costa, P. Modeling and Control of a DC Motor Coupled to a Non-Rigid Joint. Appl. Syst. Innov. 2020, 3, 24. https://doi.org/10.3390/asi3020024.

The two next publications detail the use of non-rigid robotic joints, namely their design, model and control techniques. Physical tests were performed that allowed the estimation of all joint parameters, and state-space controllers were designed and implemented for the developed system.

Pinto, V. H., et al. ”Modeling of an Elastic Joint: An Experimental Setup Approach.” Portuguese Conference on Automatic Control. Springer, Cham, 2020.

Pinto, V.H.; Gonçalves, J.; Costa, P. Towards a More Robust Non-Rigid Robotic Joint. Appl. Syst. Innov. 2020, 3, 45. https://doi.org/10.3390/asi3040045

CHAPTER 5

Proposed Leg for Legged-Wheeled Locomotion System

"Normal people believe that if it ain't broke, don't fix it. Engineers believe that if it ain't broke, it doesn't have enough features yet."

— Scott Adams

In chapter 4, the design, modeling and control of each part that constitutes the leg were presented individually. Throughout this chapter, the proposed solution for a single leg will be presented, as well as used hardware and developed software. Some x^l, z^l trajectory tests were performed, having obtained the detailed results.

5.1 Relevant Notation

It is considered convenient to define and clarify the notation that will be used in the present chapter.

The subscript letter j will be used to identify the variables referring to the joint.

The superscript index w represents the wheel.

j varies between 0 and 2 to identify each joint.

Each leg uses the cartesian space variables x^l and z^l, and angular position variable θ_0^l (since the leg orientation relates directly and exclusively with joint 0).

The remaining joints use variables θ_j^l.

The variables x^l and z^l are positioned in the center of the wheel.

The wheel's angular speed is defined as ω_3^l.

5.2 Design

Considering all the requirements described in Section 1.4, the symbolic representation of the idealized leg is present in Figure 5.1, consisting in three revolute joints and a driving wheel on the leg's bottom. The presented leg design is innovative, considering the legged-wheeled context, since the leg does not use a horizontal rotation axis along *x* in the hip joint, like the platforms existent in the literature. The presented approach uses a vertical axis, along *z*, that allows the leg to rotate, therefore allowing a purely-wheeled configuration similar to a classic mobile robot.

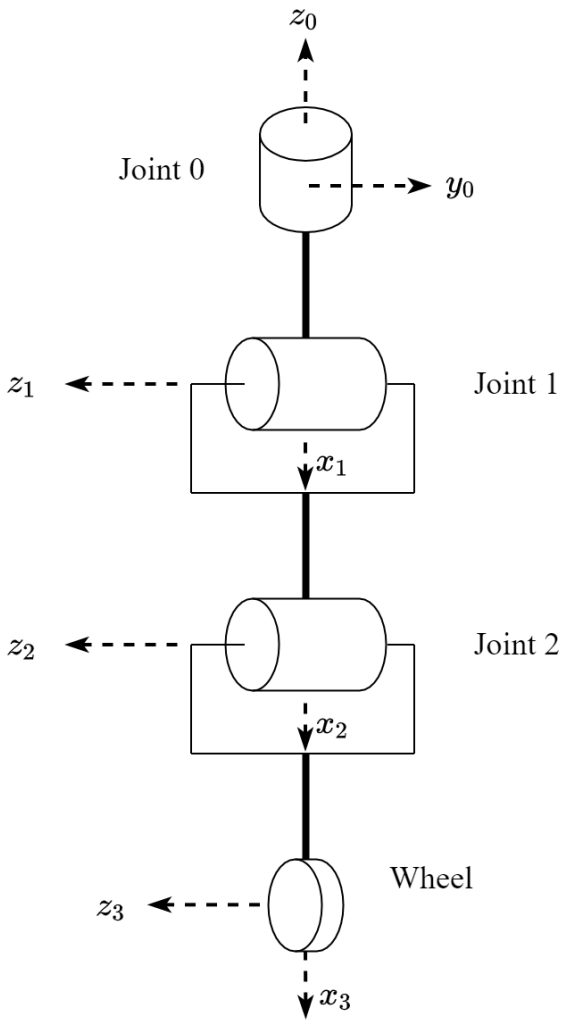


Figure 5.1: Symbolic Representation of the Leg.

5.2.1 Torque Sensing Link

Since it is necessary to accurately sense the forces exerted on each leg link, it was decided to divide it in half and use a load cell to couple both. With this configuration, there are two complementary measures of the force exerted on the leg. Connecting each of the L2 and L3 semilinks, two load cells —TAL220— of 10 kg are mounted, thus allowing to measure the force of each link in real time. The controller uses this information to close the loop for the compliance mode. To read the applied force, an Amplifier — SEN 13879 load cell amplifier board — is used for each cell, which uses the HX711 integrated circuit. Figure 5.2 illustrates the mentioned system.

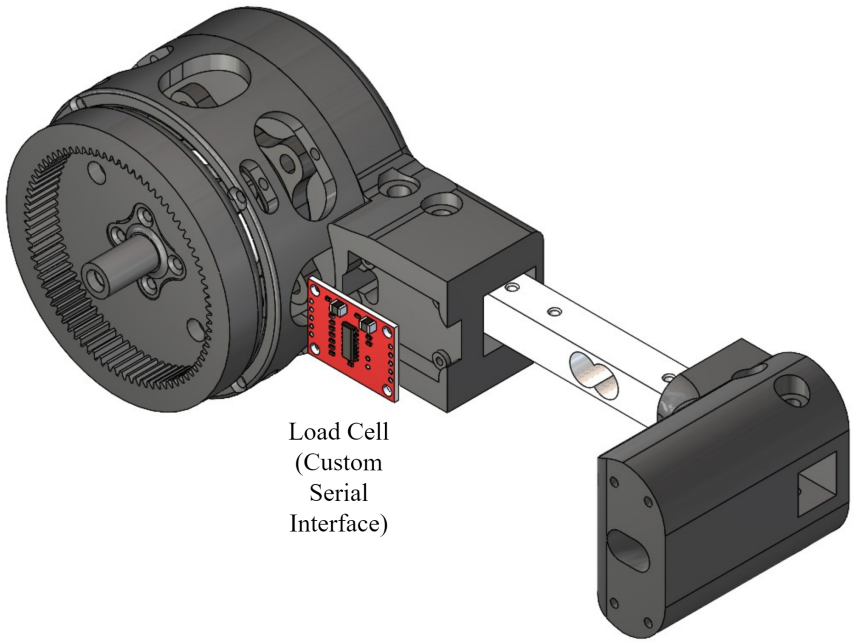


Figure 5.2: Load Cell Connections.

5.2.2 Prototype Leg *Foot*

To fulfill the requirement of having a low-power configuration, in which the vehicle can navigate with a substantially reduction of its energy consumption, it was decided to design the last link with an inclination, as well as to use an omni-directional wheel. This allows the leg to rest on the ground, reducing the consumption of the two central motors to zero, navigating with only the wheel rotation motor and the leg steering motor. For this purpose, an omni-directional wheel was added, allowing that position to be passively stable. In this traction configuration, the platform will have four driving

wheels and four free wheels on the ground. Although they have a fixed axle, when using omni-wheels, it allows the vehicle, by rotating its legs, to travel omnidirectionally. At the bottom of each leg, it was decided to use rubber wheels with a 100 mm diameter, since smaller wheels reduce the angle of action of the leg. This is explained since the wheel support easily started to hit the floor when the leg is rotated, although with a low angle. For the alignment and support of the wheel, the same concept of the joints was used, differentiated by the use of a smaller (12 mm) diameter bearing (6001 2Z). The mechanical explosion of the mentioned is presented in Figure 5.3.

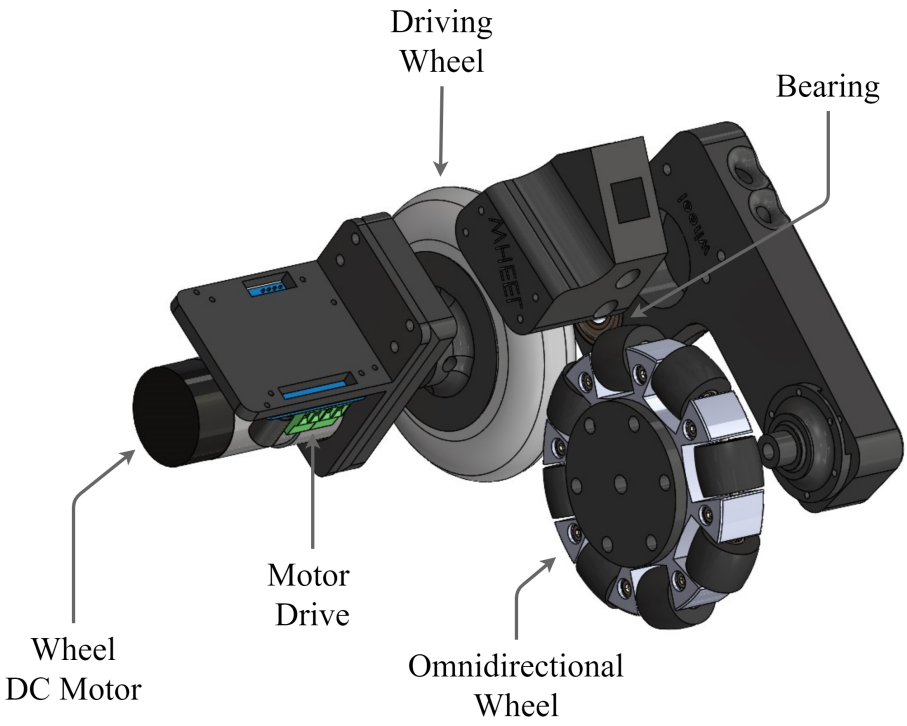


Figure 5.3: Mechanical Explosion Assembly Wheel.

5.2.3 Leg

The final designed solution, with the respective joint numbering and rotation axis is presented in Figure 5.4, while Figure 5.5 presents the improved leg configuration.

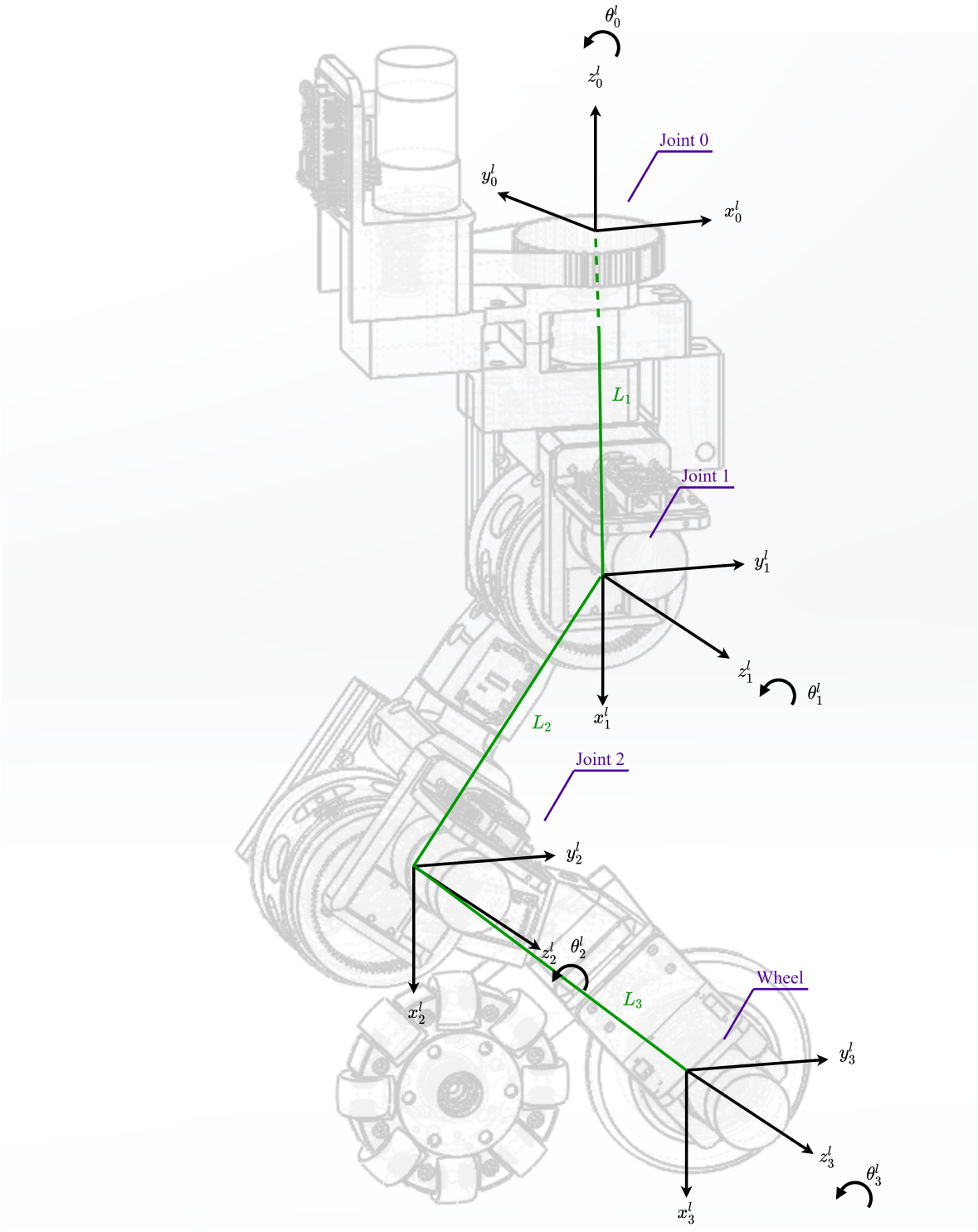


Figure 5.4: Leg with axes, links, and joints numbering.

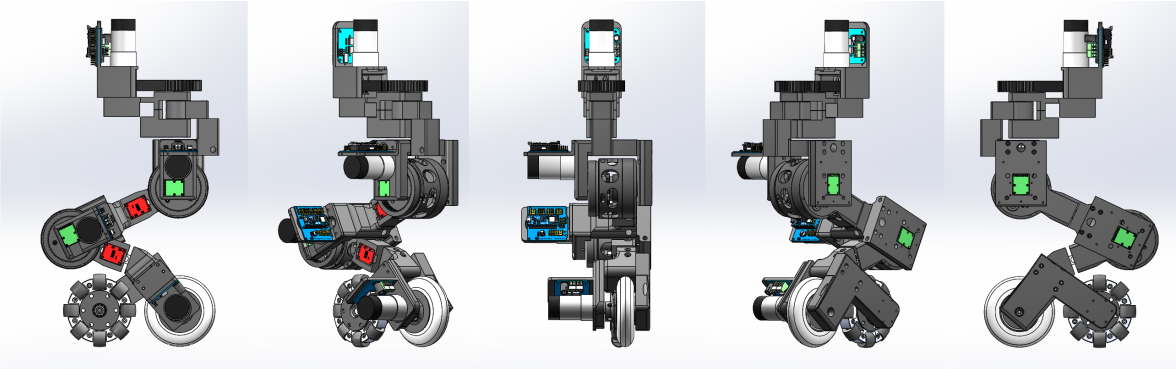


Figure 5.5: Improved Leg Configuration (front left leg).

Regarding the electrical design, there are some considerations to take into account. First of all is the type of motor to be applied. Considering that it is an objective to use the vehicle in autonomous mode, as well as the use of Series Elastic Actuators (SEA), it was decided to use DC motors with a gearbox. Specifically, the motor used for the hip joint is the CQR37D64CPR70GR, with a gearbox ratio of 70:1 and top speed of 157 RPM. Joints 2 and 3 use the same type of motor —CQRGL20180525-52GB— with ratio of 210:1 and top speed of 52 RPM. Finally, for the wheel, motor —CQR37D64CPR43.8GR— with a 43.8:1 gearbox with maximum speed of 251 RPM was used. Each motor uses its own drive, being that joint 0 and wheel use the BTS7960, enabling the control of motors up to 30 V and 43 A. Joints 1 and 2 use the Adafruit 3190 DRV8871 drive, with a maximum of 3.6 A.

In order to facilitate connections, but also because the number of needed I/O ports rapidly becomes large due to the use of a high number of sensors, each leg uses three microcontrollers. This approach also reduces the length of the wires that connect the sensors to the I/O ports.

The hip motor uses one microcontroller, joint 1 has another, while the set of the two motors of joint 3 and wheel connect to the third. The hip joint only uses one absolute encoder, as well as the motor drive. Joint 1 uses both encoders, plus the load cell. Both encoders from joint 3 and the load cell of the L3 link are connected to the microcontroller that is in the adapter of joint 3. Figure 5.6 shows where the microcontroller boards are located in the leg.

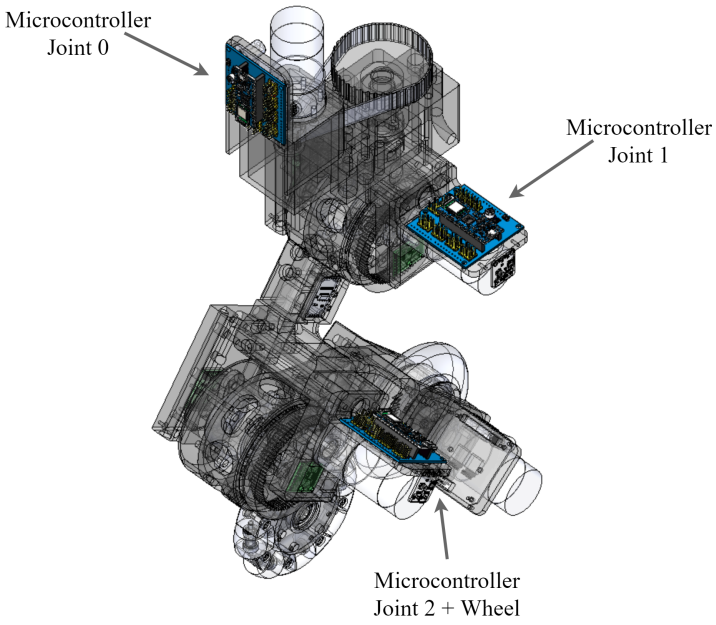


Figure 5.6: Microcontroller Boards' Location on the Leg.

All the microcontrollers (Arduino Nano 33 IOT) from each leg connect with a Teensy 4.1 board, which is the robot main microprocessor. This board has increased processing power, number of available serial ports, as well as I/O pins.

The flash memory of the NINA W102 ESP32 module that the Arduino Nano 33 IOT has in-built was used to save each joint position on the robot. For example, joint 1 of the leg 0 was numbered *joint10*, being possible to identify which are the low, middle or top level joints. In the flash memory are also saved the parameters associated with each joint, namely absolute encoder offsets, controller tuning parameters, among others.

As a safety measure, Reed effect end-switches were included in each of the non-rigid joints, preventing them from hitting the mechanical limit. The sensors used for this function were the LittelFuse 59025, and the respective 57025 magnets.

Most calculations are performed in the main microcontroller (Teensy), such as calculating direct and inverse kinematics, angular and speed references for each of the joints, as well as the smoother trajectories of each of the joints. Additionally, it is also in this board that the robot's kinematics is calculated for its trajectory tracking mode.

With this configuration, the joint's microcontrollers are only responsible for receiving the position and speed references, calculate them through the respective controllers, and defining the PWM values

of the respective motors.

Figure 5.7 illustrates the mentioned hardware, as well as their positioning in the leg, and in the joint.

