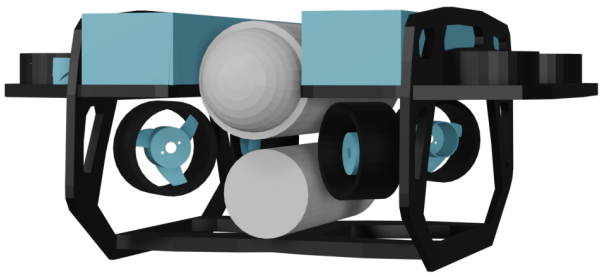
TÉCNICO LISBOA



Advanced control and navigation of ROV/AUVs

Eduardo Coelho Martins de Almeida Cunha

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: David Cabecinhas
António Pascoal

Examination Committee

Chairperson:
Supervisor: David Cabecinhas
António Pascoal
Members of the Committee: Joel Oliveira Reis

October 2022

Declaration

I declare that this document is an original work of my own authorship and that it fulfils all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgements

This thesis represents the cumulative work and effort of five years at Instituto Superior Técnico. Looking back to the last months of 2017, I still remember how frightened and anxious I was when I first started this degree. Everything seemed too much to take in and I often wondered if I would be able to keep up with all the pressure, while still being able to meet the academic standards I had set for myself long ago. But the most important lesson I learnt during all these years was that no journey should be taken alone and that a hard laugh with the ones you care the most about is invaluable.

First and foremost, I can't go on without expressing my deepest gratitude to my girlfriend, Irina. She has been a pillar of strength for me and stood by my side during all the hardships I faced, always available to offer her sincere and kind words of comfort and emotional support. Thank you for bringing me down to Earth (and out of the sea!) and remember me that life is much more than what meets the eye at first glance.

I would also like to thank my parents, Maria Leonor and Henrique, my sister, Carolina, and my grandparents, Laura e Joaquim, for always being there for me. Thank you Mom and Dad for helping me grow up and be the person I am today and for keep on pushing me to achieve the best that I can and never shoot lower than the stars. Thank you Carolina for helping me and warning me about how difficult and full of struggles the IST journey would be. Thank you Grandma and Grandpa, for always worrying about my studies, my health and everything else. Thank you Grandpa for teaching me how to play the Draughts, "Bisca" and the Dominoes and how to fix a roller shutter; thank you Grandma for teaching me from a young age all arithmetic calculations, for taking care of me and for always preparing the most wonderful dishes.

I now have to express how thankful I am for the amazing supervisors I had to guide me through this thesis. Thank you so much Professor David Cabecinhas for all the guidance throughout the last 16 months - ever since the beginning of this endeavour, you have kept answering all my doubts and pushing me forward beyond the boundary of what I thought were the limits of my own perseverance. Thank you Professor António Pascoal, for inspiring me during Control classes and making me see each problem through the most diverse perspectives, which only paved the way to the flourishing of my interest for control theory and robotics. To both of you, thank you so much for trusting me with such a challenging thesis, which let me explore areas of knowledge I had never thought before and revisit and deepen the understanding of subjects I had once learnt before.

It is a foregone conclusion that I would not have been able to take this work to the next level without the

help and support from the DSOR Team. Thank you Marcelo Jacinto, David Souto and João Quintas for helping me take the first steps in this brand new world, for always being ready to give me a helping hand and for your never-ending wisdom. Thank you to André Potes, Luís Sebastião, Manuel Rufino, Francisco Branco, Francisco Rego and Helena Santana for all the help in everything I needed to fit in the team and make the best out of this work.

Also, I need to thank the whole team from EMEPC (Task Group for the Extension of the Continental Shelf) for letting me work at their facilities and do all the experimental tests with the BlueROV vehicle at their tank. Their contribution was invaluable and greatly increased the quality of this thesis.

I would also like to express my immense gratitude to Alice Rosa and Pedro Pedrosa, with whom we formed together JEEC|23's Coordination. These two people walked with me through one of the most challenging and nerve-wracking experiences of my life, dealing with months and months of stress, sleepless days and endless responsibilities. A word of infinite thankfulness also goes to Ricardo Espadinha, for your experience and "big-brain" advices. This thesis could not have come to fruition without any of you.

Furthermore, I also need to mention my highschool's "squad". I have to thank you for all the moments we have had in the last 12 years and for keeping in touch even after our journey together ended. Many more adventures lie in front of us, which I hope to traverse with you.

Finally, some people with whom I no longer keep in touch as often also deserve to have their names in this thesis' acknowledgements, since their impact on my life definitely changed my perspectives on the world and made me the person I am today. I would like to thank my primary school's teacher, William Cardoso, and my middle school's Mathematics teacher, João Elias.