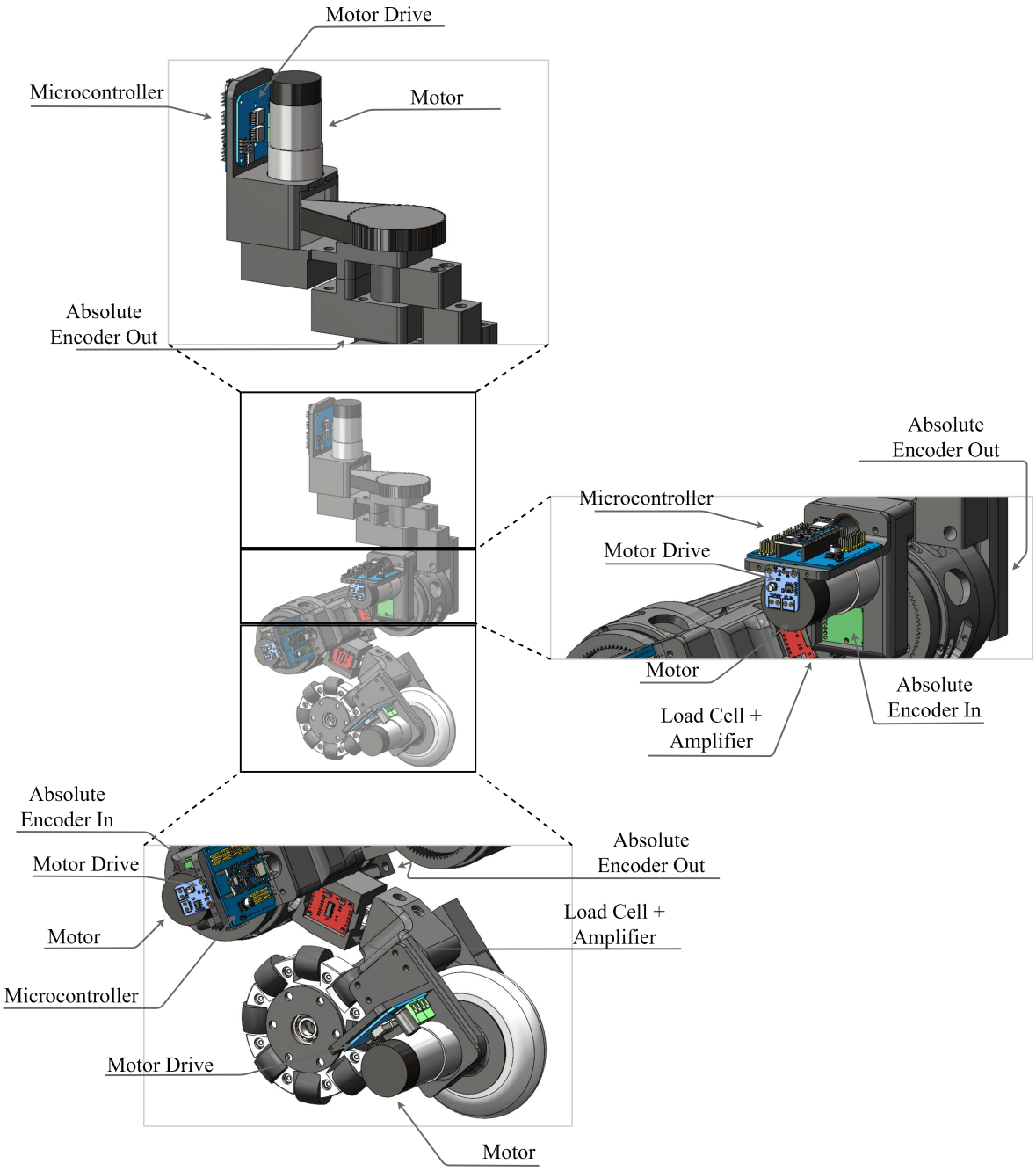


Figure 5.7: Single Leg Hardware Diagram.

Finally, in order to synchronize both microcontrollers, since the implementation of the inverse

kinematics for the leg movement requires that the commands are sent for all the joints at the same time, serial port 1 (set on RX/TX pins) is used. For this, a connection between the two pins from each microcontroller is performed. These connections are further detailed in Figure 5.12.

An obstacle surpassing situation is depicted in Figure 5.8. It is noticeable that this configuration is capable of handling horizontal and vertical impacts, as well as floor irregularities. The only configuration where this joint does not exhibit its vertical absorption capability is when it is fully stretched.

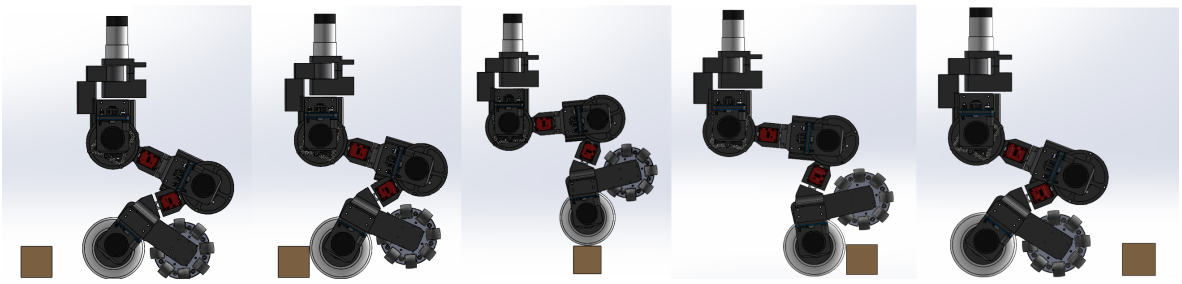


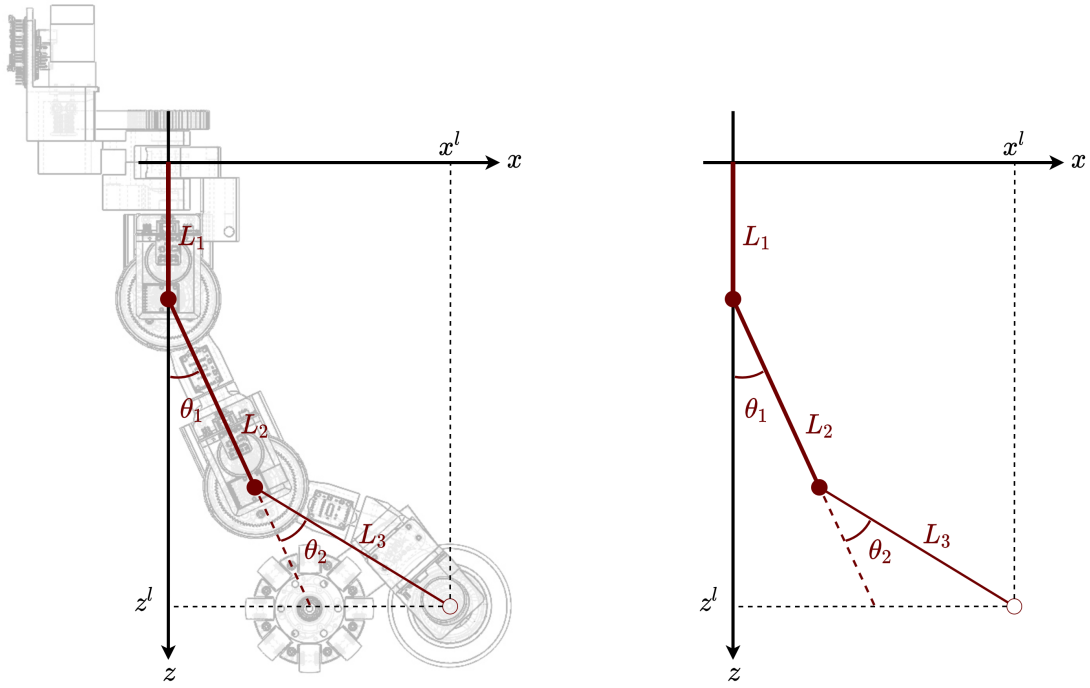
Figure 5.8: Obstacle surpassing simulation (left to right).

5.3 Control

Having already covered the control techniques used for each joint individually, this chapter will focus on the control of the wheel positioning. Thus, the control and trajectory following algorithms that were used to control the leg in x^l, z^l will be presented.

5.3.1 Direct Kinematics

Knowing the data from the angular positions of a manipulator, the Denavit–Hartenberg method (DH) of direct kinematics can be used to obtain the values of the x^l and z^l points from the tip. Since the developed leg can be compared to an anthropomorphic manipulator, this method will be used to discover the x^l, z^l coordinates of the wheel with respect to the measured θ_1 and θ_2 angles.



(a) (b)

Figure 5.9: Direct Kinematics Representations.

Considering the constants illustrated in Figure 5.9, Table 5.1 presents the respective DH parameters.

Table 5.1: DH Parameters for the Leg.

Link	a_i	α_i	d_i	θ_i
1	0	$\frac{\pi}{2}$	0	θ_1
2	L_2	0	0	θ_2
3	L_3	0	0	θ_3

The coordinate transformation matrix obtained through the DH method is shown in Equation (5.1).

$$T_3^0\begin{pmatrix}\theta_0\\ \theta_1\\ \theta_2\end{pmatrix}=\begin{bmatrix}\cos\theta_0\cos(\theta_1+\theta_2) & -\cos\theta_0\sin(\theta_1+\theta_2) & \sin\theta_0 & \cos\theta_0(L_2\cos\theta_1+L_3\cos(\theta_1+\theta_2))\\ \sin\theta_0\cos(\theta_1+\theta_2) & -\sin\theta_0\sin(\theta_1+\theta_2) & -\cos\theta_0 & \sin\theta_0(L_2\cos\theta_1+L_3\cos(\theta_1+\theta_2))\\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & 0 & L_2\sin\theta_1+L_3\sin(\theta_1+\theta_2)\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{5.1}$$

Despite being a very useful approach for several situations, considering that the calculations will be performed in a microcontroller, it was decided to obtain the following geometric equations, which are computationally less demanding:

$$x_d = L_2\sin\theta_1 + L_3\sin(\theta_1+\theta_2) \tag{5.2}$$

$$x^l = x_d\cos\theta_0 \tag{5.3}$$

$$y^l = x_d\sin\theta_0 \tag{5.4}$$

$$z^l = L_1 + L_2\cos\theta_1 + L_3\cos(\theta_1+\theta_2) \tag{5.5}$$

The length of each leg link is measured and presented in Table 5.2.

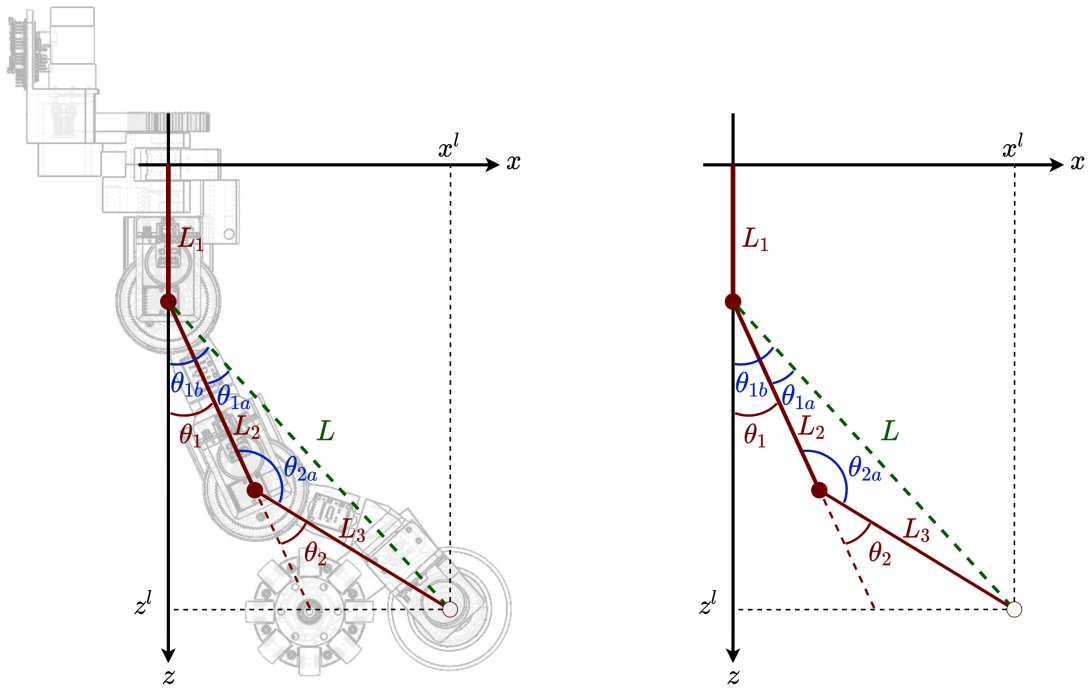
Table 5.2: Lengths of the Leg Links.

Link	**Length (m)**
L_1	0.130
L_2	0.168
L_3	0.186

Positioning the wheel in a given x^l, z^l position is achieved using the mentioned direct kinematics technique. However, it is necessary to consider that, on the robot, this positioning is directly related to the angle θ_0.

5.3.2 Inverse Kinematics

On the other hand, to obtain the appropriate angular positions for the wheel to reach the desired x^l, z^l position, it is necessary to apply the inverse kinematics.



(a) (b)

Figure 5.10: Inverse Kinematics Representations.

The respective angular positions can be obtained through the set of equations presented below, that were deduced using the representation shown in Figure 5.10.

To calculate the position of the wheel x_d, and the θ_0 angle of the upper joint, Equations (5.6) and (5.7) are used.

$$d = \sqrt{(x^{l^2} + z^{l^2})} \tag{5.6}$$

$$\theta_0 = \text{atan2}\,(z^l, x^l) \tag{5.7}$$

Then, the auxiliary length L is obtained

$$L = \sqrt{(z^l - L_1)^2 + d^2} \tag{5.8}$$

The auxiliary angles θ_{1b} and θ_{1a} are also calculated through Equations (5.9), (5.10) and (5.11), thus obtaining θ_1.

$$\theta_{1b} = \tan^{-1}\left(\frac{z^l - L_1}{x^l}\right) \tag{5.9}$$

$$\cos\theta_{1a} = \frac{-L_3^2 + L_2^2 + L^2}{2LL_2} \tag{5.10}$$

$$\theta_{1a} = \cos^{-1}\left(\frac{-L_3^2 + L_2^2 + L^2}{2LL_2}\right) \tag{5.11}$$

$$\theta_1 = \theta_{1b} - \theta_{1a} \tag{5.12}$$

Then, the auxiliary angle θ_{2b} is calculated through Equations (5.13) and (5.14), obtaining θ_2 through Equation (5.15).

$$\cos\theta_{2a} = \frac{-L^2 + L_2^2 + L_3^2}{2L_2L_3} \tag{5.13}$$

$$\theta_{2a} = \cos^{-1}\left(\frac{-L^2 + L_2^2 + L_3^2}{2L_2L_3}\right) \tag{5.14}$$

$$\theta_2 = \pi - \theta_{2a} \tag{5.15}$$

Finally, by applying a profile generator to the desired x^l, z^l positions, as mentioned before, and using the previously mentioned Inverse Kinematics implementation as input to the previously presented controllers solutions, the system can now produce the desired wheel positions. The full controller block diagram is presented in Figure 5.11.

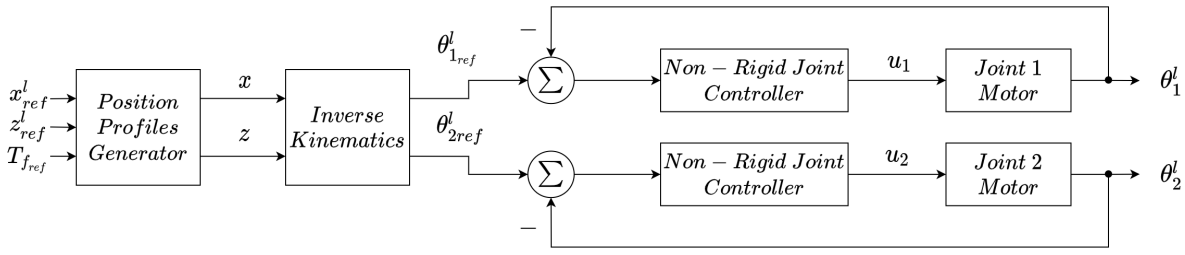


Figure 5.11: Leg Control Block Diagram.

5.4 Software

This chapter is dedicated to exploring the developed software for the leg. Furthermore, the communication system and message flow is also detailed.

5.4.1 Communication System

Regarding the communication, it was decided to use a serial protocol. The adopted communication strategy for the vehicle is a mix of direct communication from the Teensy to each one of the legs' Arduinos (TX to all RX) and a bus mode (Arduinos TX in open collector mode to Teensy RX).

To communicate with the microcontrollers placed on the joints, the board uses 4 dedicated Serial ports, each one pre-defined to a leg, which allows the software to always know to/from which leg it is transmitting/receiving messages. In each leg, a serial bus is implemented, through an adapter that allows for the hardware connections to be performed, as well as to protect the input output pins. Figure 5.12 presents the diagram that demonstrates the physical implemented connections, for one leg.

In the diagram, resistors $R_{i\,TX}$ and $R_{i\,RX}$ provide some over-voltage protection and, together with the input capacitance, producing a low-pass filter. The diodes D_i transform the TX lines in open collector, allowing a wired-end on this line. R_T is a pull-up resistor used in the RX line, keeping it at the high digital value.

Figure 5.13 presents the information flow in each microcontroller, also for one leg.

The Teensy 4.1 board uses the serial port to communicate with the Arduino boards, bi-directionally, using the defined message structure. The communication protocol follows the defined structure where the message is composed by a channel — defined as a upper case letter from G to Z — and a sub-channel — lower case letter from g to z. The letters contained in the American Standard Code for Information Interchange (ASCII) code are not used — *A* to *F* and *a* to *f*. After these, there are 4 bytes (32 bits) used to exchange the desired information. Finally, a byte referring the message sender or receiver can be included, depending on the information flow direction.

Tables 7.1 to 7.12, placed in Appendix 7.3 specifies the used protocol messages in each direction.

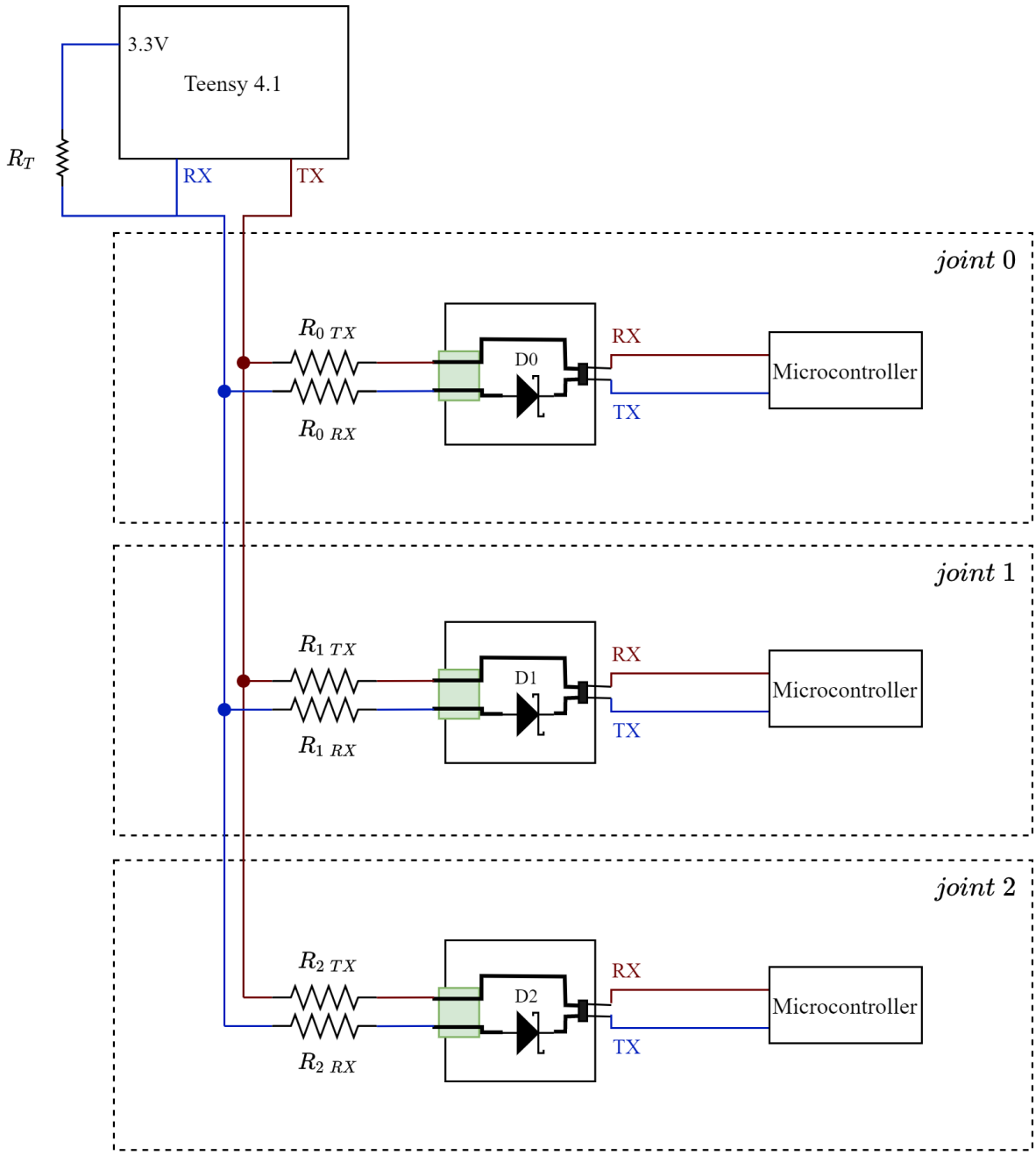


Figure 5.12: Communication Diagram.