Abstract

In the last decades, Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs) have become tools par excellence for the execution of scientific, commercial and military missions at sea, due to their versatility and relatively low cost. This work addresses the problem of endowing such vehicles with the capability to follow specified paths automatically, while being robust to model uncertainty and external disturbances. This thesis starts with a brief overview of motion control problems, followed by the guidance outer-loop, the solutions adopted for inner-loop control using Sliding Mode Control (SMC) and thrust allocation methods. Two marine robotic vehicles take central stage in this work, MEDUSA and BlueROV, for which kinematic and dynamic models are introduced. This is followed by the design of two types of inner-loop controllers resorting to SMC strategies. These controllers are first implemented in *MATLAB* software for rapid iteration and finally coded in *C*++/ROS framework, enabling their application in real vehicles. Simulation results are performed with the objective of assessing the performance of the controllers designed and how it compares with the performance of existing Proportional-Integral-Derivative (PID) control laws. The results obtained show that SMC laws exhibit a good degree of robustness against model uncertainty, together with external disturbance attenuation. Furthermore, two thrust allocation methods are also compared and analysed. The SMC controllers are tested in a real environment with BlueROV, attesting to their adequacy as a robust control approach for ROVs/ AUVs. Finally, future work is described, focusing on promising developments in this field of work.

Keywords: Nonlinear Control, Autonomous Underwater Vehicle (AUV), Remotely Operated Vehicle (ROV), Sliding Mode Control (SMC)

Resumo

Nas últimas décadas, Veículos Remotamente Operados (ROVs) e Veículos Autónomos Subaquáticos (AUVs) tornaram-se ferramentas de excelência para a execução de missões no âmbito científico, comercial e militar, dado que estes veículos são extremamente versáteis e de custo relativamente baixo. Este projeto foca-se no problema de capacitar estes veículos de forma a que sigam certos caminhos de forma automática, focando-se em eliminar a influência de incertezas de modelação e atenuar perturbações externas. Começa-se por fazer uma breve introdução a problemas de controlo de movimento, seguido do "outer-loop", das soluções adotadas para o controlo de "inner-loop", usando Sliding Mode Control (SMC) e dos métodos de alocação de forças. Os veículos marinhos MEDUSA e BlueROV são introduzidos, bem como os seus modelos cinemático e dinâmico. Para além disso, dois controladores baseados em SMC são desenvolvidos e implementados em software *MATLAB* para rápidas iterações e em *C*++/ "ROS framework", possibilitando a sua aplicação em veículos reais. Os resultados de simulação têm como objetivo a avaliação do desempenho dos controladores desenvolvidos e comparar o mesmo com um controlador Proporcional-Integral-Derivativo (PID) existente. Estes resultados mostram um bom nível de robustez do SMC relativamente a incertezas de modelação, bem como capacidade de atenuação de perturbações externas. Dois métodos de alocação de forças são também comparados e analisados. Seguidamente, os controladores baseados em SMC são testados num ambiente real com o veículo BlueROV, atestando a sua capacidade de controlo robusto em ROVs/AUVs. Finalmente, é feito um sumário do trabalho futuro na área, focando-se em desenvolvimentos bastante promissores.

Palavras-Chave: Controlo Não-Linear, Autonomous Underwater Vehicle (AUV), Remotely Operated Vehicle (ROV), Sliding Mode Control (SMC)

Contents

Contents **xi**

List of Figures **xiv**

List of Tables **xvi**

Acronyms **xvii**

1 Introduction **1**

1.1 Motivation and Purpose 1

1.2 Problem Description and Objectives 2

1.2.1 AUV uses in extreme environments 2

1.2.2 Motion Control in the presence of Model Uncertainty and External Disturbances . 3

1.2.3 Motion Control Solutions 3

1.2.4 Main Objectives 3

1.2.5 Key Contributions 4

1.3 Thesis Outline 4

2 State of the Art **5**

2.1 Trajectory Tracking and Path Following 5

2.1.1 Trajectory Tracking vs. Path Following 5

2.1.2 Line of Sight Algorithm 5

2.1.3 Aguiar and Rómulo algorithms 6

2.2 Control Methods 7

2.2.1 Proportional-Integral-Derivative (PID) Control 8

2.2.2 Gain Scheduling Control 9

2.2.3 Iterative Learning Control 9

2.2.4 Sliding Mode Control (SMC) 10

2.2.5 Quasi-Sliding Mode 14

2.2.6 SMC Smoothing Solution 15

2.3 Actuation on Underwater Vehicles 16

2.3.1 Underactuated vs. Fully Actuated Underwater Vehicle 16

2.3.2 Allocation Methods for Underwater Vehicles . . . 17

3 Vehicle Model **21**

3.1 Reference Frames and General Notation . . . 21
3.2 AUV Kinematics . . . 22
3.2.1 Euler Angles . . . 22
3.2.2 Unit Quaternions . . . 23
3.3 AUV Dynamics . . . 24
3.4 AUV Simplified Motion . . . 24
3.5 MEDUSA-class Vehicles . . . 25
3.5.1 MEDUSA - Overactuated (3 DOF) . . . 26
3.6 BlueROV vehicle . . . 27
3.7 Thrust Allocation . . . 27
3.7.1 6 DOF Vehicle . . . 28
3.7.2 3 DOF Vehicle . . . 28

4 SMC Design and Application **29**

4.1 Simple SMC Controllers . . . 29
4.1.1 Surge Velocity u . . . 29
4.1.2 Sway Velocity v . . . 30
4.1.3 Yaw Angle ψ . . . 30
4.2 Improved SMC Controllers . . . 31
4.2.1 Surge Velocity u . . . 31
4.2.2 Sway Velocity v . . . 32
4.2.3 Yaw Angle ψ . . . 33
4.3 SMC parameters' influence on controller performance . . . 33
4.3.1 Parameter ϵ . . . 33
4.3.2 Parameter Δ . . . 34
4.3.3 Parameters c and λ . . . 34
4.3.4 Parameter k_c . . . 34

5 Simulation **35**

5.1 *MATLAB*/*Simulink* Implementation . . . 35
5.1.1 Software Simulation . . . 35
5.1.2 Inertial to Body Frame Transformation . . . 35
5.1.3 Thrust Allocation and Actuator Dynamics . . . 36
5.1.4 MEDUSA Model . . . 36
5.1.5 State Sensor . . . 36
5.1.6 Results and Discussion . . . 36
5.2 *C*++/ROS Implementation . . . 39

5.2.1 Farol Stack . . . 39
5.2.2 SMC . . . 40
5.2.3 Thrust Allocation with Optimisation Methods . . . 41
5.2.4 Results and Discussion - SMC and PID . . . 42
5.2.5 Results and Discussion - Unconstrained and Constrained Allocation . . . 58

6 Experimental Results **63**
6.1 Location . . . 63
6.2 Results and Discussion - SMC . . . 64
6.2.1 Constant References . . . 64
6.2.2 Path Following Missions . . . 65

7 Conclusion **69**
7.1 Future Work . . . 70

References **71**

A MEDUSA Motor References **75**

List of Figures

1.1 KAIKO (かいこう) ROV before a mission in the Pacific Ocean. 2

2.1 Geometric illustration of path following (adapted from [12]). 6
2.2 Conventional Feedback Structure (adapted from [3]). 8
2.3 SMC 2D state-space $\{x\ \dot{x}\}$ phase portrait for two initial states; s is the sliding surface and t_1 and t_2 are state space trajectories. 10
2.4 Sign function. 14
2.5 Saturation function. 15
2.6 Relay function. 15

3.1 Reference Frames used, according to [7]. 21
3.2 Example of a MEDUSA vehicle (3D Model). 26
3.3 BlueROV vehicle: heavy configuration (3D Model). 27

5.1 Simulated System block diagram. 35
5.2 Actuator Dynamics block diagram. 36
5.3 Comparison between improved and simple SMC solutions - reference tracking. 38
5.4 Comparison between improved and simple SMC solutions - controller generated forces and torques. 38
5.5 Inner-outer loop structure, considering Simplified Motion as in Section 3.4. 39
5.6 Graphical interfaces integrated with the Farol stack. 40
5.7 Yaw references - comparison between SMC and PID. 42
5.8 Surge references - comparison between SMC and PID. 43
5.9 Surge references - thrusters' RPM command for SMC's case. 43
5.10 Sway references - comparison between SMC and PID. 44
5.11 Yaw Controller in the presence of a constant disturbance - comparison between SMC for different values of ϵ_ψ and PID. 45
5.12 Lawnmower Path Following with $u_d = 0.3$ m/s - comparison between SMC and PID. 47
5.13 Lawnmower Path Following with $u_d = 0.3$ m/s: Cross-track error and surge velocity - comparison between SMC and PID. 48
5.14 Lawnmower Path Following with $u_d = 1.0$ m/s - comparison between SMC and PID. 49