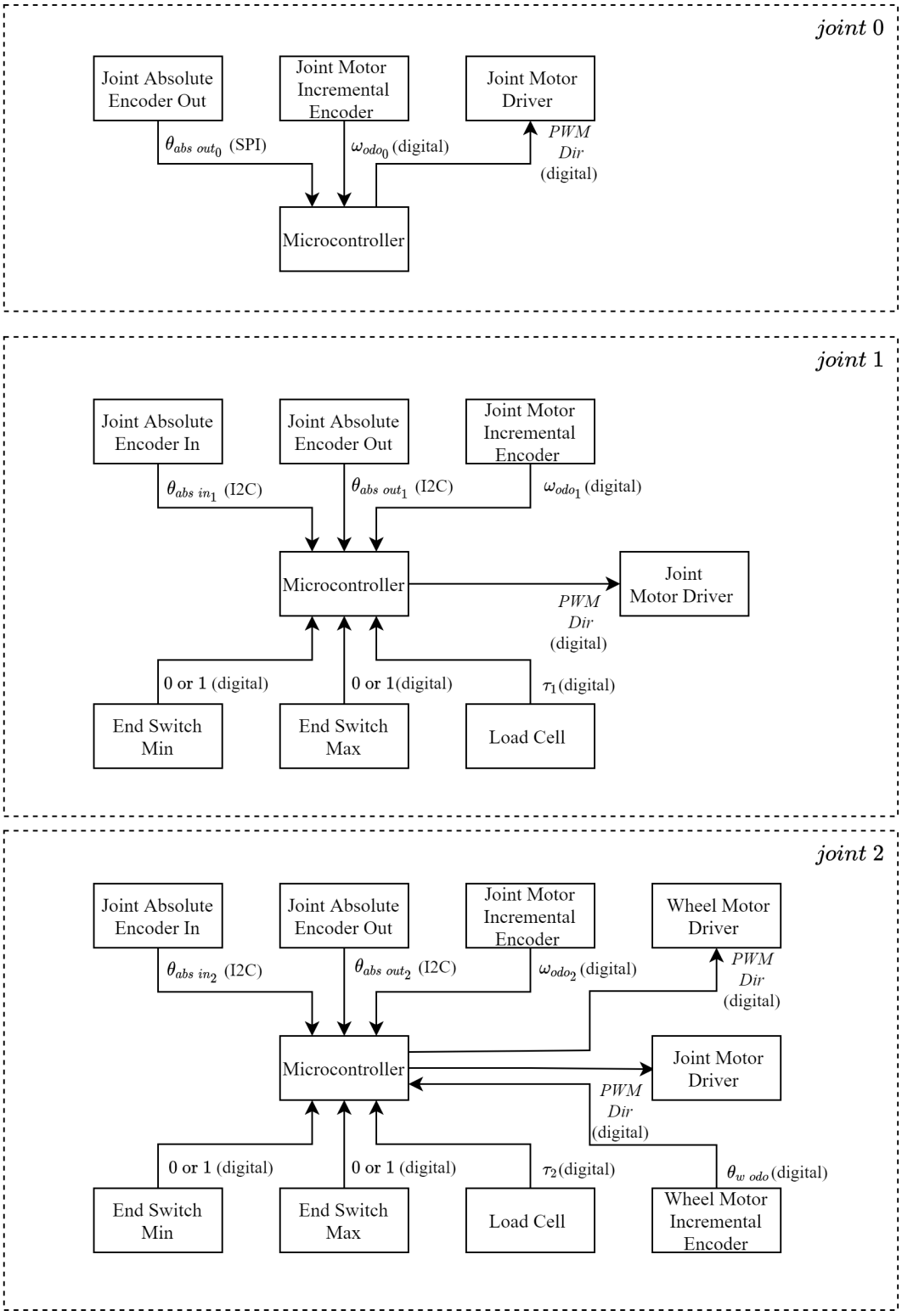


Figure 5.13: Information Flow Diagram.

5.5 JointTune — Single Joint Software

In addition to the serial communication through the General Purpose Input/Output (GPIO), the USB port can also be used, by connecting it directly to the Teensy board, through the software Joint Tune (presented in Figure 5.14), specifically created for this purpose. One of the reasons for developing this software was to test each of the joints individually. This allows to obtain, for example, local parameters or graphics from each of the joints. Another reason is quickly debug one of the hardware components — the DC motor, for example — enabling local tests without the need to involve the entire system. This software was very important to collect the required data to obtain the joint and motor models that were previously presented. Another important task enabled by this software is the controller tuning and testing.

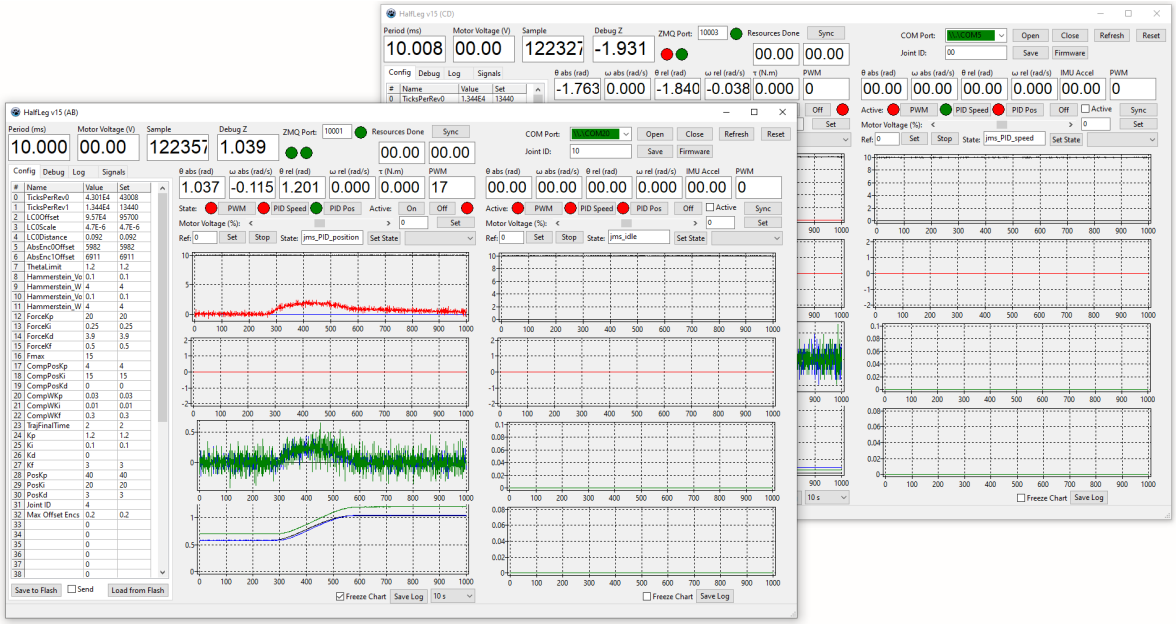


Figure 5.14: Joint Tune.

5.6 Trajectory Following

In order to test the controllers and validate the results obtained in simulation, a test set of x^l, z^l points was defined that was applied to the leg trajectory planner, being tested the points that are shown in Table 5.3.

Graphically, Figures 5.15 to 5.17 illustrate the leg's physical behavior when some test sets of reference points are applied to the system. As the graphical representations demonstrate, the desired points can be reached, although with some associated error. Analyzing more specifically each of the tests carried out, it is possible to obtain the following errors presented in Tables 5.4 and 5.5, namely the average and maximum absolute errors in x^l, z^l, θ_1 and θ_2, as well as the maximum error of each in steady state (SS).

Table 5.3: Set of x^l, z^l positions.

Test Set	**Point**	x^l **(m)**	z^l **(m)**	T_f **(s)**
1	1	0	0.3	n/a
	2	−0.2	0.3	2
	3	0.2	0.3	2
	4	0	0.3	2
2	1	−0.2	0.3	n/a
	2	0	0.3	0.5
	3	−0.2	0.3	0.5
	4	0.2	0.3	0.5
	5	0	0.3	0.5
3	1	−0.2	0.35	n/a
	2	0.3	0.35	1
	3	−0.2	0.35	1

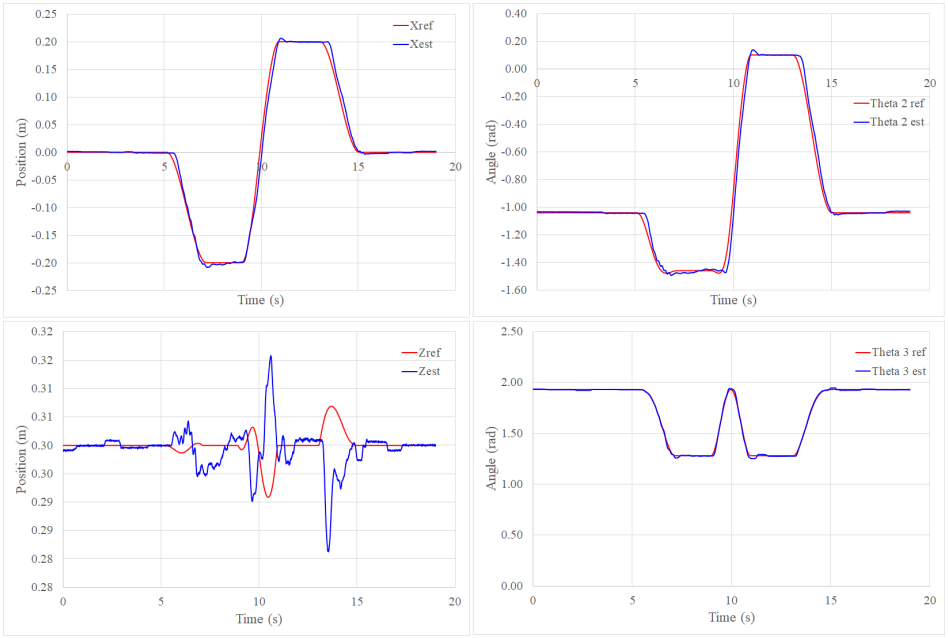


Figure 5.15: Trajectory Following Results—Test Set 1.

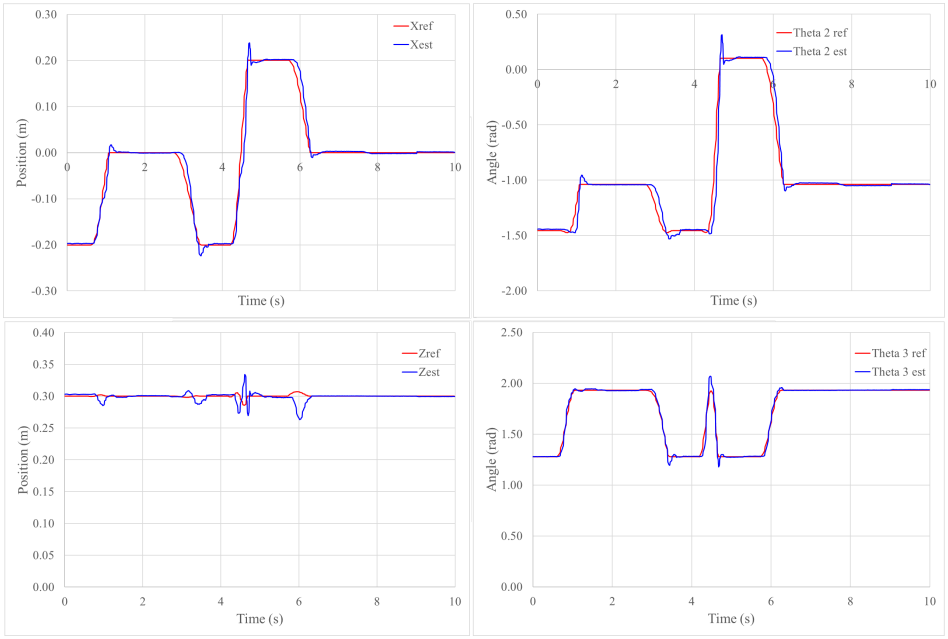


Figure 5.16: Trajectory Following Results—Test Set 2.

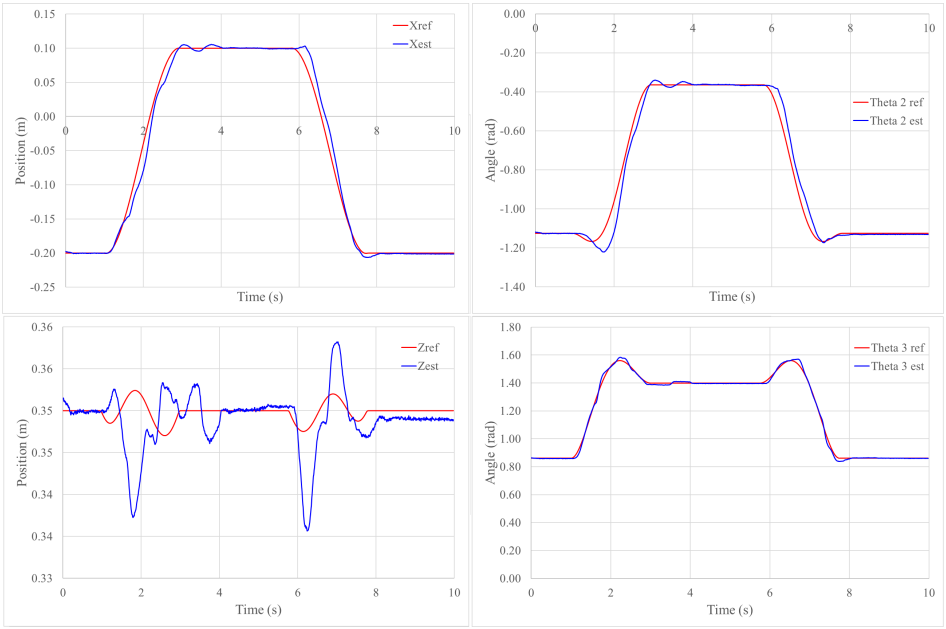


Figure 5.17: Trajectory Following Results—Test Set 3.

Evaluating the registered error values, the maximum average error in positioning in x^l is 0.00042 m, in z^l is 0.00046 m, in θ_1 is 0.00207 rad, and in θ_2 is 0.00016 rad. Regarding the maximum absolute error values, there are some larger values, in x^l is 0.10211 m, in z^l is 0.04302 m, in θ_1 is 0.51481 rad, and in θ_2 is 0.14368 rad. Finally, the maximum errors in steady state are in x^l is 0.00367 m, in z^l is 0.00238 m, in θ_1 is 0.02158 rad, and in θ_2 is 0.02268 rad. The errors average values are low, indicating that the maximum absolute error values are residual, and it can be analyzed that these values are obtained due to the overshoot of the system when the final time (T_f) is set to 0.5 s. In steady state the maximum errors are below 0.3 cm in x^l and z^l and lower than 0.02 rad in θ_1 and θ_2. Regarding the final time to reach the desired value, which is also a variable that can be changed, the three test sets use different values, confirming that the positioning can take different settling times. However, it is also noticeable that the error decreases when this time is increased, not requiring as much performance in the movement of the joints. Table 5.4 presents the errors of the tip x^l, z^l positioning, in m. Table 5.5 presents the θ_1, θ_2 angular positioning errors, in *rad*.

Table 5.4: Errors in x^l, z^l Positioning (in m).

Test Set	x^l **Avg**	z^l **Avg**	x^l **Max**	z^l **Max**	x^l **Max SS**	z^l **Max SS**
1	0.00005	0.00044	0.03541	0.02528	0.00367	0.00238
2	0.00042	0.00046	0.10211	0.04302	0.00251	0.00234
3	0.00011	0.00031	0.03992	0.01509	0.00127	0.0001

Table 5.5: Errors in θ_1, θ_2 Positioning (in *rad*).

Test Set	θ_1 **Avg**	θ_2 **Avg**	θ_1 **Max**	θ_2 **Max**	θ_1 **Max SS**	θ_2 **Avg SS**
1	0.00019	0.00011	0.18293	0.03915	0.02185	0.02268
2	0.00207	0.00016	0.51481	0.14368	0.01204	0.00800
3	0.00042	0.00000	0.15745	0.04164	0.00443	0.00307

5.7 Conclusions

Throughout the last chapter, the core elements that were described in chapter 4 were used to assemble a single leg. The leg's hardware structure, developed software and communication strategy were also presented. The equations for the direct and inverse kinematics calculation were described. Moreover, the control methods for the leg positioning were detailed. Some leg positioning tests were performed, and the results obtained were presented. Tests on x^l, z^l positioning obtained maximum steady-state errors of about 0.3 cm were obtained, while the errors in angular position of the joints θ_1, θ_2 remained below 0.02 rad.

The research contributions of the work contained in this chapter are published in the following article:

Pinto, V.H.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of a Single Leg for a Legged-Wheeled Locomotion System with Non-Rigid Joint. Actuators 2021, 10, 29 https://doi.org/10.3390/act10020029.

CHAPTER 6

Proposed Hybrid Legged-Wheeled Locomotion System

"Manufacturing is more than just putting parts together. It's coming up with ideas, testing principles and perfecting the engineering as well as final assembly."
— James Dyson

In chapter 5, the proposed solution for a single leg was presented. This chapter will focus on introducing the relevant aspects of the robot prototype, kinematics and developed software. The path following algorithms are detailed, along with the obtained results.

6.1 Relevant Notation

Some relevant notation was introduced in chapter 5. However, it is important to introduce extra notation that will be used in the present chapter.

The subscript letter r will be used to identify the variables referring to the robot.

The superscript indexes l and w represent the leg and wheel, respectively.

Regarding the positioning and orientation variables, the robot uses x_r, y_r and z_r as cartesian space variables, as well as α_r, β_r and γ_r (roll, pitch and yaw) as Euler angles.

For linear and angular speeds, the robot uses v_r, v_{n_r} and ω_r, respectively, while each wheel speed is represented by v_3^l (since the wheel speed relates directly and exclusively to joint 3).

Examples are the linear speed of the robot — v_r, or the angular position of the joint 0, of the leg 1 — θ_0^1.

l varies between 0 and 3 to identify the legs.

6.2 Design

A four legged prototype was used as testbed platform, and the final result is shown in Figure 6.1. Figure 6.2 represent the robot's position and orientation variables.

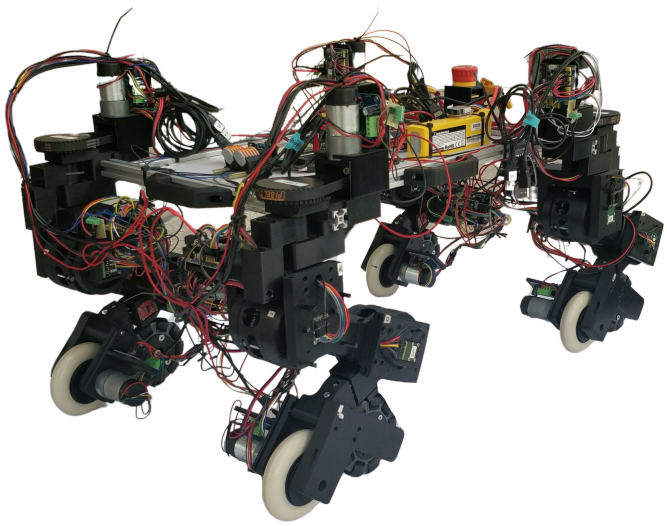


Figure 6.1: Hybrid Legged-Wheeled Robotic Vehicle.

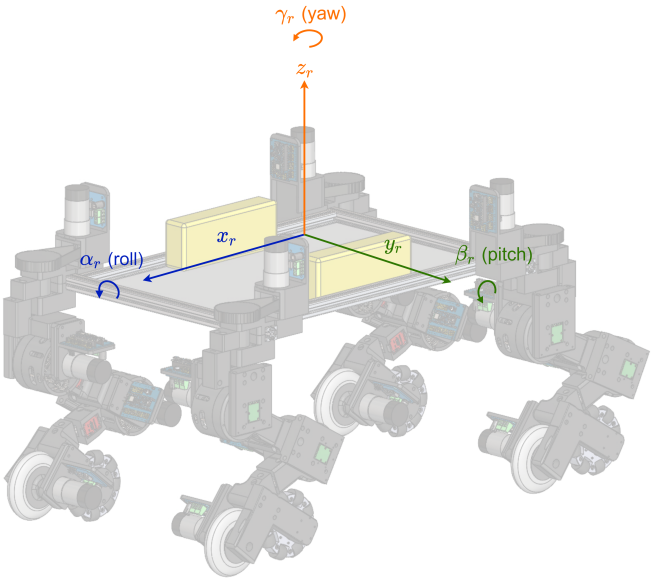
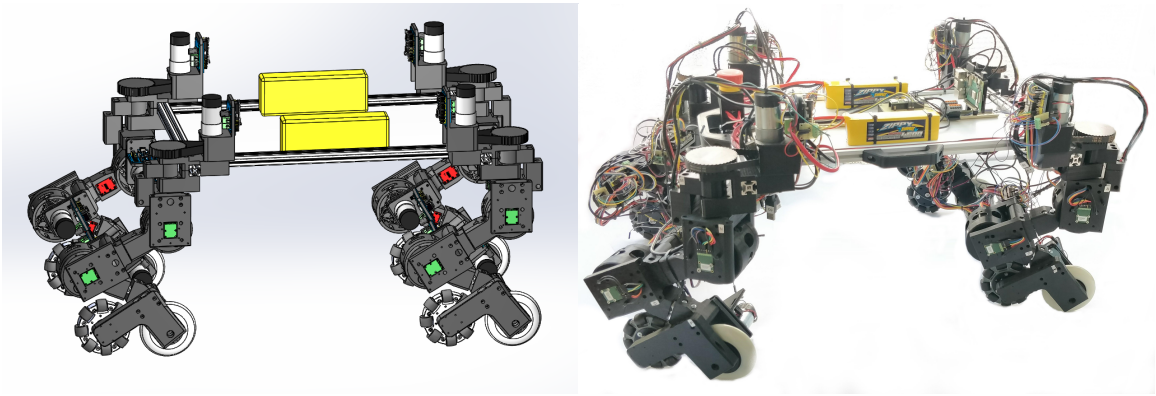


Figure 6.2: Robot Position and Orientation Variables.