5.15 Lawnmower Path Following with u_d = 1.0 m/s: Cross-track error and surge velocity - comparison between SMC and PID. 50
5.16 Star Path Following with u_d = 0.3 m/s - comparison between SMC and PID. 51
5.17 Star Path Following with u_d = 0.3 m/s: Cross-track error and surge velocity - comparison between SMC and PID. 52
5.18 Star Path Following with u_d = 1.0 m/s - comparison between SMC and PID. 53
5.19 Star Path Following with u_d = 1.0 m/s: Cross-track error and surge velocity - comparison between SMC and PID. 55
5.20 Star Path Following with u_d = 1.0 m/s - thrusters' RPM command for SMC's case. 56
5.21 Circumference Path Following with profile velocity of 0.3 m/s - comparison between SMC and PID. 57
5.22 Circumference Path Following with profile velocity of 1.0 m/s - comparison between SMC and PID. 57
5.23 Thruster saturation and its impact on the vehicle's behaviour - static thruster allocation. . . 59
5.24 Thruster saturation and its impact on the vehicle's behaviour - optimised thruster allocation. 60

6.1 The BlueROV vehicle during experimental tests inside EMEPC's tank. 63
6.2 Yaw SMC - performance in response to constant references. 64
6.3 Surge and sway SMC - performance in response to constant references. 65
6.4 Lawnmower Path Following with profile velocity of 0.1 m/s. 66
6.5 Circumference Path Following with profile velocity of 0.2 m/s. 66

A.1 MEDUSA Motor References; lengths in milimetres. 75

List of Tables

3.1 Notation used (adapted from Fossen et al. [7]). 22
3.2 MEDUSA's parameters. 26
3.3 BlueROV's parameters. 27

5.1 Simple SMC controllers - parameters for the *MATLAB* implementation. 37
5.2 Improved SMC controllers - parameters for the *MATLAB* implementation. 37
5.3 SMC's parameters for the *C*++/ROS implementation. 42
5.4 Performance of SMC for different values of ϵ_ψ, with and without a constant disturbance in yaw torque (values in degrees). 46

Acronyms

ROV	Remotely Operated Vehicle
AUV	Autonomous Underwater Vehicle
VSC	Variable Structure Control
VSCS	Variable Structure Control System
SMC	Sliding Mode Control
ISMC	Integral Sliding Mode Control
PID	Proportional-Integral-Derivative
DOF	Degree-of-Freedom
LS	Least-Squares
QP	Quadratic Programming
NED	North-East-Down
IST	Instituto Superior Técnico
ISR	Institute for Systems and Robotics
DSOR	Dynamical Systems and Ocean Robotics Laboratory
LARSyS	Laboratory for Robotics and Engineering Systems
EMEPC	Task Group for the Extension of the Continental Shelf
ROS	Robot Operating System

Chapter 1

Introduction

1.1 Motivation and Purpose

The ocean is an immensely vast body of water, covering up to 70.8% of the Earth's surface [24]. The ocean has been a crucial resource for Mankind ever since the dawn of Humanity and there is evidence that it played a significant role in the emergence of life on Earth, 3.77 billion years ago [5].

Even though ocean exploration dates as far back as 4500 BC [20], it is a fact that more than 80% of it is still unmapped, unobserved and unexplored [22]. The reason for this is twofold - first, humans are biologically unable to reach such deep waters without extra equipment; second, attempts to use robots for such operations are met with formidable difficulties since their deployment, operation, and retrieval can be extremely difficult and perilous. In order to overcome these hurdles, tremendous progress has been made over the past decades towards the development and operation of Remotely Operated Vehicles (ROVs) for seafloor observation and inspection and intervention on underwater infrastructures and Autonomous Underwater Vehicles (AUVs) for marine habitat mapping and oceanographic studies.

Over the years, ROVs/AUVs have grown in popularity due to their operational versatility, ranging from military applications (maritime surveillance and reconnaissance, harbour security, meteorology and other underwater inspection tasks) to scientific uses (study of deep sea fauna and flora, underwater archaeology projects, to name a few) [4, 10, 6, 11]. Also, being able to perform this type of tasks with ROVs/AUVs is unquestionably advantageous, since these vehicles are much cheaper than traditional crewed marine vehicles, such as regular submarines. Moreover, they are easier to transport from one place to another, making them simpler to deploy and use in most real-world situations, without the need of a large scientific ship and a numerous crew, which is needed for most Work-Class vehicles such as described next. ROVs are normally classified according to their function and capacity to interact with the environment:

- **Class I - Observation ROVs**: usually small vehicles, fitted with a camera and lights, designed for observation;
- **Class II – Observation ROVs with Payload Option**: vehicles with multiple sensors and possibly some simple manipulative capability. As described later, the BlueROV fits into this category;

- **Class III - Work-Class Vehicles**: usually larger vehicles, with large capability to interact with the environment using a variety of tools and to reach deeper waters. The ROVLuso, used by the Task Group for the Extension of the Continental Shelf (EMEPC), fits into this category;
- **Class IV - Towed and Bottom-Crawling Vehicles**: large and heavy vehicles, designed for specific tasks and with little manoeuvrability;
- **Class V - Prototype or Development Vehicles**: in-development vehicles and other vehicles not fitting into the previous classes. The MEDUSA vehicle, which is described later, fits in this category, as with all other AUVs.

Encouraged by the importance of this topic, the main goal of the present manuscript is to design a method capable of undertaking the task of autonomously operating an ROV/AUV using state of the art techniques that are rooted in a solid mathematical foundation.

1.2 Problem Description and Objectives

In the last decades, considerable progress has been done in the field of marine robotics, in an effort to make these vehicles more accessible, cheaper and useful in underwater inspections tasks and empower scientists with the tools to unrestrainedly explore and discover the Earth's ocean. Nonetheless, considerable work remains to be done in the field of marine vehicle motion control, to endow underwater robots with the capability to perform in a reliable and efficient manner in extremely harsh environments.

1.2.1 AUV uses in extreme environments

AUVs have been used in several exploration missions in severe environments such as the Arctic. For instance, the Arctic Gakkel Vents Expedition (AGAVE) had the objective of exploring and understanding the marine biology, chemistry and geology of a hydrothermal section of a near-seabed area in the Gakkel Ridge [15]. Furthermore, the Challenger Deep, the deepest place in Earth's surface, which is situated almost 11 kilometres below sea level, has been explored by ROVs/AUVs since 1995, such as KAIKO, a japanese ROV - [29]. This vehicle is shown in Figure 1.1.

Figure 1.1: KAIKO (かいこう) ROV before a mission in the Pacific Ocean.

As stated before, it is clear that this type of missions will greatly benefit from accurate and robust motion control techniques, as a means to afford scientists advanced tools for underwater exploration in extreme environments. Furthermore, effective tracking and good external disturbance rejection facilitates work-class ROV operators, which are typically directly controlled by them, in open-loop.

1.2.2 Motion Control in the presence of Model Uncertainty and External Disturbances

There are two key motion control building blocks at the core of any advanced underwater vehicle, commonly known as Trajectory Tracking and Path Following systems.

In order to make sure that AUV missions are executed successfully, their motion control systems should be designed in such a way as to reduce the closed-loop impact of external disturbances (currents and waves) and sensor noise. They must also exhibit robustness against plant model uncertainty associated with a number of parameters, of which added-mass and damping terms are representative examples.

1.2.3 Motion Control Solutions

One of the most widely used type of controllers is Proportional-Integral-Derivative (PID) control, which has become quite popular in the case of linear plant models due to their simplicity and good performance in a vast number of practical situations. However, it has been shown by Pretagostini et al. in [25] that this type of controllers is highly dependent on gain tuning and may exhibit poor performance in the presence of external disturbances and process noise.

In the past decades, a new breed of controllers that fall in the scope of Sliding Mode Control (SMC) theory have emerged [19] that show good potential to explicitly reject external disturbances and model uncertainty, even in the case of nonlinear plant models.

The aim of this work is, on the one hand, to analyse and develop a state of the art control method called SMC, so as to operate at the system's dynamic level (inner-loop). On the other hand, it is also the focus of this thesis to design, test and simulate a motion control approach for a ROV/AUV both virtually and in the real world, using SMC theory for the inner-loop controllers and Path Following methodologies at the kinematic level, for the outer-loop controllers.

1.2.4 Main Objectives

The work pursued in this thesis unfolds in four main steps:

- **Vehicle Modelling** - derive the laws that dictate the kinematics and dynamics of the ROV/AUV;
- **Vehicle Control** - develop control laws that that yield good performance of a ROV or AUV during path following missions;
- **Simulation** - simulate different vehicles' kinematic and dynamics and realistic sensor and actuator models, together with the complete motion control system, with a view to assessing the expected

performance of the integrated systems and comparing different control approaches;

- **Field Experiments** - test the developed control laws on a real vehicle, in order to assess their performance in a real world environment.

1.2.5 Key Contributions

In this thesis, it is shown that Sliding Mode Control (SMC) strategies applied to marine robotic vehicles offer a good degree of robustness against external disturbances and rejects model uncertainty, presenting a noteworthy alternative to Proportional-Integral-Derivative (PID) control and representing an important contribution to the scientific community.