(a) (b)

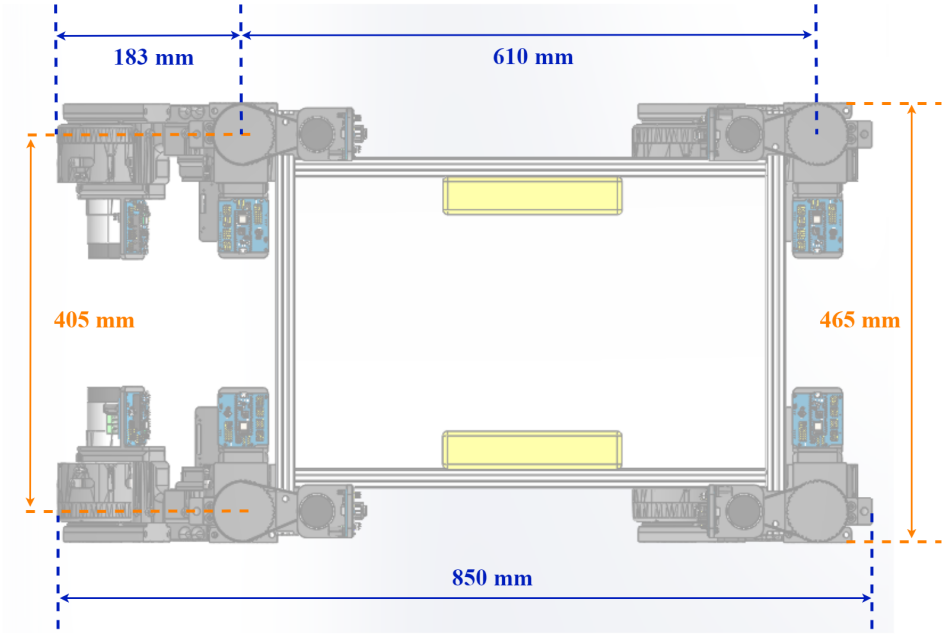
Figure 6.3: CAD Render and Real Robot

The platform that was developed to serve as a testbed is present in Figs. 6.3a and 6.3b, which show the robot's rendering in CAD as well as its real version, respectively.

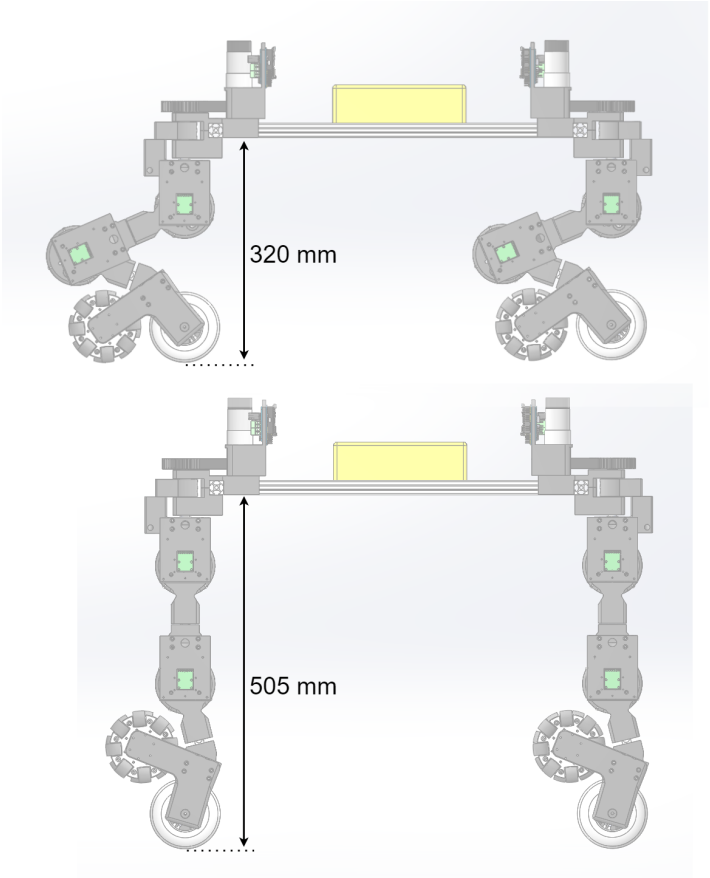
As can be seen by the diagram on Figure 6.4a, the robot's maximum dimensions are 850 $\times$ 465 mm (L $\times$ W). Figure 6.4b shows that its height can vary between 320 and 505 mm. In this configuration, the robot weighs approximately 18 kg. Table 6.1 aggregates all the mentioned information.

Table 6.1: Robot Dimensions.

Dimensions	
Length	0.85 m
Width	0.465 m
Mininum Height	0.320 m
Maximum Height	0.505 m
Weight	18 kg



(a)



(b)

Figure 6.4: Robot Dimensions.

6.3 Control

For the full robot control, the same control strategy is used, where the controllers receive a reference for a desired wheel position. The block diagram is present in Figure 6.5.

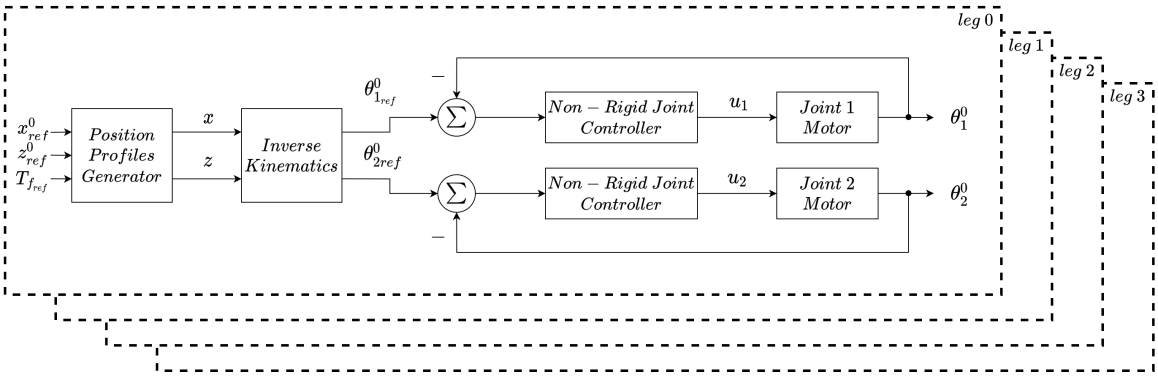


Figure 6.5: Wheels' x, z Position Joint Control Block Diagram.

Regarding the movement of the vehicle, there are three possible locomotion modes. In the purely wheeled mode, whether in low-power, or when it is at a defined height, the robot always tries to maintain the wheels' x^l axis aligned with the rotation axis of joint 0, for all the legs. This strategy prevents the wheel from dragging laterally when rotating the leg, guaranteeing that the contact point of the wheel with the floor does not move in y, rather rotating only on its axis.

The variables applied to calculate the robot kinematics are present in Figure 6.6. The variables a, b and d are the linear distances to the center of the robot. v_r and v_{n_r} represent the robot's linear speeds along the x and y axis, respectively, while ω_r represents its angular speed. For each wheel, a v_{i_w} speed is defined, as well as its orthogonal projections v_x and v_y. The angle θi_w is the wheel angular position, relative to the x axis.

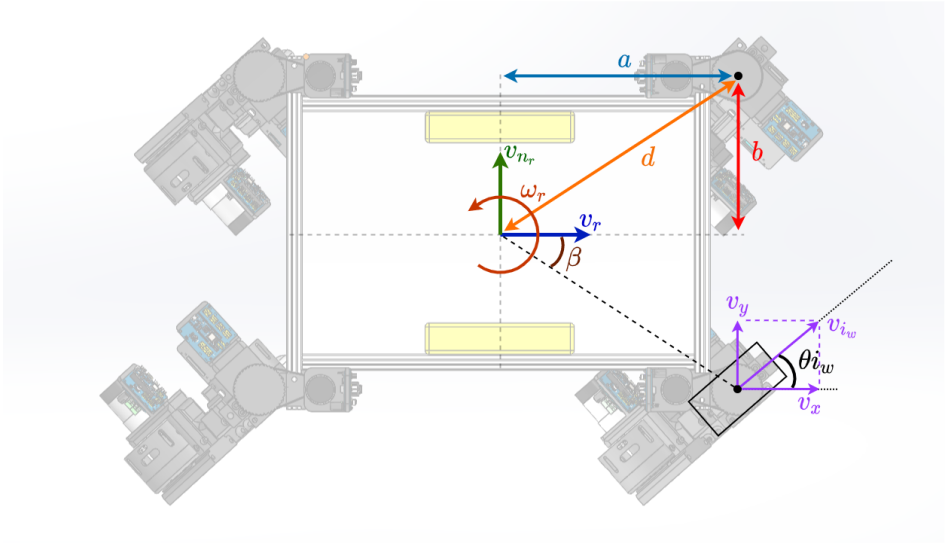


Figure 6.6: Robot Kinematic Variables.

Following are the equations relative to the robot kinematics, considering, as already mentioned, that the rotation axis of joint 0 is aligned with the rotation axis of the wheel, regardless of the robot's height. The variables to be controlled are each leg joint 0 angle, which defines the wheel's angle (θ_0^l), as well as the speed of the wheel (v_3^l).

Equations 6.1 to 6.10 present the relation between each wheel speed and the robot's v_r, v_{n_r} and ω_r speeds. Although equations 6.1 to 6.3 are straightforward, three computationally demanding functions were being used (*sin*, *cos* and *atan*2). So, equations 6.4 to 6.6 present a simplification, which makes it possible to decrease the computation cost. The only one that was not possible to avoid was *atan*2. The simplified equations are presented in Equations 6.7 to 6.10.

$$\beta = \text{atan2}(y,x) \tag{6.1}$$

$$v_x = v - \omega \cdot d \cdot \sin\beta \tag{6.2}$$

$$v_y = v_n + \omega \cdot d \cdot \cos\beta \tag{6.3}$$

Considering:

$$d = \sqrt{x^2 + y^2} \tag{6.4}$$

$$\sin\beta = \frac{y}{\sqrt{x^2 + y^2}} = \frac{y}{d} \tag{6.5}$$

$$\cos\beta = \frac{x}{\sqrt{x^2 + y^2}} = \frac{x}{d} \tag{6.6}$$

So, Equations (6.2) and (6.3) can be written as

$$v_x = v - \omega \cdot y \tag{6.7}$$

$$v_y = v_n + \omega \cdot x \tag{6.8}$$

with

$$v_{wheel} = \sqrt{v_x^2 + v_y^2} \tag{6.9}$$

$$\theta_{wheel} = \text{atan2}(v_y, v_x) \tag{6.10}$$

Next, the equations that were implemented in software are summarized in Table 6.2, which allow the calculation of the individual angle of each leg, as well as the respective speed of each wheel.

Table 6.2: Individual Leg Kinematic Calculation.

Leg 0	**Leg 1**	**Leg 2**	**Leg 3**
$x = a$	$x = a$	$x = -a$	$x = -a$
$y = -b$	$y = b$	$y = b$	$y = -b$
$v_x = v + \omega \cdot b$	$v_x = v - \omega \cdot b$	$v_x = v - \omega \cdot b$	$v_x = v + \omega \cdot b$
$v_y = v_n + \omega \cdot a$	$v_y = v_n + \omega \cdot a$	$v_y = v_n + \omega \cdot a$	$v_y = v_n - \omega \cdot a$

After obtaining $\theta_{0_j\ ref}$ and $\omega_{w_l\ ref}$ for the desired v_r, v_{n_r} and ω_r, joints 0 are controlled with PID position controllers, while the wheels speeds are controlled through PID speed controllers. The diagram in Fig. 6.7 present the mentioned controllers.

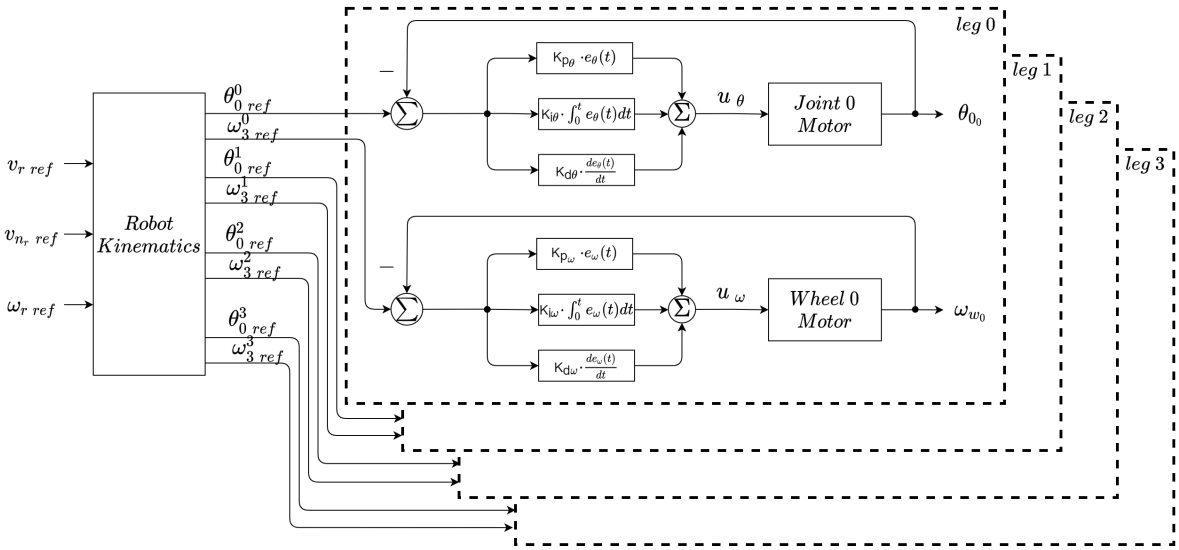


Figure 6.7: Robot's Kinematic Control Block Diagram.

6.4 RobotTune — Robot Software

The firmware developed for the Arduino placed at the joint can communicate in two independent ways — USB and Serial — and, as far as possible, maintain the working compatibility of both modes. This enables the use of the Robot Tune application, presented in Figure 6.8, to control the robot, which communicates through USB with the Teensy board and, at the same time, control each joint individually through Serial Port. This duality allows for, i.e., the tuning of parameters in real time, their recording to flash memory, obtaining data directly from the Arduino, debugging serial communications, among others.

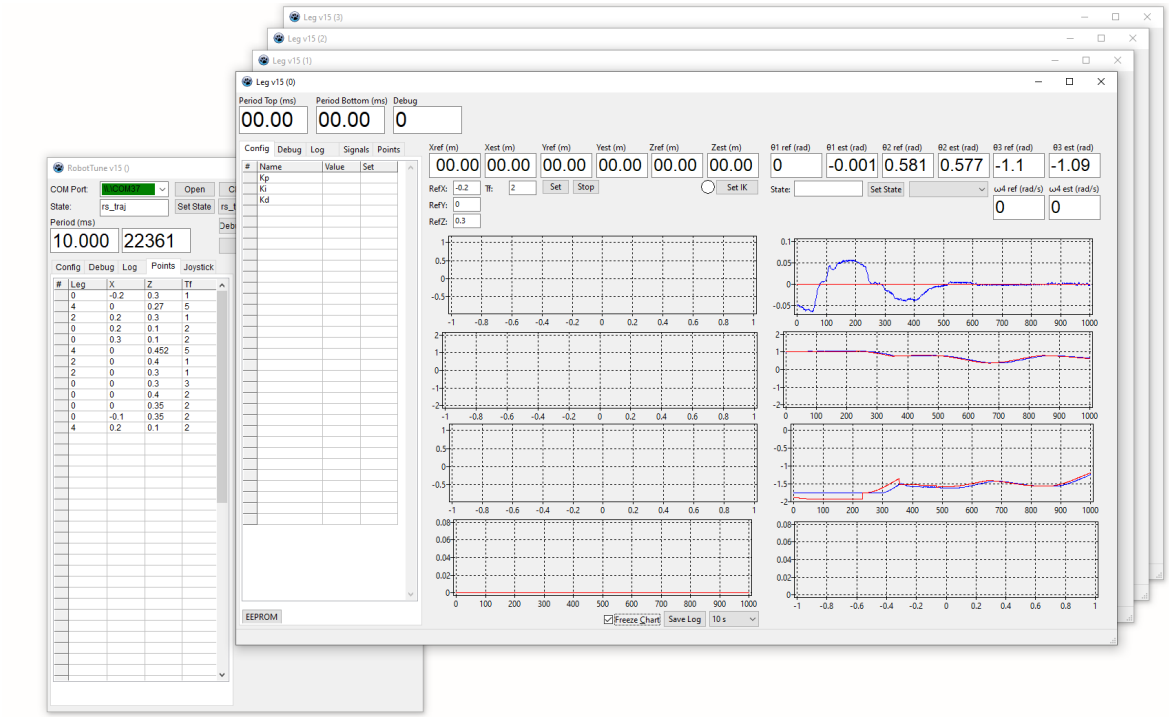


Figure 6.8: Robot Tune.

The communication dynamics between the multiple applications and processing boards is shown in Figure 6.9.

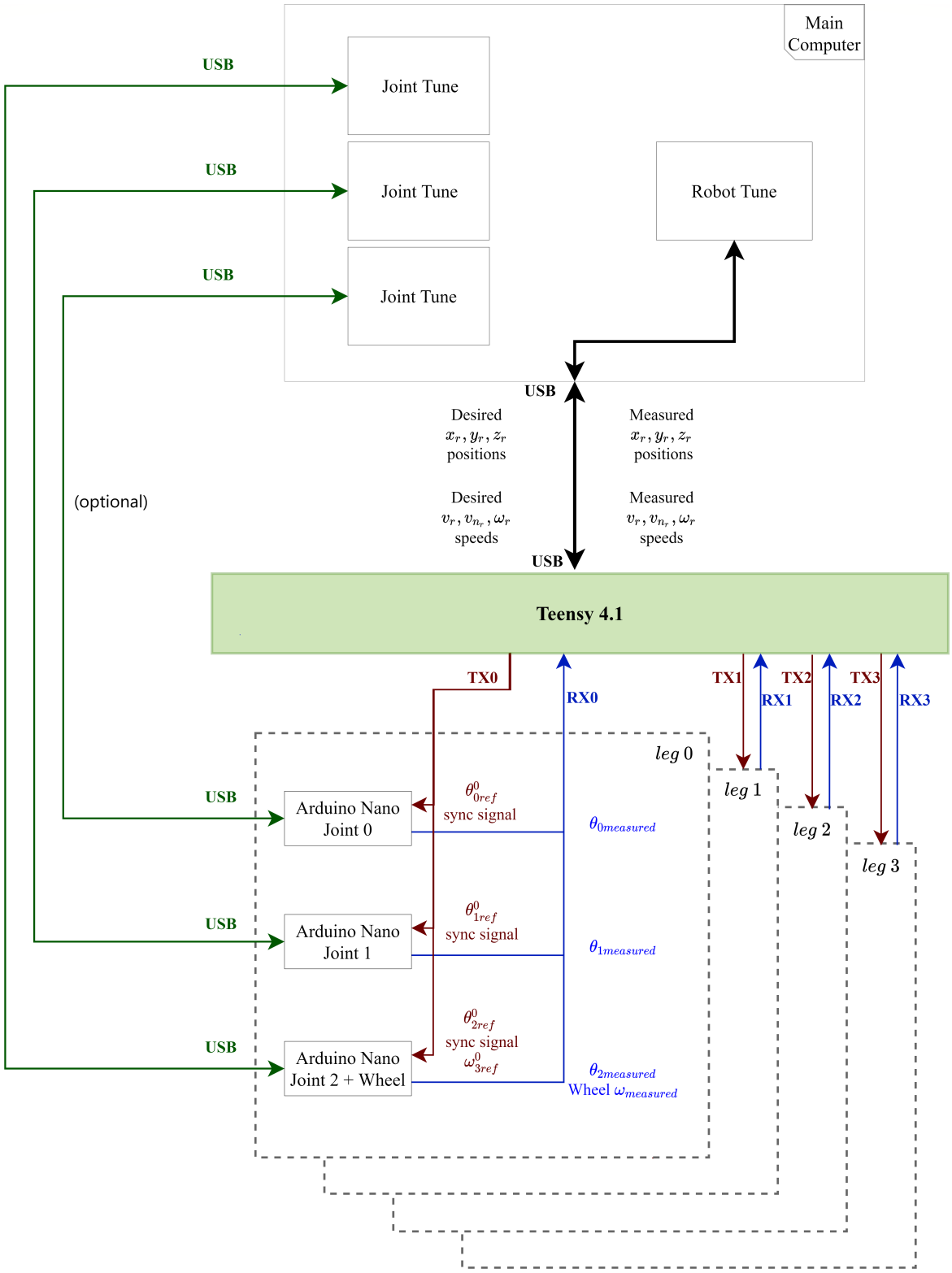


Figure 6.9: Communication Software and Hardware Macro Diagram.

An important feature is the possibility of remote Over The Air (OTA) firmware upload to the Arduino boards. Since these are typically not connected through USB, it would require to connect each and every board to the PC to upload the joint firmware to each Arduino. To avoid this issue, a remote OTA upload feature is embedded in the joint firmware, which enables the remote request to the Apache server, triggered by a message sent by the main microcontroller.

6.5 Realistic Simulation

The research and implementation of this vehicle realistic simulation was a theme of a Master's thesis, by Inês Soares [124].

The simulation model of this legged-wheeled robot, built in the SimTwo simulator, was created by decomposing it into a set of rigid bodies having mass and size equal to the corresponding parts of the real robot.

Leg links and the main body were represented as cuboids, while joints and wheels as cylinders (joints allow connection between different rigid bodies). Except for the joint for rotation of the omni-directional wheel, all the others are actuated. Thus, and as shown in Figure 6.10, the model comprises four legs that connect to the main body on their uppermost joint, and eight wheels.

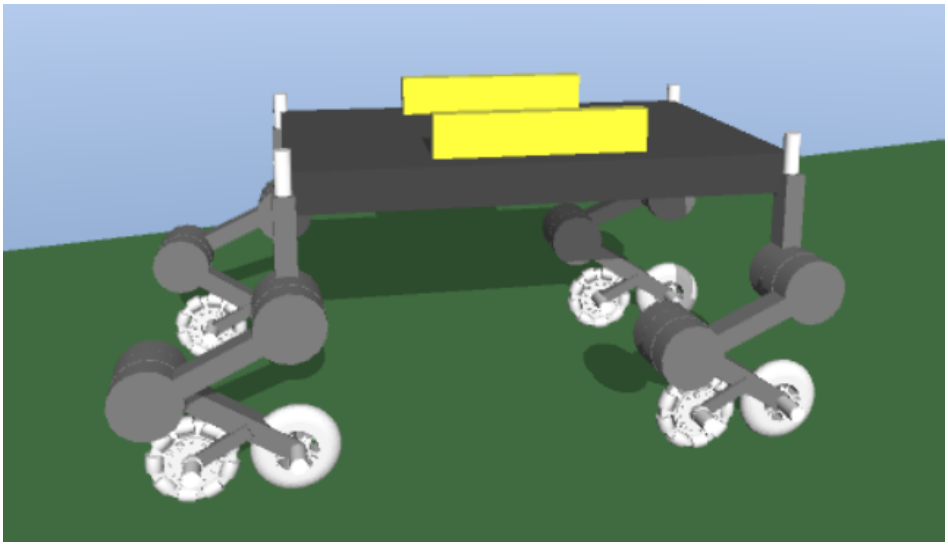


Figure 6.10: Robot simulation model.

The locomotion system of the robot was modeled by combining, per leg, three links with four joints (includes the wheels' joint). The z-axis is the actuation axis of the hip joint, while the y-axis is

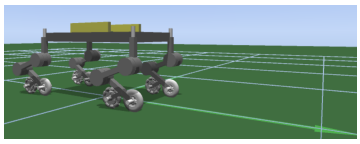
that of the remaining ones. Non-rigidity of the two joints that follow the hip joint were modeled by assembling a compound joint: two separate joints connect the leg links to an intermediate cylindrical link; one of them is actuated and the other has a spring, plus viscous damping and coulomb friction, simulating the elastic component of the joint.

In the real robot, while in purely wheeled mode and performing in low power, its lowest height is kept due to mechanical constraints.

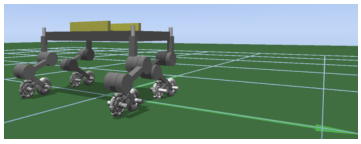
SimTwo has built-in position and speed PID controllers, which were used to control the hip joints and wheels of the model. The state-space controllers were fully implemented within SimTwo using its IDE.

Control of the vehicle in the x and y axes was included in simulations by applying the equations of the robot kinematics described above. Consequently, a first approach to path control was developed.

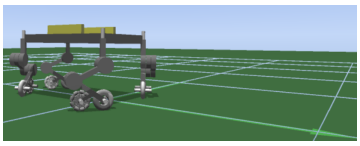
Figure 6.11 shows the obtained positions of the robot legs for some pre-defined speeds. It can be visually determined that, according to the speed references given to the robot, the legs are positioned as expected.



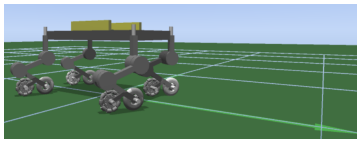
(**a**) v=0.5 v_n=0 w=0



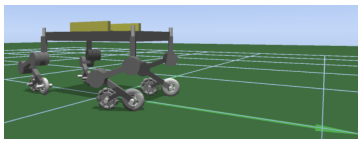
(**b**) v=0 v_n=0.5 w=0



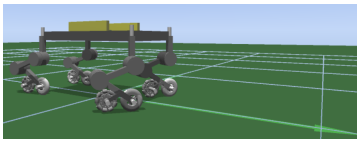
(**c**) v=0 v_n=0 w=0.5



(**d**) v=0.5 v_n=0.5 w=0



(**e**) v=0.5 v_n=0 w=1.0



(**f**) v=0.5 v_n=0.5 w=1.0

Figure 6.11: Control in the x and y axes: setting the speeds v (m/s), v_n (m/s), and w (rad/s) of the robot (all images were taken from the same perspective; the axis in the foreground corresponds to the x-axis).

6.5.1 Path Following

The implemented tracking method requires the robot to follow a line or curve from an initial point to a target point.

Line

A line is defined by a set of two points, corresponding to the initial and target points. The omnidirectional motion ability of this robot enabled the use of controllers to correct its position with simultaneous and independent linear and angular motion. When following a line, the reference is the line itself. The linear speed of the robot is constant, up to a defined distance to the target, where it initiates a smooth, linear deceleration motion to reach and stop at the final position.

A diagram illustrating a typical line tracking problem is represented in Figure 6.12. The robot position must be corrected, so that it converges to the path.

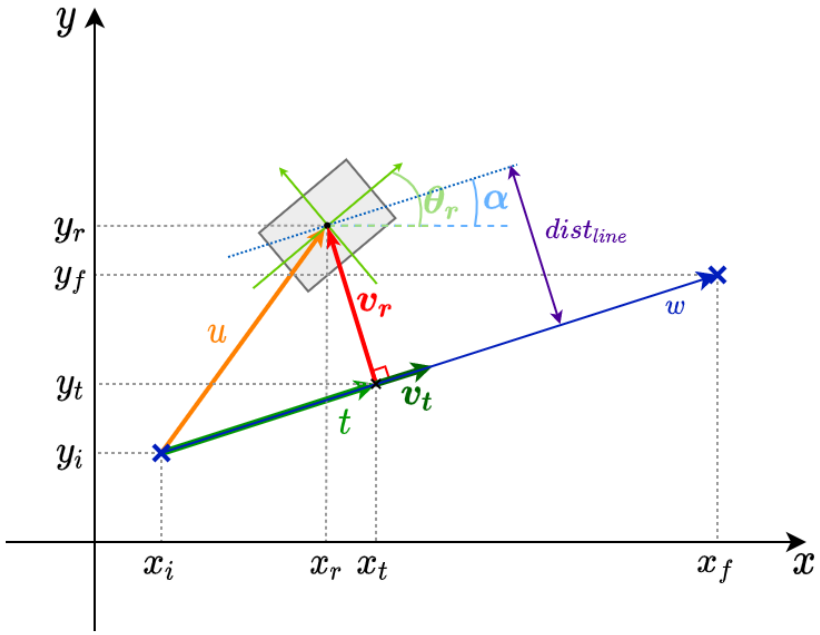


Figure 6.12: Line Tracking Diagram; Robot - Grey Rectangle, Path - Blue Line, Global Frame - Black, Robot Frame - Green.

The implemented line following algorithm is described by Equations (6.29) to (6.43). x_i, x_f, y_i and y_f are, respectively, the defined initial and final points to define the line. Equations (6.11) and (6.12) use these points to calculate the vector $\hat{w}$:

$$w_x = x_f - x_i \tag{6.11}$$

$$w_y = y_f - y_i \tag{6.12}$$

The length of vector $\hat{w}$ is obtained through Equation (6.13)

$$w_d = \sqrt{w_x^2 + w_y^2} \tag{6.13}$$

Vector $\hat{u}$ use the initial points and the robot position $(x_r,\ y_r)$, and is calculated through Equations (6.14) and (6.15).

$$u_x = x_i - x_r \tag{6.14}$$

$$u_y = y_i - y_r \tag{6.15}$$

Vector $\hat{t}$ connects the initial point and the closer point to the line , and can be calculated through:

$$t = -\frac{w_x u_x + w_y u_y}{w_x^2 + w_y^2} \tag{6.16}$$

$$t_x = (1-t)x_i + t x_f \tag{6.17}$$

$$t_y = (1-t)y_i + t y_f \tag{6.18}$$

Vector $\hat{v_r}$ connects the point of the line and the center of the robot. It is calculated through Equations (6.19) and (6.20).

$$v_{rx} = x_r - t_x \tag{6.19}$$

$$v_{ry} = y_r - t_y \tag{6.20}$$

$\hat{v_t}$ is the vector tangent to the line at the intersection point defined in the figure. It is calculated through Equations (6.21) and (6.22).

$$v_{tx} = \frac{w_x}{r_d} \tag{6.21}$$

$$v_{ty} = \frac{w_y}{r_d} \tag{6.22}$$

Equation (6.23) calculates the difference between the α angle, calculated through $\text{atan2}\,(v_{ty}, v_{tx})$, and the robot angle.

$$\theta_e = \theta_r - \text{atan2}\,(v_{ty}, v_{tx}) \tag{6.23}$$

Equations (6.24) to (6.26) describe the path tracking controller.

$$v_x = v_{des} v_{tx} - k_r v_{rx} \tag{6.24}$$

$$v_y = v_{des} v_{ty} - k_r v_{ry} \tag{6.25}$$

$$\omega_r = -k_w(\theta_e - \theta_{des}) \tag{6.26}$$

where k_r and k_w are the controllers gains. Finally, Equations (6.27) and (6.28) calculate the linear speeds of the robot.

$$v_r = \cos(\theta_r) v_x + \sin(\theta_r) v_y \tag{6.27}$$

$$v_{n_r} = -\sin(\theta_r) v_x + \cos(\theta_r) v_y \tag{6.28}$$

The performance of the developed line path tracking technique can be analyzed in the results presented in Figures 6.13 and 6.14. The robot moves through a line defined from point (1,1) to point (2,1) with two different speeds.

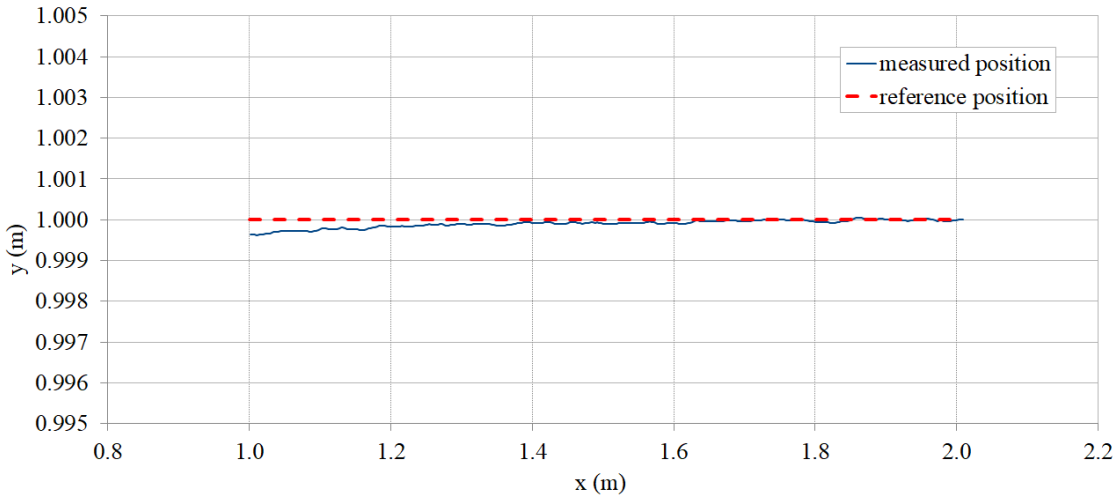


Figure 6.13: Horizontal Line Tracking: 2D Position (m) — Test at 0.3 m/s.

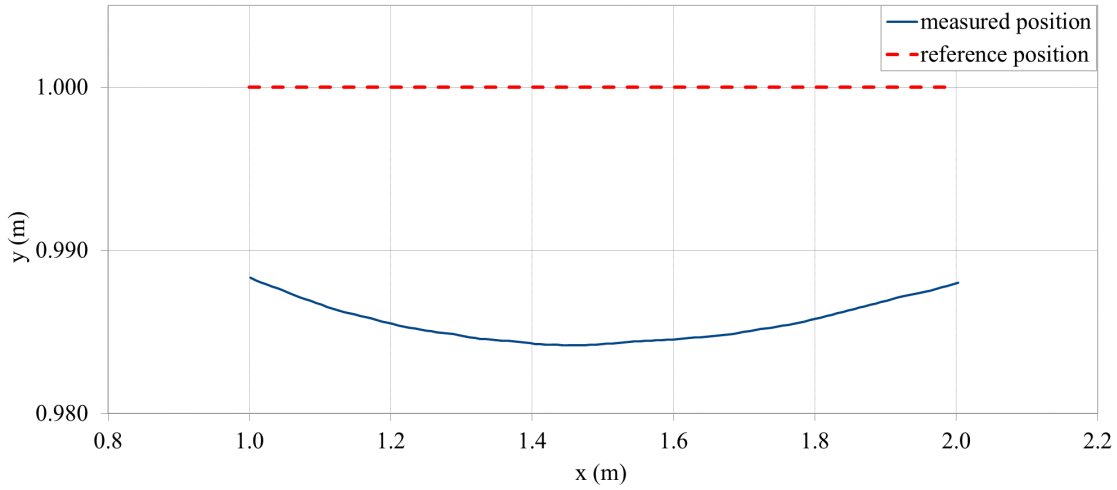


Figure 6.14: Horizontal Line Tracking: 2D Position (m) — Test at 0.8 m/s.

The two following tests were performed using a line that connects (-1,-1) to (1,1), meaning that it is a diagonal line, with an angle of 45^o ($\approx$0.785 rad). Figures 6.15 and 6.16 present the collected data.

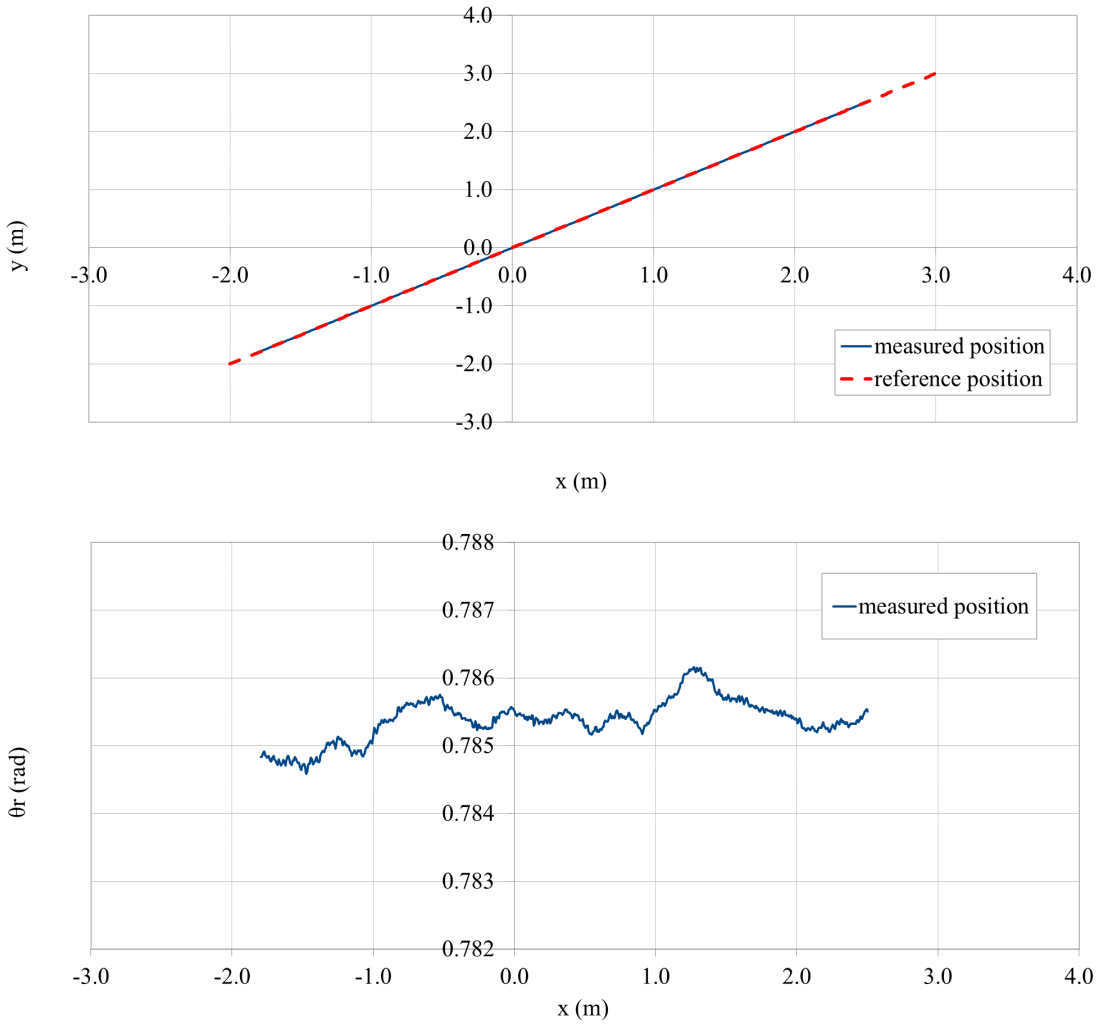


Figure 6.15: Diagonal Line Tracking: 2D Position (m) — Test at 0.3 m/s.