Also, it is important to mention the assessment of different thrust allocation methods regarding their performance. It is shown that using a Linear Quadratic Constrained Allocation Method in situations of high actuator demand and saturation, in detriment of its unconstrained counterpart, provides the ability to prioritise the success in controlling certain Degrees of Freedom (DOFs) over others.

1.3 Thesis Outline

This document is organised in seven different chapters:

- **Chapter 2: State of the Art** - presents an introduction to trajectory tracking and path following, a review of different control methods with emphasis on Sliding Mode Control (SMC) and analysis of several actuation methods for AUVs;
- **Chapter 3: Vehicle Model** - contains a description of the kinematics and dynamics of the vehicles to be used for controller development and simulation;
- **Chapter 4: SMC Design and Application** - presents the design of SMC controllers and discussion of SMC's parameter tuning influence on controller performance;
- **Chapter 5: Simulation** - simulation-driven performance discussion regarding *MATLAB* and *C++*/ROS implementations;
- **Chapter 6: Experimental Results** - experimental testing of developed controllers with an actual BlueROV vehicle in a real environment, followed by its discussion and analysis;
- **Chapter 7: Future Work** - summary of envisioned developments that warrant future R&D work.

Chapter 2

State of the Art

2.1 Trajectory Tracking and Path Following

In general, autonomously manoeuvring a marine vehicle like an Autonomous Underwater Vehicle (AUV) is achieved by ensuring the craft can follow a specific path, either predetermined or computed on-the-fly, and encompasses two main motion control problems: Trajectory Tracking and Path Following.

2.1.1 Trajectory Tracking vs. Path Following

On the one hand, Trajectory Tracking is concerned with the design of control laws that impose a vehicle to reach and pursue a time parameterised reference $p_{\text{ref}}(t)$, i.e. a geometrically defined path along with an associated timing law [1].

On the other hand, Path Following is defined as the design of control laws such that a vehicle converges and follows a determined path without an associated temporal law [1]. A common method to approach in these situations is to only control the vehicle's steering, in function of a predetermined path-dependent speed law [16]. Mathematically, this path can be represented as $p_{\text{ref}}(\gamma)$, where γ is the vehicle's progression along it and $\dot{\gamma}$ is the referred speed law.

According to Lapierre et al. in [16], typically, Path Following strategies are preferred to Trajectory Tracking ones, as the former are generally more likely to demonstrate smoother convergence and less likely to saturate the control signals. For instance, if the starting position of the craft is relatively far away from the desired path, the references given by the Path Following algorithm will stay the same until the vehicle is closer to the mentioned path; on the contrary, the Trajectory Tracking algorithm would give references depending only on its time parameterisation, making it difficult to ensure convergence. Moreover, Path Following also contributes to eliminate performance restrictions [2].

2.1.2 Line of Sight Algorithm

The Line of Sight Algorithm is a very popular approach to the Path Following control problem. Its main goal is to steer the vehicle in the direction of a point $[x_{\text{los}}\ y_{\text{los}}]^T$ in the desired path that is arbitrarily further

ahead of the vehicle's projection on the specified path [18], such that the vehicle's heading orientation ψ approaches

$$\psi_{\text{los}} = \tan^{-1}\left(\frac{y_{\text{los}} - y}{x_{\text{los}} - x}\right), \tag{2.1}$$

where $[x \ y]^T$ is the position of the vehicle [8].

This arbitrarily chosen distance ahead of the projection of the vehicle on the path is a tuning parameter and can be used to speed up or slow down the path convergence. It should be noticed that a too small distance makes the vehicle go past the desired path, overshooting it and stabilising further up ahead in the path, sometimes after several other overshoots.

2.1.3 Aguiar and Rómulo algorithms

Aguiar ([12], Method 6, p. 19-20) and Rómulo ([12], Section 5, p. 26-27) algorithms are two different strategies to achieve the task of path following, through the implementation of Lyapunov-based controller designs, as reviewed by Hung et. al in [12]. In order to introduce both algorithms, the general path following considerations, notation and formulation should be settled. It is established that, as seen in Figure 2.1, $\{\mathcal{I}\}$ is the inertial frame, $\{\mathcal{B}\}$ is the vehicle's body frame and Q and P are the points in space corresponding to the vehicle and the path's reference which the vehicle should track, respectively. Furthermore, $\mathbf{p}$ and $\mathbf{p}(\gamma)$ are the vehicle's and the reference point's position in the inertial frame $\{\mathcal{I}\}$, while ψ is the vehicle's heading angle. Finally, $\mathbf{e}_{\mathcal{B}}$ is the position error between the vehicle and the path in the vehicle's body frame $\{\mathcal{B}\}$ and $\boldsymbol{\epsilon} = [\epsilon_1 \ \epsilon_2]^T$ is an arbitrarily small non-zero constant vector.

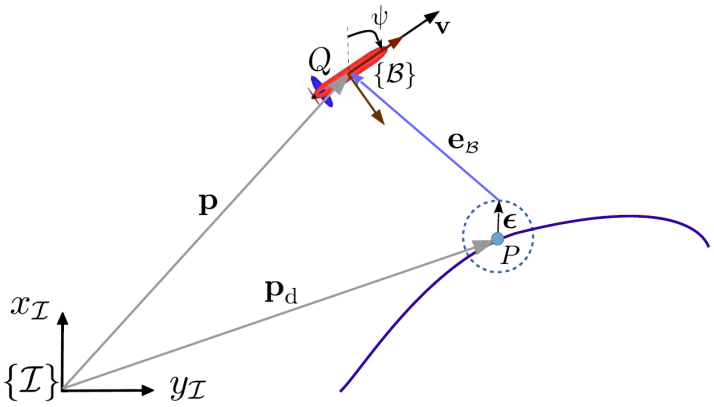


Figure 2.1: Geometric illustration of path following (adapted from [12]).

These two algorithms also differ in what reference values they compute. The Aguiar algorithm outputs a surge velocity and yaw rate, while the Rómulo algorithm computes surge and sway velocities. Also, these algorithms only consider the kinematic model of a vehicle.

Aguiar algorithm

This algorithm computes the desired velocity of the vehicle in its body's x axis, u_d, and the desired vehicle's heading angle rate, r_d, which lends itself to be used in underactuated vehicles, which are defined

later in Section 2.3. Moreover, it also computes the second derivative of the path's progression parameter, $\ddot{\gamma}$, which can then be sequentially integrated in order to obtain the referred path's progression parameter, γ. As reviewed by Hung et. al in [12], the control law for these parameters should be as follows, in (2.2),

$$\begin{aligned} \begin{bmatrix} u_d \\ r_d \end{bmatrix} &= \bar{\Delta}(R_{\mathcal{I}}^{\mathcal{B}}(\psi)\mathbf{p}_d'(\gamma)v_d - K_p\mathbf{e}_{\mathcal{B}}) \\ \ddot{\gamma} &= -k_\gamma e_\gamma + \mathbf{e}_{\mathcal{B}}^T R_{\mathcal{I}}^{\mathcal{B}}(\psi)\mathbf{p}_d'(\gamma) + \dot{v}_d \end{aligned}, \tag{2.2}$$

where v_d is the desired speed of the path parameter γ, $e_\gamma = \dot{\gamma} - v_d$, K_p is a positive definite matrix with appropriate dimensions, k_γ is a positive constant, $\bar{\Delta} = \Delta^T(\Delta\Delta^T)^{-1}$, $\Delta = \begin{bmatrix} 1 & \epsilon_2 \\ 0 & -\epsilon_1 \end{bmatrix}$ and $R_{\mathcal{I}}^{\mathcal{B}}(\psi) = \begin{bmatrix} \cos(\psi) & \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix}$.

In the first equation of (2.2), it can be analysed that matrix $\bar{\Delta}$ weighs the influence of the x and y components of $\mathbf{p}_d'(\gamma)$ and $\mathbf{e}_{\mathcal{B}}$ on the computed references for surge velocity u_d and yaw rate r_d; the y components only affect the desired yaw rate through a weight of $-\epsilon_1$, while the x components affect the desired surge velocity with a factor of 1 and the yaw rate through a factor of ϵ_2. Changing ϵ_1 and ϵ_2 as tuning parameters changes how the computed velocities react to the path's derivative and the error $\mathbf{e}_{\mathcal{B}}$.

In the second equation of (2.2), it can be seen that the derivative of the path's speed $\dot{\gamma}$ is influenced by its own error through parameter k_γ, by the error and path derivative and by the rate of change of the desired speed of γ.