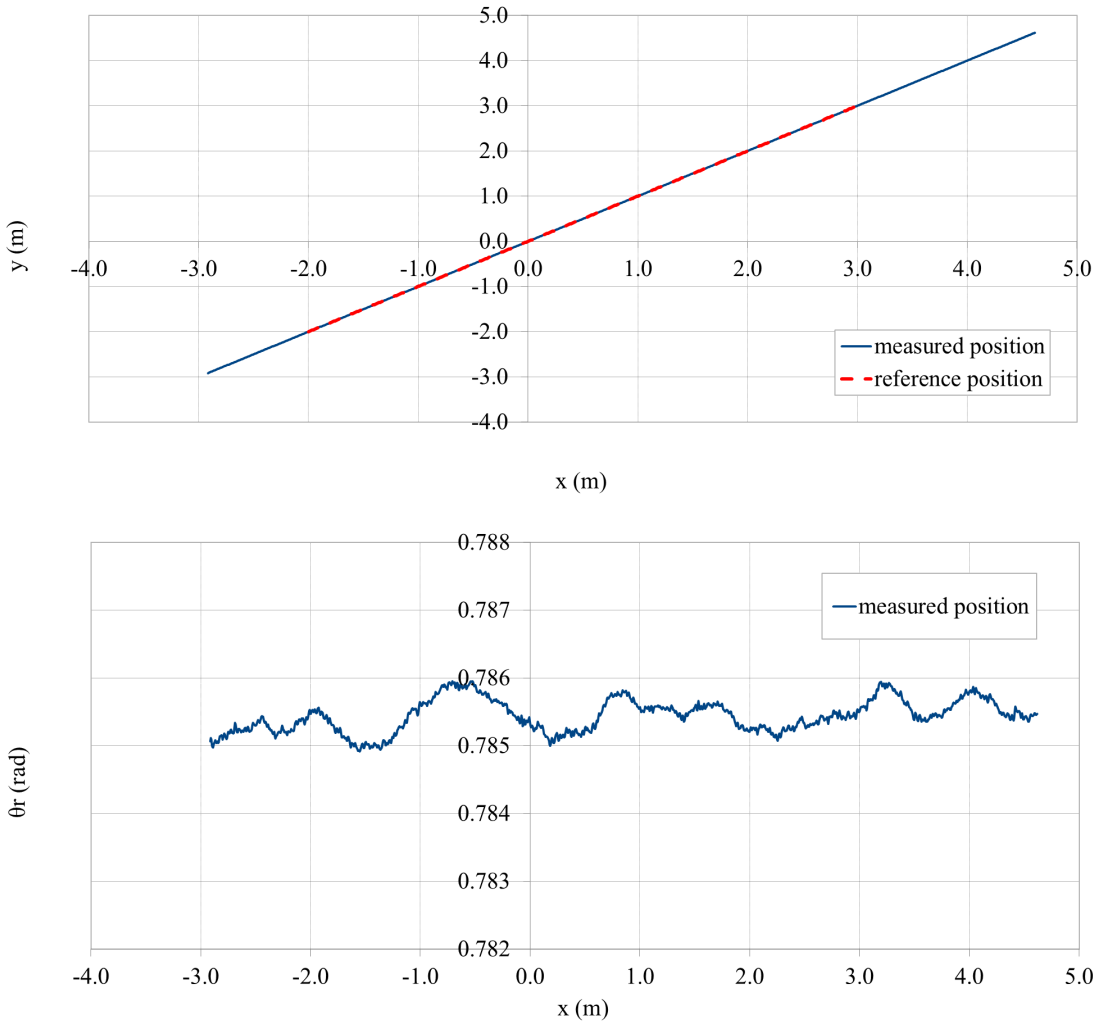


Figure 6.16: Diagonal Line Tracking: 2D Position (m) — Test at 0.8 m/s.

Finally, it was performed a test using a horizontal line that connects point (1,1) to point (2,1), but requiring the robot to move with a yaw of 25^o ($\approx$0.436 rad). Figure 6.17 presents the collected data.

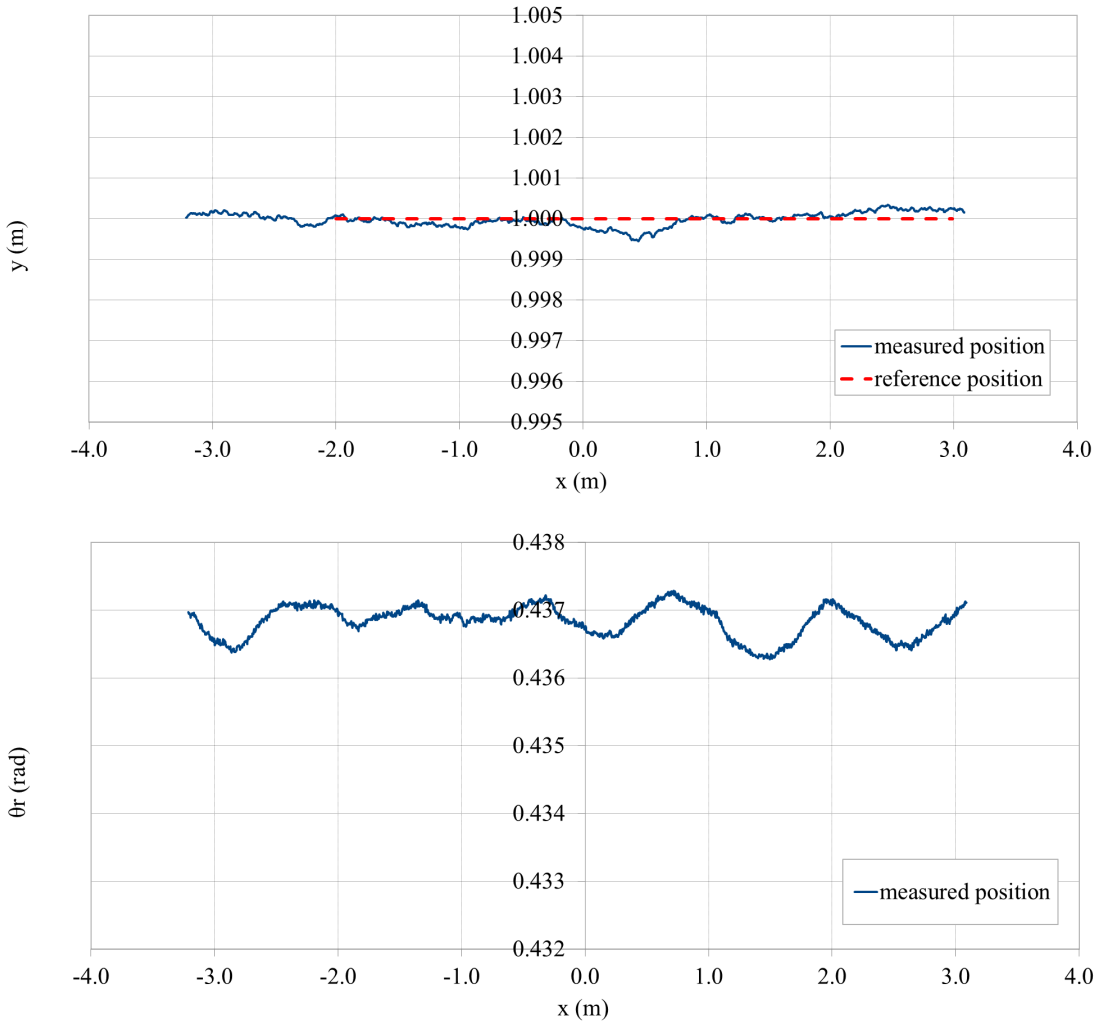


Figure 6.17: Horizontal Line Tracking with $\theta_r = 25^o$: 2D Position (m) — Test at 0.3 m/s.

The maximum absolute error obtained in θ_r for this last test was 0.0006 rad, while the average was 0.0005 rad. The remaining obtained errors are gathered in Table 6.3.

Table 6.3: Line Path Following Errors.

Test Speed	**Absolute Error**	x_r **(m)**	y_r **(m)**
Horizontal 0.3 m/s	Average (m)	n/a	0.00001
	Maximum (m)	n/a	0.0004
Horizontal 0.8 m/s	Average (m)	n/a	0.0148
	Maximum (m)	n/a	0.0158
Diagonal 0.3 m/s	Average (m)	0.0044	0.0044
	Maximum (m)	0.0103	0.0102
Diagonal 0.8 m/s	Average (m)	0.0038	0.0047
	Maximum (m)	0.0067	0.0078

Next, the robot was placed away from the desired line and the path following algorithm was turned on. The results of this test is presented in Figure 6.18.

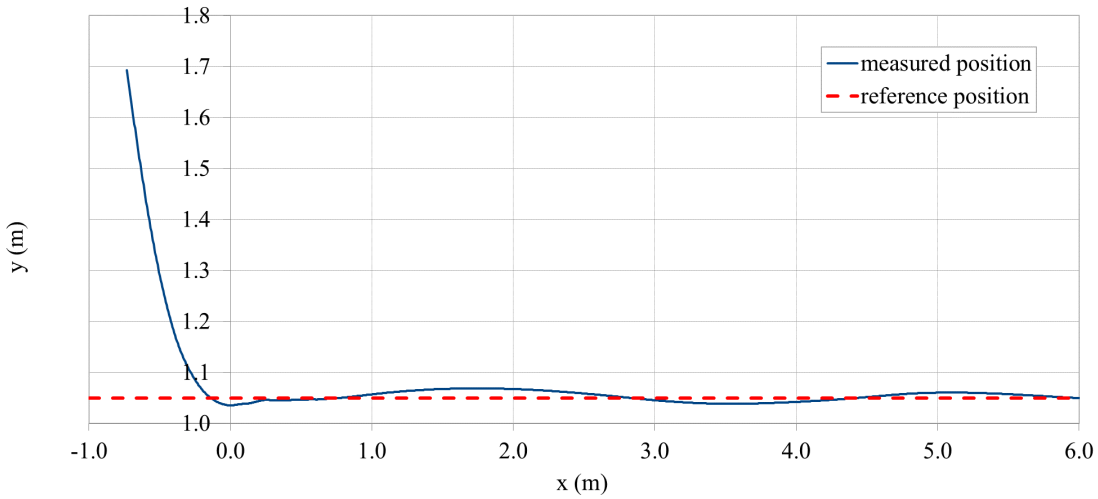


Figure 6.18: Line Tracking: 2D Position (m) with Outside Approach.

Curve

A circumference requires a set of three points to be defined. When following a cimcumference, the reference corresponds to the tangent to the cimcumference in the nearest point between the curve and the robot's position. Figure 6.19 represents the typical problem the curve tracking algorithm must solve.

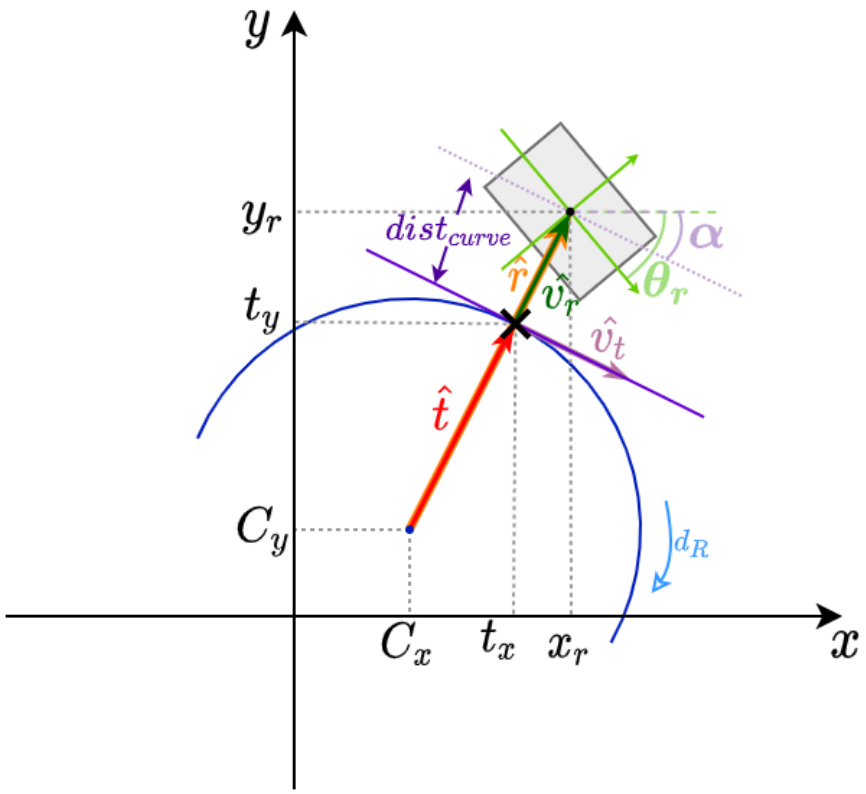


Figure 6.19: Curve Tracking Diagram; Robot - Grey Rectangle, Path - Blue Curve, Global Frame - Black, Robot Frame - Green.

The implemented curve following algorithm is described by Equations (6.29) to (6.43). C_x, C_y and C_r are, respectively, the defined central point and radius of the desired curve. Equations (6.29) and (6.30) use the central point, the radius and the robot position (x_r, y_r) to calculate the vector $\hat{r}$:

$$r_x = x_r - C_x \tag{6.29}$$

$$r_y = y_r - C_y \tag{6.30}$$

The length of vector $\hat{r}$ is obtained through Equation (6.31)

$$r_d = \sqrt{r_x^2 + r_y^2} \tag{6.31}$$

Vector $\hat{t}$ is the one illustrated in red in the figure, and is calculated through Equations (6.32) and (6.33).

$$t_x = \frac{r_x}{r_d} C_r \tag{6.32}$$

$$t_y = \frac{r_y}{r_d} C_r \tag{6.33}$$

Vector $\hat{v_r}$ is the one illustrated in green in the figure, connecting the point of the curve and the center of the robot. It is calculated through Equations (6.34) and (6.35).

$$v_{rx} = r_x - t_x \tag{6.34}$$

$$v_{ry} = r_y - t_y \tag{6.35}$$

$\hat{v_t}$ is the vector tangent to the curve at the intersection point defined in the figure. It is calculated through Equations (6.36) and (6.37).

$$v_{tx} = \frac{r_y}{r_d} \tag{6.36}$$

$$v_{ty} = -\frac{r_x}{r_d} \tag{6.37}$$

Equation (6.38) calculates the difference between the α angle, calculated through $\text{atan2}(v_{ty}, v_{tx})$, and the robot angle.

$$\theta_e = \theta_r - \text{atan2}(v_{ty}, v_{tx}) \tag{6.38}$$

Equations (6.39) to (6.41) describe the path tracking controller.

$$v_x = v_{des} v_{tx} - k_r v_{rx} \tag{6.39}$$

$$v_y = v_{des} v_{ty} - k_r v_{ry} \tag{6.40}$$

$$\omega_r = -\frac{v_{des}}{C_r} - k_w(\theta_e - \theta_{des}) \tag{6.41}$$

where k_r and k_w are the controllers gains. Finally, Equations (6.42) and (6.43) calculate the linear speeds of the robot.

$$v_r = \cos(\theta_r) v_x + \sin(\theta_r) v_y \tag{6.42}$$

$$v_{n_r} = -\sin(\theta_r) v_x + \cos(\theta_r) v_y \tag{6.43}$$

Two curve path following tests were performed, fixing a central point $(-1,0)$ and a radius of 1 meter, with desired speeds of 0.3 and 0.8 m/s. The performance of the implemented path tracking technique can be analyzed through Figures 6.20 and 6.21. The errors of each path are presented in Table 6.4.

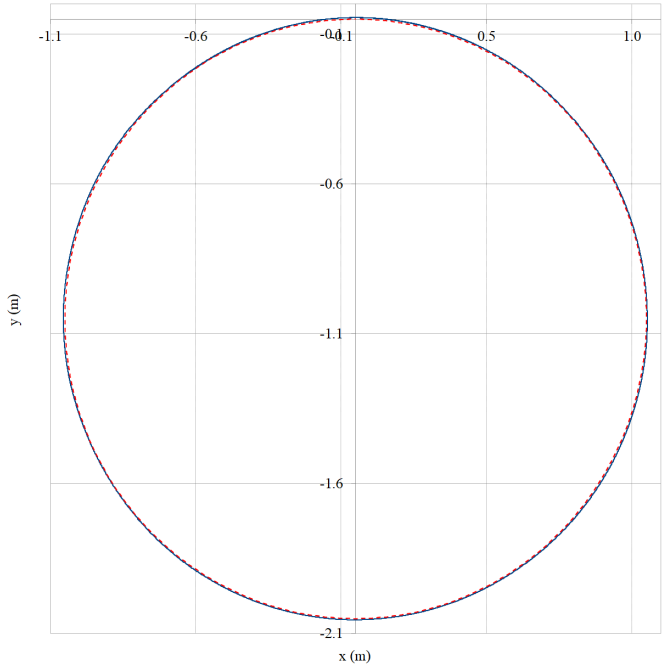


Figure 6.20: Curve Tracking: 2D Position (m) — Test at 0.3 m/s.

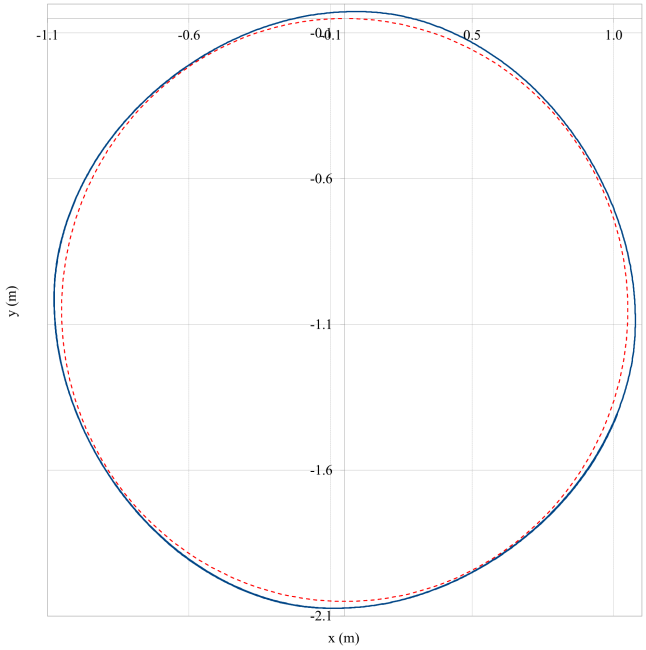


Figure 6.21: Curve Tracking: 2D Position (m) — Test at 0.8 m/s.

Table 6.4: Curve Path Following Errors.

Test Speed	**Absolute Error**	x **(m)**	y **(m)**
0.3 m/s	Average (m)	-0.0010	-0.0005
	Maximum (m)	0.0052	0.0072
0.8 m/s	Average (m)	0.0015	0.0016
	Maximum (m)	0.0305	0.0319

These tests demonstrate that the robot is able to continuously follow the desired path. The errors increased when the robot speed was increased, being in the order of millimeters in the test with a speed of 0.3 m/s, and in the order of centimeters in the test at 0.8 m/s.

Next, the results of three approach tests are presented. The robot was placed on a region inside/outside the desired circumference and the path following algorithm was turned on. The results of these tests are presented in Figures 6.22 and 6.23, where the robot was placed inside and outside the cirmcumference, respectively.

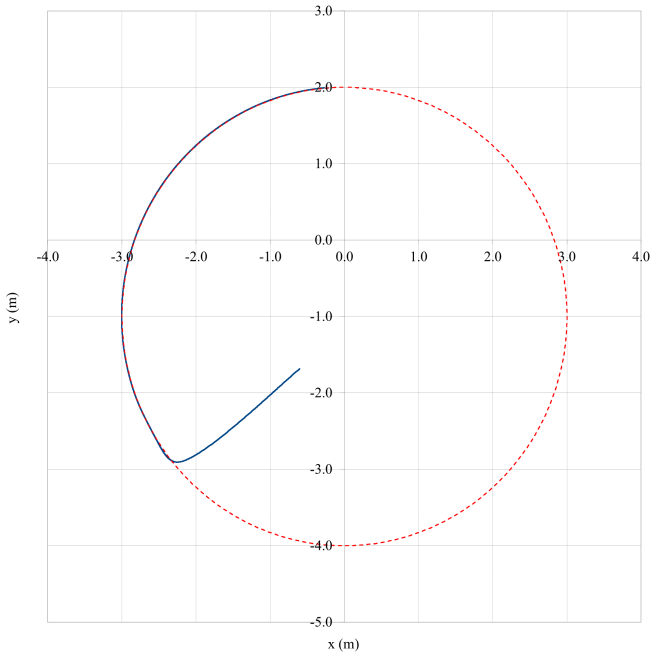


Figure 6.22: Curve Tracking: 2D Position (m) — Test with Interior Approach.

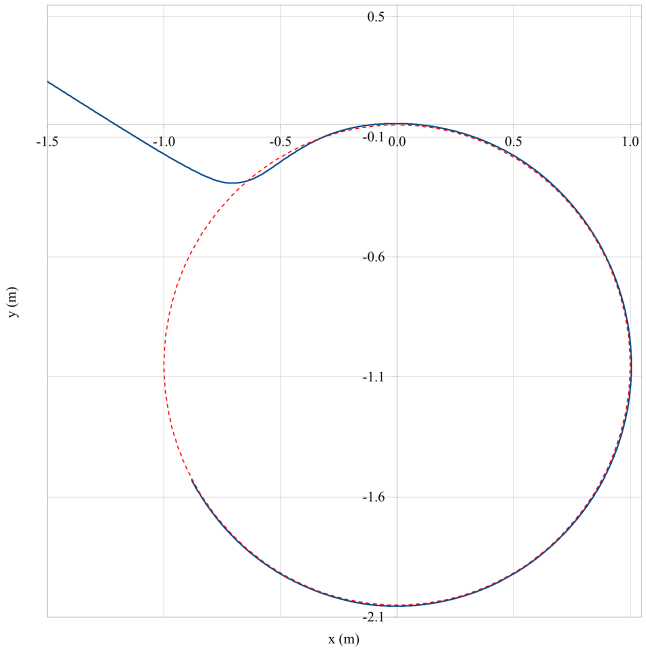


Figure 6.23: Curve Tracking: 2D Position (m) — Test with Exterior Approach.

In Figure 6.24, the robot was following the circumference with a bigger radius. Then, the desired radius was exchanged twice, for a smaller one, and again to the biggest one.

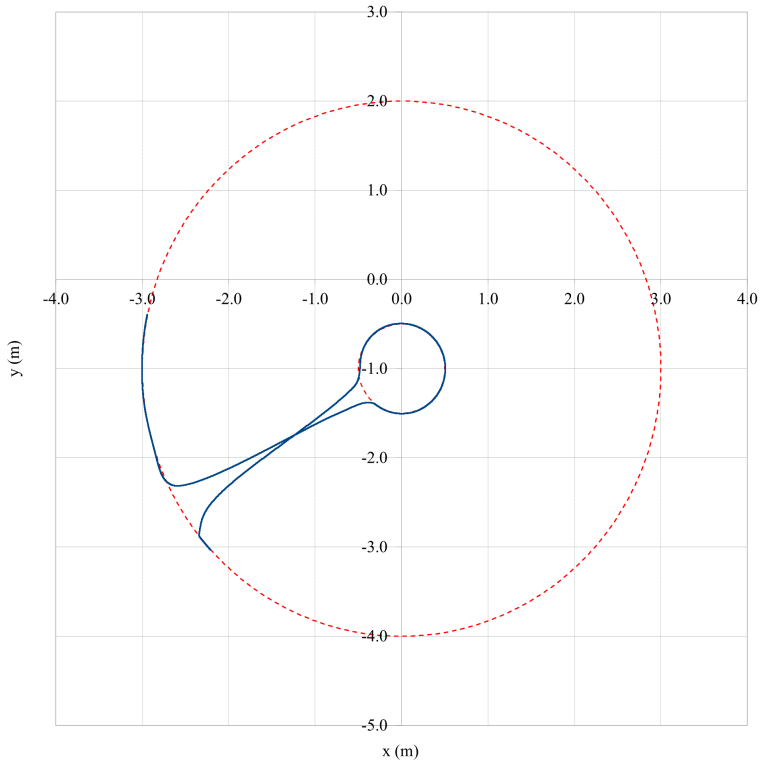


Figure 6.24: Curve Tracking: 2D Position (m) — Test between Interior and Exterior Approaches.

These tests demonstrate that the robot is able to converge to the path from several points in the space, and maintain it.

6.6 Conclusions

Throughout the presented chapter, the leg introduced in chapter 5 was used to assemble the quadruped. The robot design, its software interface, and the proposed algorithms for path following control were detailed. The robot realistic simulation was also introduced. Several tests were performed to the implemented path following algorithms. These showed that the errors in linear paths are at most 1 cm. For circle paths, the maximum error is 3 cm.

The presented research on this topic crosses several publications that led to the design, model and control of the obtained hybrid platform. Despite this, the publication that specifically presents the

research carried out within the scope of the autonomous hybrid vehicle is the journal article:

Pinto, V.H.; Soares, I.N.; Rocha, M.; Lima, J.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of an Autonomous Legged-Wheeled Hybrid Robotic Vehicle with Non-Rigid Joints. Appl. Sci. 2021, 11, 6116. https://doi.org/10.3390/app11136116

CHAPTER 7

Conclusion and Future Work

"It's not that I'm so smart, it's just that I stay with problems longer."

— Albert Einstein

This chapter will present an overview of the work presented in the previous chapters. The main contributions will also be presented, as well as suggestions for topics for future research.

7.1 Conclusion

The design of a hybrid legged-wheeled locomotion system can combine the advantages of both legged and wheeled systems, while trying to avoid their disadvantages. However, their design, construction and control are more complex. Also, these systems tend to be more expensive, as well as exhibiting increased power consumption.

The conception and design of a new platform using a legged-wheeled locomotion system was performed. This is modular, cost-effective and reconfigurable, and can be used for multiple purposes. A quadruped was designed using a novel leg design, based on a non-rigid joint approach.

Non-rigid joints increase the system tolerance to impacts, thus improving the system ability to work in highly dynamic environments. Thus, this solution was applied to design the legs of the proposed hybrid legged-wheeled locomotion system, with potential applications in legged, humanoid and manipulator robots. The proposed leg's design, hardware structure, developed software and communication strategy were presented. The equations for the direct and inverse kinematics calculation were described. Moreover, the control methods for the leg positioning were detailed. Some leg positioning tests were performed, and the results obtained were presented. Tests on x^l, z^l positioning obtained maximum steady-state errors of about 0.3 cm were obtained, while the errors in angular position of the joints θ_1, θ_2 remained below 0.02 rad.

However, its design is more complex and the positioning precision is inferior. To overcome these, more accurate models and improved controllers are required. The joints models with and without non-linear elements were deduced and presented. Linear (or linearized) models were used for the design and implementation of the studied controllers, while non-linear models were used in the realistic simulation. The parameters for the motor, rigid and non-rigid joints were estimated. Furthermore, those models were compared with physical experiments, obtaining successful results. The performed tests were detailed and the obtained data were presented. All the core components used to build the system were described. The off-the-shelf elements that were acquired were briefly presented, while the elements that were custom designed for final solution were detailed.

The robot design, its software interface, and the proposed algorithms for path following control were detailed. The robot realistic simulation was also introduced. Several tests were performed to the proposed path following algorithms. These showed that the steady-state error in linear paths is at most 1 cm. For circular paths, the maximum steady-state error is 3 cm.

7.2 Scientific Contributions

The scientific contributions of this thesis are:

- The design of a joint with a passive approach to its non-rigid elements, enhancing the system's performance and increasing safety and robustness;
- The model conception and estimation, containing non-linear elements applied to the proposed non-rigid joints;
- The proposal of several control modes for the system, capable of driving the proposed joint with accuracy and compliance;
- The conception and design of a new platform using a legged-wheeled locomotion system, which is modular, cost-effective and reconfigurable, and can be used for multiple purposes;
- The proposed platform presents several different locomotion modes, capable of optimizing energy consumption, speed and locomotion through irregular environments;

These contributions resulted in several relevant publications, that are mentioned below.

Conference Articles

The proposed models and control approaches for the DC motor were published in the articles that follow

- Pinto, Vítor H., José Gonçalves, and Paulo Costa. ”Model of a DC motor with worm gearbox.” Portuguese Conference on Automatic Control. Springer, Cham, 2020.

- Pinto, Vítor H., José A. Gonçalves, and Paulo Costa. ”Introduction to DC motors for engineering students based on laboratory experiments.” 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT). IEEE, 2019.

The design of the non-rigid joint, the linear and non-linear model deduction and the proposed control approaches were published in the following conference article:

- Pinto, Vítor H., et al. ”Modeling of an Elastic Joint: An Experimental Setup Approach.” Portuguese Conference on Automatic Control. Springer, Cham, 2020.

Journal Articles

The design of the non-rigid joint, the model conception and estimation, containing non-linear elements and the proposed control approaches also originated a dedicated journal publication:

- Pinto, V.H.; Gonçalves, J.; Costa, P. Towards a More Robust Non-Rigid Robotic Joint. Appl. Syst. Innov. 2020, 3, 45. https://doi.org/10.3390/asi3040045

The proposed models and control approaches for the DC motor coupled to a non-rigid were also published in the journal article that follows

- Pinto, V.H.; Gonçalves, J.; Costa, P. Modeling and Control of a DC Motor Coupled to a Non-Rigid Joint. Appl. Syst. Innov. 2020, 3, 24. https://doi.org/10.3390/asi3020024

The conception and design of a new platform using a legged-wheeled system, capable of using several different locomotion modes, optimizing energy consumption, speed and locomotion through irregular environments resulted in the following journal articles:

- Pinto, V.H.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of a Single Leg for a Legged-Wheeled Locomotion System with Non-Rigid Joint. Actuators 2021, 10, 29 https://doi.org/10.3390/act10020029.
- Pinto, V.H.; Soares, I.N.; Rocha, M.; Lima, J.; Gonçalves, J.; Costa, P. Design, Modeling, and Control of an Autonomous Legged-Wheeled Hybrid Robotic Vehicle with Non-Rigid Joints. Appl. Sci. 2021, 11, 6116. https://doi.org/10.3390/app11136116

7.3 Future Work

Throughout this thesis, several research challenges were addressed. However, there are also research directions and challenges that can be further explored.

In the future, although the presented solutions obtained satisfactory results, there is always room for design improvements of the non-rigid joint. It will be interesting to test the proposed algorithm for following linear and circular paths in the physical robot. Moreover, it will be appropriate to propose algorithms for trajectory following. To operationalize vehicle navigation, it is also important to incorporate lasers in the hips of the frontal legs, allowing area mapping and obstacle detecting.

The tuning and testing of the remaining modes of locomotion, allowed by the proposed platform, should also be explored. Also, the proposal of climbing algorithms will also be interesting, to take advantage of the proposed platform configuration, as well as optimizing the gait.

Appendix A

Throughout this appendix, the tables regarding the message flow between the Teensy and each Arduino Nano located in the Legs is detailed.

Table 7.1: Communication Channels Transmission - Robot Tune

Channel	**Subchannel**	**Function**
G ... J	i ... x	send parameter data - frames 1 to 4
R	v	robot V speed reference
	n	robot V_n speed reference
	w	robot ω speed reference
S	m	robot state
	n	activate leg 0 controllers
	o	activate leg 1 controllers
	p	activate leg 2 controllers
	q	activate leg 3 controllers
	r	activate all legs controllers
T	x	leg x position reference
	z	leg z position reference
	t	leg trajectory final time reference
U	g	matrix configuration files transmission
	j	matrix configuration files index
	m	reload matrix files command

Table 7.2: Communication Channels Reception - Robot Tune

Channel	**Subchannel**	**Function**
G ... J	i ... x	receive parameter data - frames 1 to 4
S	g	debug data
	k	robot state
	r	absolute angular position - joint 2
	s	absolute angular position - joint 1
	t	absolute angular position + wheel speed - joint 0
R	r	angular position reference - joint 2
	s	angular position reference - joint 1
	t	angular position + wheel speed references - joint 0
P	0	control loop cycle time (control debug)

Table 7.3: Communication Channels Transmission - Joint Tune

Channel	**Subchannel**	**Function**
G ... J	i ... x	send joint parameter data - frames 1 to 4
R	p	set reference value - motor 0
	q	set reference value - motor 1
S	k	begin sweep signal with defined function
	s	set state - motor 0
	t	set state - motor 1
	y	load joint parameters data from flash memory
	z	save joint parameters data to flash memory

Table 7.4: Communication Channels Reception - Joint Tune

Channel	**Subchannel**	**Function**
G ... J	i ... x	receive joint parameter data - frames 1 to 4
L	g	active controller
	i	current consumption
	o	absolute angular position - motor 0
	p	absolute angular position - motor 1
	s	load cell measurement
P	i	data sample number (communication debug)
	o	control loop cycle time (control debug)
R	p	angular position reference - motor 0
	q	angular position reference - motor 1
	s	angular speed reference - motor 0
	t	angular speed reference - motor 1
	u	load cell reference value
S	s	active PWM value - motor 0
	t	active PWM value - motor 1
W	j	absolute angular speed - motor 0
	k	absolute angular speed - motor 1
	o	relative angular speed - motor 0
	p	relative angular speed - motor 1
	s	relative angular position - motor 0
	t	relative angular position - motor 1
Z	z	general debug information parameter

Table 7.5: Communication Channels Transmission - Robot Firmware USB Port

Channel	**Subchannel**	**Function**
G ... J	i ... x	send parameter data - frames 1 to 4
R	r	absolute angular position reference - joint 2
	s	absolute angular position reference - joint 1
	t	absolute angular position + wheel speed references - joint 0
S	k	robot state + control loop cycle time (control debug)
P	o	synchronism delta time

Table 7.6: Communication Channels Reception - Robot Firmware USB Port

Channel	Subchannel	Function
G ... J	i ... x	send parameter data - frames 1 to 4
P	g	debug data
R	s	matrix configuration files transmission
	t	matrix configuration files index
	v	robot V speed reference
	n	robot V_n speed reference
	w	robot ω speed reference
S	m	robot state
	n	activate leg 0 controllers
	o	activate leg 1 controllers
	p	activate leg 2 controllers
	q	activate leg 3 controllers
	r	leg activation signal
T	g	leg x position reference
	k	leg z position reference
	o	leg trajectory final time reference
U	all	all subchannels sent directly to the legs

Table 7.7: Communication Channels Transmission - Robot Firmware Serial Port

Channel	Subchannel	Function
P	g	desired joint states
S	r	absolute angular position reference - joint 2
	s	absolute angular position reference - joint 1
	t	absolute angular position + wheel speed references - joint 0
	y	desired robot state

Table 7.8: Communication Channels Reception - Robot Firmware Serial Port

Channel	Subchannel	Function
S	r	absolute angular position - joint 2
	s	absolute angular position - joint 1
	t	absolute angular position + wheel speed - joint 0

Table 7.9: Communication Channels Transmission - Joint Firmware USB Port

Channel	Subchannel	Function
G ... J	i ... x	send parameter data - frames 1 to 4
R	r	absolute angular position reference - joint 2
	s	absolute angular position reference - joint 1
	t	absolute angular position + wheel speed references - joint 0
S	k	robot state + control loop cycle time (control debug)
P	o	synchronism delta time

Table 7.10: Communication Channels Reception - Joint Firmware USB Port

Channel	Subchannel	Function
G ... J	i ... x	receive parameter data - frames 1 to 4
S	g	save joint ID
	k	put PWM value buffer - motor 0
	l	put PWM value buffer - motor 1
	o	set PWM value - motor 0
	p	set PWM value - motor 1
	s	set joint state
	y	load parameters command
	z	save parameters command
T	p	angular position reference - joint
	q	angular speed reference - wheel
	t	leg x position reference
	x	leg z position reference
	z	leg trajectory final time reference
R	o	activate controller - motor 0 and 1
	p	absolute angular position reference - joint 1
	q	absolute angular position reference - joint 2
	r	relative angular position reference - joint 2
	s	relative angular position reference - joint 2
U	i	debug data
	r	reload matrix files command

Table 7.11: Communication Channels Transmission - Joint Firmware Serial Port

Channel	Subchannel	Function
S	r	absolute angular position - joint 2
	s	absolute angular position - joint 1
	t	absolute angular position + wheel speed - joint 0

Table 7.12: Communication Channels Reception - Joint Firmware Serial Port

Channel	Subchannel	Function
G ... J	i ... x	receive parameter data - frames 1 to 4
S	o	activate joint controllers - motor 0 and 1
	r	absolute angular position - joint 2
	s	absolute angular position - joint 1
	t	absolute angular position + wheel speed - joint 0
	y	sync factor + robot state + controllers activation command
U	g ... l	matrix values reception
	m	debug data
	r	reload matrix values
P	g	desired joint state

References

[1] L. Jaiem, S. Druon, L. Lapierre, and D. Crestani, "A step toward mobile robots autonomy: Energy estimation models," in *Towards Autonomous Robotic Systems*, L. Alboul, D. Damian, and J. M. Aitken, Eds. Cham: Springer International Publishing, 2016, pp. 177–188.

[2] G. Grandesso, G. Bravo-Palacios, P. Wensing, M. Fontana, and A. Del Prete, "Exploring the limits of a hybrid actuation system through co-design," *IEEE Robotics and Automation Letters*, 2020.

[3] Y. Ding and H.-W. Park, "Design and experimental implementation of a quasi-direct-drive leg for optimized jumping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 300–305.

[4] S. Yu, T.-H. Huang, X. Yang, C. Jiao, J. Yang, Y. Chen, J. Yi, and H. Su, "Quasi-direct drive actuation for a lightweight hip exoskeleton with high backdrivability and high bandwidth," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 4, pp. 1794–1802, 2020.

[5] J. Kim, T. Kang, D. Song, and S.-J. Yi, "Design and Control of a Open-Source, Low Cost, 3D Printed Dynamic Quadruped Robot," *Applied Sciences*, vol. 11, no. 9, p. 3762, 2021.

[6] C. Lee, S. Kwak, J. Kwak, and S. Oh, "Generalization of series elastic actuator configurations and dynamic behavior comparison," in *Actuators*, vol. 6, no. 3. Multidisciplinary Digital Publishing Institute, 2017, p. 26.

[7] W. Huo, M. A. Alouane, Y. Amirat, and S. Mohammed, "Force control of SEA-based exoskeletons for multimode human–robot interactions," *IEEE Transactions on Robotics*, vol. 36, no. 2, pp. 570–577, 2019.

[8] Q. Zhang, D. Sun, W. Qian, X. Xiao, and Z. Guo, "Modeling and control of a cable-driven rotary series elastic actuator for an upper limb rehabilitation robot," *Frontiers in neurorobotics*, vol. 14, p. 13, 2020.

[9] M. F. Silva and J. T. Machado, "A literature review on the optimization of legged robots," *Journal of Vibration and Control*, vol. 18, no. 12, pp. 1753–1767, 2012. [Online]. Available: https://doi.org/10.1177/1077546311403180

[10] V. H. Pinto, J. Gonçalves, and P. Costa, "Model of a DC motor with worm gearbox," in *Portuguese Conference on Automatic Control*. Springer, 2020, pp. 638–647.

[11] V. H. Pinto, J. A. Gonçalves, and P. Costa, "Introduction to dc motors for engineering students based on laboratory experiments," in *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2019, pp. 1522–1527.

[12] V. H. Pinto, J. Lima, J. Gonçalves, and P. Costa, "Modeling of an Elastic Joint: An Experimental Setup Approach," in *Portuguese Conference on Automatic Control*. Springer, 2020, pp. 676–685.

[13] V. H. Pinto, J. Gonçalves, and P. Costa, "Modeling and control of a dc motor coupled to a non-rigid joint," *Applied System Innovation*, vol. 3, no. 2, p. 24, 2020.

[14] V. H. Pinto, J. Gonçalves, and P. Costa, "Towards a more robust non-rigid robotic joint," *Applied System Innovation*, vol. 3, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2571-5577/3/4/45

[15] V. H. Pinto, J. Gonçalves, and P. Costa, "Design, modeling, and control of a single leg for a legged-wheeled locomotion system with non-rigid joint," in *Actuators*, vol. 10, no. 2. Multidisciplinary Digital Publishing Institute, 2021, p. 29.

[16] V. H. Pinto, I. N. Soares, M. Rocha, J. Lima, J. Gonçalves, and P. Costa, "Design, modeling, and control of an autonomous legged-wheeled hybrid robotic vehicle with non-rigid joints," *Applied Sciences*, vol. 11, no. 13, p. 6116, 2021.

[17] S. Y. Nof, *Handbook of industrial robotics*. John Wiley & Sons, 1999.

[18] H. Widditsch, "Spurv-the first decade," WASHINGTON UNIV SEATTLE APPLIED PHYSICS LAB, Tech. Rep., 1973.

[19] N. J. Nilsson, *Triangle tables: A proposal for a robot programming language*. SRI International. Artificial Intelligence Center, 1985, vol. 60.

[20] J. Tenreiro Machado and M. Silva, "An overview of legged robots," in *International symposium on mathematical methods in engineering*, 2006, pp. 1–40.

[21] P. Flores, J. C. P. Claro, and A. Ribeiro, "Design of a hexapod robotic system," 2001.

[22] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "Bigdog, the rough-terrain quadruped robot," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 10 822–10 825, 2008.

[23] M. Hutter, C. D. Remy, M. A. Hoepflinger, and R. Siegwart, "Scarleth: Design and control of a planar running robot," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 562–567.

[24] Y. Shi, P. Wang, F. Zha, L. Sun, M. Li, W. Jia, and Z. Li, "Mechanical design and force control algorithm for a robot leg with hydraulic series-elastic actuators," in *International Journal of Advanced Robotic Systems*, vol. 17, no. 3. SAGE Publications Sage UK: London, England, 2020.

[25] M. Koshy, A. R. Nair, and A. Krishnan, "Design, manufacturing and testing of a hybrid locomotion mobile robot "HyLMoR"," *Materials Today: Proceedings*, vol. 24, pp. 1292–1301, 2020.

[26] Q. Shi, S. Miyagishima, S. Konno, S. Fumino, H. Ishii, A. Takanishii, C. Laschi, B. Mazzolai, V. Mattoli, and P. Dario, "Development of the hybrid wheel-legged mobile robot wr-3 designed to interact with rats," in *2010 3rd IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*. IEEE, 2010, pp. 887–892.

[27] G. Freitas, F. Lizarralde, L. Hsu, V. Paranhos, N. R. S. Dos Reis, and M. Bergerman, "Design, modeling, and control of a wheel-legged locomotion system for the environmental hybrid robot," in *Proceedings of the IASTED International Conference, Pittsburgh, Penn*, 2011, pp. 7–9.

[28] S.-C. Chen, K.-J. Huang, W.-H. Chen, S.-Y. Shen, C.-H. Li, and P.-C. Lin, "Quattroped: a leg–wheel transformable robot," *IEEE/ASME Transactions On Mechatronics*, vol. 19, no. 2, pp. 730–742, 2013.

[29] W. Chen, H. Lin, Y. Lin, and P. Lin, "Turboquad: A novel leg–wheel transformable robot with smooth and fast behavioral transitions," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1025–1040, 2017.

[30] C. Zheng and K. Lee, "Wheeler: Wheel-leg reconfigurable mechanism with passive gears for mobile robot applications," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9292–9298.

[31] C. Semini, N. G. Tsagarakis, E. Guglielmino, M. Focchi, F. Cannella, and D. G. Caldwell, "Design of hyq–a hydraulically and electrically actuated quadruped robot," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 225, no. 6, pp. 831–849, 2011.

[32] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2245–2252.

[33] N. Paine, J. S. Mehling, J. Holley, N. A. Radford, G. Johnson, C.-L. Fok, and L. Sentis, "Actuator control for the nasa-jsc valkyrie humanoid robot: A decoupled dynamics approach for torque control of series elastic robots," *Journal of field robotics*, vol. 32, no. 3, pp. 378–396, 2015.

[34] H. Liu, S. Cui, Y. Liu, Y. Ren, and Y. Sun, "Design and vibration suppression control of a modular elastic joint," *Sensors*, vol. 18, no. 6, p. 1869, 2018.

[35] K. Lochan, B. Roy, and B. Subudhi, "A review on two-link flexible manipulators," *Annual Reviews in Control*, vol. 42, pp. 346–367, 2016.

[36] B. Yüksel, N. Staub, and A. Franchi, "Aerial robots with rigid/elastic-joint arms: Single-joint controllability study and preliminary experiments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1667–1672.

[37] B. Bischof, T. Glück, M. Böck, and A. Kugi, "Path following control for elastic joint robots," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4806–4811, 2017.

[38] A. Kakogawa and S. Ma, "A differential elastic joint for multi-linked pipeline inspection robots," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 949–954.

[39] T. Zhang, H. Huang, H. Guo, and B. Li, "Singularity avoidance for a deployable mechanism using elastic joints," *Journal of Mechanical Design*, vol. 141, no. 9, 2019.

[40] A. Rosyid and B. El-Khasawneh, "Multibody dynamics of nonsymmetric planar 3prr parallel manipulator with fully flexible links," *Applied Sciences*, vol. 10, no. 14, p. 4816, 2020.

[41] E. Hoffman, B. Clément, C. Zhou, N. Tsagarakis, J.-B. Mouret, and S. Ivaldi, "Whole-body compliant control of icub: first results with opensot," 2018.

[42] S.-C. Wang, H. Hikita, H. Kubo, Y.-S. Zhao, Z. Huang, and T. Ifukube, "Kinematics and dynamics of a 6 degree-of-freedom fully parallel manipulator with elastic joints," *Mechanism and Machine Theory*, vol. 38, no. 5, pp. 439–461, 2003.

[43] T. Yoneyama and K. Ito, "Proposal of a flexible translational joint mechanism for humanoid robots," in *ICCAS 2010*, 2010, pp. 1034–1039.

[44] A. Kakogawa and S. Ma, "Design of a multilink-articulated wheeled pipeline inspection robot using only passive elastic joints," *Advanced Robotics*, vol. 32, no. 1, pp. 37–50, 2018.

[45] K. Kong, J. Bae, and M. Tomizuka, "Control of rotary series elastic actuator for ideal force-mode actuation in human–robot interaction applications," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 1, pp. 105–118, 2009.

[46] D. W. Robinson, J. E. Pratt, D. J. Paluska, and G. A. Pratt, "Series elastic actuator development for a biomimetic walking robot," in *1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Cat. No.99TH8399)*, 1999, pp. 561–568.

[47] J. E. Pratt, B. T. Krupp, C. J. Morse, and S. H. Collins, "The roboknee: an exoskeleton for enhancing strength and endurance during walking," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 3. IEEE, 2004, pp. 2430–2435.

[48] A. De Luca, A. Isidori, and F. Nicolo, "Control of robot arm with elastic joints via nonlinear dynamic feedback," in *1985 24th IEEE conference on decision and control*. IEEE, 1985, pp. 1671–1679.

[49] P. Tomei, "A simple pd controller for robots with elastic joints," *IEEE Transactions on automatic control*, vol. 36, no. 10, pp. 1208–1213, 1991.

[50] M. Makarov, M. Grossard, P. Rodríguez-Ayerbe, and D. Dumur, "Modeling and preview h_∞ control design for motion control of elastic-joint robots with uncertainties," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 10, pp. 6429–6438, 2016.

[51] A. Calanca, R. Muradore, and P. Fiorini, "Impedance control of series elastic actuators: Passivity and acceleration-based control," *Mechatronics*, vol. 47, pp. 37–48, 2017.

[52] M. C. Özparpucu, "Optimal control strategies for maximizing the link velocity of elastic joints with nonlinear impedance," Ph.D. dissertation, ETH Zurich, 2018.

[53] A. T. Azar, F. E. Serrano, A. Koubâa, N. A. Kamal, S. Vaidyanathan, and A. Fekik, "Adaptive terminal-integral sliding mode force control of elastic joint robot manipulators in the presence of hysteresis," in *International Conference on Advanced Intelligent Systems and Informatics*. Springer, 2019, pp. 266–276.

[54] M. Ruderman, "On stability of virtual torsion sensor for control of flexible robotic joints with hysteresis," *Robotica*, vol. 38, no. 7, pp. 1191–1204, 2020.

[55] L. Zollo, B. Siciliano, A. De Luca, E. Guglielmelli, and P. Dario, "Compliance Control for an Anthropomorphic Robot with Elastic Joints: Theory and Experiments," *Journal of Dynamic Systems, Measurement, and Control*, vol. 127, no. 3, pp. 321–328, 07 2004. [Online]. Available: https://doi.org/10.1115/1.1978911

[56] G. Ferretti, G. Magnani, and P. Rocco, "Impedance control for elastic joints industrial manipulators," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 3, pp. 488–498, 2004.

[57] M. Yin, H. Wu, Z. Xu, W. Han, and Z. Zhao, "Compliant control of single tendon-sheath actuators applied to a robotic manipulator," *IEEE Access*, vol. 8, pp. 37 361–37 371, 2020.

[58] T. Oomichi and T. Ibe, "Development of vehicles with legs and wheels," *Advanced Robotics*, vol. 1, no. 4, pp. 343–356, 1986. [Online]. Available: https://doi.org/10.1163/156855386X00210

[59] H. Kimura, E. Nakano, and Y. Nonaka, "Development of leg-wheel robot and cooperational motion of legs and wheels," *Journal of the Robotics Society of Japan*, vol. 10, no. 4, pp. 520–525, 1992.

[60] N. Eiji and N. Sei, "Leg-wheel robot: a futuristic mobile platform for forestry industry," in *Proceedings of 1993 IEEE/Tsukuba International Workshop on Advanced Robotics*, 1993, pp. 109–112.

[61] S. Nakajima and E. Nakano, "Adaptive gait for large rough terrain of a leg-wheel robot (fourth report: step-over gait)," *Journal of Robotics and Mechatronics*, vol. 21, no. 2, p. 285, 2009.

[62] S. Nakajima, "RT-Mover: a rough terrain mobile robot with a simple leg–wheel hybrid mechanism," *The International Journal of Robotics Research*, vol. 30, no. 13, pp. 1609–1626, 2011.

[63] A. Halme, I. Leppänen, S. Salmi, and S. Ylönen, "Hybrid locomotion of a wheel-legged machine," in *3rd Int. Conference on Climbing and Walking Robots (CLAWAR'00)*, 2000.

[64] A. Halme, I. Leppänen, J. Suomela, S. Ylönen, and I. Kettunen, "WorkPartner: interactive human-like service robot for outdoor applications," *The international journal of robotics Research*, vol. 22, no. 7-8, pp. 627–640, 2003.

[65] J. Szrek and P. Wójtowicz, "Idea of wheel-legged robot and its control system design," *Bulletin of the Polish Academy of Sciences. Technical Sciences*, vol. 58, no. 1, pp. 43–50, 2010.

[66] A. Gronowicz and J. Szrek, "Design of LegVan wheel-legged robot's mechanical and control system," in *SYROM 2009*. Springer, 2010, pp. 145–158.

[67] J. A. Smith, I. Sharf, and M. Trentini, "PAW: a hybrid wheeled-leg robot," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE, 2006, pp. 4043–4048.

[68] A. A. Hayat, K. Elangovan, M. Rajesh Elara, and M. S. Teja, "Tarantula: Design, modeling, and kinematic identification of a quadruped wheeled robot," *Applied Sciences*, vol. 9, no. 1, p. 94, 2019.

[69] R. Parween, A. A. Hayat, K. Elangovan, K. G. S. Apuroop, M. Vega Heredia, and M. R. Elara, "Design of a self-reconfigurable drain mapping robot with level-shifting capability," *IEEE Access*, vol. 8, pp. 113 429–113 442, 2020.

[70] T. Yoshioka, T. Takubo, T. Arai, and K. Inoue, "Hybrid locomotion of leg-wheel ASTERISK H," *Journal of Robotics and Mechatronics*, vol. 20, no. 3, pp. 403–412, 2008.

[71] L. Bruzzone and P. Fanghella, "Mantis: hybrid leg-wheel ground mobile robot," *Industrial Robot: An International Journal*, 2014.

[72] L. Bruzzone, P. Fanghella, and G. Quaglia, "Experimental performance assessment of mantis 2, hybrid leg-wheel mobile robot," *International Journal of Automation Technology*, vol. 11, no. 3, pp. 396–403, 2017.

[73] M. Schwarz, T. Rodehutskors, M. Schreiber, and S. Behnke, "Hybrid driving-stepping locomotion with the wheeled-legged robot momaro," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5589–5595.

[74] M. Schwarz, T. Rodehutskors, D. Droeschel, M. Beul, M. Schreiber, N. Araslanov, I. Ivanov, C. Lenz, J. Razlaw, S. Schüller, D. Schwarz, A. Topalidou-Kyniazopoulou, and S. Behnke, "NimbRo Rescue: Solving disaster-response tasks through mobile manipulation robot momaro," *CoRR*, vol. abs/1810.01345, 2018. [Online]. Available: http://arxiv.org/abs/1810.01345

[75] J.-M. Jehanno, A. Cully, C. Grand, and J.-B. Mouret, "Design of a wheel-legged hexapod robot for creative adaptation," in *Mobile Service Robotics*. World Scientific, 2014, pp. 267–276.

[76] N. Kashiri, L. Baccelliere, L. Muratore, A. Laurenzi, Z. Ren, E. M. Hoffman, M. Kamedula, G. F. Rigano, J. Malzahn, S. Cordasco *et al.*, "CENTAURO: A hybrid locomotion and high power resilient manipulation platform," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1595–1602, 2019.

[77] M. Bjelonic, C. D. Bellicoso, Y. de Viragh, D. Sako, F. D. Tresoldi, F. Jenelten, and M. Hutter, "Keep rollin'—whole-body motion control and planning for wheeled quadrupedal robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2116–2123, 2019.

[78] M. Bjelonic, R. Grandia, O. Harley, C. Galliard, S. Zimmermann, and M. Hutter, "Whole-body mpc and online gait sequence generation for wheeled-legged robots," 2020.

[79] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch *et al.*, "ANYmal-a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 38–44.

[80] X. Li, H. Zhou, H. Feng, S. Zhang, and Y. Fu, "Design and experiments of a novel hydraulic wheel-legged robot (wlr)," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3292–3297.

[81] X. Li, H. Zhou, S. Zhang, H. Feng, and Y. Fu, "Wlr-ii, a hose-less hydraulic wheel-legged robot," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4339–4346.

[82] C. Grand, P. Jarrault, F. B. Amar, and P. Bidaud, "Experimental evaluation of obstacle clearance by a hybrid wheel-legged robot," in *Experimental Robotics*. Springer, 2016, pp. 47–58.

[83] A. Bouton, C. Grand, and F. Benamar, "Design and control of a compliant wheel-on-leg rover which conforms to uneven terrain," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 5, pp. 2354–2363, 2020.

[84] J. Bałchanowski and A. Gronowicz, "Design and simulations of wheel-legged mobile robot," *acta mechanica et automatica*, vol. 6, no. 2, pp. 11–16, 2012.

[85] J. Bałchanowski, "Modelling and simulation studies on the mobile robot with self-leveling chassis," *Journal of Theoretical and Applied Mechanics*, vol. 54, no. 1, 2016.

[86] P. Costa, J. Gonçalves, J. Lima, and P. Malheiros, "Simtwo realistic simulator: a tool for the development and validation of robot software," *Theory and Applications of Mathematics & Computer Science*, vol. 1, no. 1, pp. 17–33, Apr. 2011.

[87] S. J. Miller, "The method of least squares," in *The Probability Lifesaver*. Princeton University Press, 2017, pp. 625–635.

[88] C. De Boor, *A practical guide to splines*. springer-verlag New York, 1978, vol. 27.

[89] J. Niedziela, "Bessel functions and their applications," *University of Tennessee-Knoxville,(Dated: October 29, 2008)*, 2008.

[90] M. A. Johnson, *PID Control Technology*. Springer-Verlag London, pp. 1–46.

[91] A. Visioli, *Basics of PID Control*. Springer-Verlag London, pp. 1–18.

[92] W. Yu, X. Li, and R. Carmona, "A novel PID tuning method for robot control," *Industrial Robot: An International Journal*, 2013.

[93] A. Latif, K. Shankar, and P. T. Nguyen, "Legged fire fighter robot movement using PID," *Journal of Robotics and Control (JRC)*, vol. 1, no. 1, pp. 15–19, 2019. [Online]. Available: https://journal.umy.ac.id/index.php/jrc/article/view/7419

[94] J. Tavoosi and F. Mohammadi, "A 3-PRS parallel robot control based on Fuzzy-PID controller," in *2019 6th International Conference on Control, Instrumentation and Automation (ICCIA)*, 2019, pp. 1–4.

[95] A. H. Khan and S. Li, "Sliding mode control with PID sliding surface for active vibration damping of pneumatically actuated soft robots," *IEEE Access*, vol. 8, pp. 88 793–88 800, 2020.

[96] H. R. Nohooji, "Constrained neural adaptive PID control for robot manipulators," *Journal of the Franklin Institute*, vol. 357, no. 7, pp. 3907–3923, 2020.

[97] K. J. Åström and T. Hägglund, *PID controllers: theory, design, and tuning*. Instrument society of America Research Triangle Park, NC, 1995, vol. 2.

[98] R. J. Vaccaro, *Digital Control: A State-Space Approach*. McGraw-Hill Higher Education, 1995.

[99] A. Visioli, *Anti-windup Strategies*. Springer-Verlag London, pp. 35–60.

[100] T. J. De Vries, W. J. Velthuis, and J. van Amerongen, "Learning feed-forward control: A survey and historical note," *IFAC Proceedings Volumes*, vol. 33, no. 26, pp. 881–886, 2000.

[101] M. Thieffry, A. Kruszewski, O. Goury, T.-M. Guerra, and C. Duriez, "Dynamic Control of Soft Robots," in *IFAC World Congress*, Toulouse, France, Jul 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01558844

[102] C. Sotelo, A. Favela-Contreras, F. Beltrán-Carbajal, G. Dieck-Assad, P. Rodríguez-Cañedo, and D. Sotelo, "A novel discrete-time nonlinear model predictive control based on state space model," *International Journal of Control, Automation and Systems*, vol. 16, no. 6, pp. 2688–2696, 2018.

[103] M. Ławryńczuk and P. Tatjewski, "Offset-free state-space nonlinear predictive control for wiener systems," *Information Sciences*, vol. 511, pp. 127–151, 2020.

[104] C. Van Loan, "Computing integrals involving the matrix exponential," *IEEE transactions on automatic control*, vol. 23, no. 3, pp. 395–404, 1978.

[105] J. Ackermann, "The design of linear control systems in the state space," *at-Automatisierungstechnik*, vol. 20, no. 1-12, pp. 297–300, 1972.

[106] J. Brynjarsdottir and A. O'Hagan, "Learning about physical parameters: The importance of model discrepancy," *Inverse problems*, vol. 30, no. 11, p. 114007, 2014.

[107] A. Dupuis, M. Ghribi, and A. Kaddouri, "Multiobjective genetic estimation of dc motor parameters and load torque," in *2004 IEEE International Conference on Industrial Technology, 2004. IEEE ICIT'04.*, vol. 3. IEEE, 2004, pp. 1511–1514.

[108] M. Mladenov, M. Mock, and K.-E. Grosspietsch, "Fault monitoring and correction in a walking robot using lms filters," in *2008 International Workshop on Intelligent Solutions in Embedded Systems*, 2008, pp. 1–10.

[109] S. Gupta, A. K. Sahoo, and U. K. Sahoo, "Parameter estimation of wiener nonlinear model using least mean square (lms) algorithm," in *TENCON 2017 - 2017 IEEE Region 10 Conference*, 2017, pp. 1399–1403.

[110] S. Dwivedi, P. Aggarwal, and A. K. Jagannatham, "Fast block LMS and RLS-Based parameter estimation and two-dimensional imaging in monostatic mimo radar systems with multiple mobile targets," *IEEE Transactions on Signal Processing*, vol. 66, no. 7, pp. 1775–1790, 2018.

[111] Y. Wang, G. Gunter, and D. B. Work, "Online parameter estimation of adaptive cruise control models with delays and lags," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–6.

[112] G. Liu and C. Hu, "Adaptive modeling of brushless dc motor drive system," in *2018 IEEE 4th International Conference on Control Science and Systems Engineering (ICCSSE)*. IEEE, 2018, pp. 173–177.

[113] R. d. F. Campos, E. Couto, J. de Oliveira, and A. Nied, "On-line parameter identification of an induction motor with closed-loop speed control using the least square method," *Journal of Dynamic Systems, Measurement, and Control*, vol. 139, no. 7, 2017.

[114] A. Montazeri, C. West, S. D. Monk, and C. J. Taylor, "Dynamic modelling and parameter estimation of a hydraulic robot manipulator using a multi-objective genetic algorithm," *International Journal of Control*, vol. 90, no. 4, pp. 661–683, 2017.

[115] A. González, A. Cerda-Lugo, A. Cardenas, M. Maya, and D. Piovesan, "A third-order model of hip and ankle joints during balance recovery: Modeling and parameter estimation," *Journal of Computational and Nonlinear Dynamics*, vol. 14, no. 10, 2019.

[116] S. S. Saab and R. A. Kaed-Bey, "Parameter identification of a dc motor: an experimental approach," in *ICECS 2001. 8th IEEE International Conference on Electronics, Circuits and Systems (Cat. No. 01EX483)*, vol. 2. IEEE, 2001, pp. 981–984.

[117] J. Gonçalves, J. Lima, and P. G. Costa, "Dc motors modeling resorting to a simple setup and estimation procedure," in *CONTROLO'2014–Proceedings of the 11th Portuguese Conference on Automatic Control*. Springer, 2015, pp. 441–447.

[118] B. Nayak and S. Sahu, "Parameter estimation of dc motor through whale optimization algorithm," *International Journal of Power Electronics and Drive Systems*, vol. 10, no. 1, p. 83, 2019.

[119] W. Ge, B. Wang, and H. Mu, "Dynamic parameter identification for reconfigurable robot using adaline neural network," in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE, 2019, pp. 319–324.

[120] R. Miranda-Colorado and J. Moreno-Valenzuela, "Experimental parameter identification of flexible joint robot manipulators," *Robotica*, vol. 36, no. 3, pp. 313–332, 2018.

[121] H. Ni, C. Zhang, T. Hu, T. Wang, Q. Chen, and C. Chen, "A dynamic parameter identification method of industrial robots considering joint elasticity," *International Journal of Advanced Robotic Systems*, vol. 16, no. 1, p. 1729881418825217, 2019.

[122] E. Eskinat, S. H. Johnson, and W. L. Luyben, "Use of hammerstein models in identification of nonlinear systems," *AIChE Journal*, vol. 37, no. 2, pp. 255–268, 1991.

[123] I. Soares, , V. H. Pinto, J. Lima, and P. Costa, "Realistic 3d simulation of a hybrid legged-wheeled robot (accepted for publication)," in *24th International Conference on Climbing and Walking Robots and Support Technologies for Mobile Machines (CLAWAR 2021)*, 2021.

[124] I. Soares, "Trajectory Control and Obstacle Avoidance in a Vehicle with Hybrid Locomotion (unpublished)," Master's thesis, Faculty of Engineering - University of Porto, Porto, Portugal, 2021.