Rómulo algorithm

This algorithm computes the desired velocities of the vehicle in its body's x and y axes, u_d and v_d, respectively, which is suited to fully or overactuated vehicles, as explained in Section 2.3. As in the previous algorithm, the Rómulo algorithm also computes the second derivative of γ. Once again, as reviewed by Hung et. al in [12], the control law for theses parameters should be as follows, in (2.3),

$$\begin{aligned} \begin{bmatrix} u_d \\ v_d \end{bmatrix} &= R_{\mathcal{I}}^{\mathcal{B}}(\psi)\mathbf{p}_d'(\gamma)v_d - K_p\mathbf{e}_{\mathcal{B}} \\ \ddot{\gamma} &= -k_\gamma e_\gamma + \mathbf{e}_{\mathcal{B}}^T R_{\mathcal{I}}^{\mathcal{B}}(\psi)\mathbf{p}_d'(\gamma) + \dot{v}_d \end{aligned}. \tag{2.3}$$

In contrast to the Aguiar algorithm, the matrix $\bar{\Delta}$ is now absent, since the x and y axes components in the body frame of $\mathbf{p}_d'(\gamma)$ and $\mathbf{e}_{\mathcal{B}}$ only influence their respective velocities, surge u_d and sway v_d.

2.2 Control Methods

Control Theory studies the task of controlling dynamic systems, i.e., systems which change through time. This consists in designing adequate input signals such that the system's state is driven towards and

stabilises at a desired state.

Dynamic systems can be divided into two big categories:

- **Linear Systems** - these are systems that obey the superposition principle; these systems can be described by a set of equations as in (2.4),

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}, \tag{2.4}$$

  where $x(t) \in \mathbb{R}^n$ is the system's state, $y(t) \in \mathbb{R}^p$ is its output, $u(t) \in \mathbb{R}^m$ is the input and matrices A through D have appropriate size. A and B are the system's parameters (represented by matrices in multidimensional systems), reflecting the current state's and the input's influence, respectively, in the state's time derivative. C and D are also system's parameters, reflecting the current state's and the input's influence, respectively, in the output.

- **Nonlinear Systems** - these are systems which do not obey the superposition principle; they represent all real-world systems, since in nature nothing is strictly linear, as exemplified by equations (2.5),

$$\begin{cases} \dot{x}(t) = f(x, t, u(t)) \\ y(t) = h(x, t, u(t)) \end{cases}. \tag{2.5}$$

2.2.1 Proportional-Integral-Derivative (PID) Control

Proportional-Integral-Derivative (PID) Control remains to this day the most widely used controller in process control since it was first developed by Elmer Sperry, in 1911 [21]. As in most control methods, the objective of the controller is to compute a control input for the process, such that its output follows a desired reference.

This controller is used in a conventional feedback structure such as in figure 2.2, where it computes the control input u based on the tracking error e, which is the difference between the desired reference r and the output of the process y.

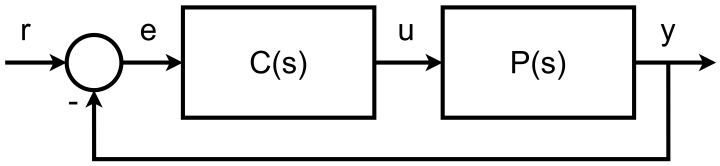


Figure 2.2: Conventional Feedback Structure (adapted from [3]).

The conventional formula for the input control u, based on the tracking error e is given by (2.6),

$$u(t) = k_P e(t) + k_I \int_0^t e(\tau)d\tau + k_D \frac{de}{dt}, \tag{2.6}$$

where $k_P, k_I, k_D > 0$ are tunable gains, reflecting the weight of the proportional (P), integral (I) and derivative (D) terms, respectively.

These terms have each its own purpose in driving the output of the process to the reference. As stated in [3], the term $k_P e(t)$ is proportional to the error at time t, called the present error. The term $k_I \int_0^t e(\tau)d\tau$ is proportional to the integral of the error up to time t, reflecting the past of the error. Finally, the term $k_D \frac{de}{dt}$ is the derivative of the error, which estimates the future of the error based on its current evolution, expressed by its derivative.

Araki M, in [3], defines the PID controller via its transfer function $C(s)$, which relates the input $U(s)$ and the tracking error $E(s)$ such that

$$C(s) = \frac{U(s)}{E(s)} = k_P \left(1 + \frac{1}{T_I s} + T_D D(s)\right), \tag{2.7}$$

where s is the complex variable $j\omega$, $T_I = k_P/k_I$ and $T_D = k_D/k_P$. Ideally, $D(s) = s$, but it is infeasible in a real system to compute a pure derivative of the error e. Thus, this term is substituted by $D(s) = s/[1+(T_D/\gamma)s]$, which is called an approximate derivative.

Although PID controllers are widely used and have a huge range of applications, they are usually sensitive to noise and external perturbations [25]. These controllers are tuned for a specific operating point and have significant performance and stability degradation with strong nonlinearities and large disturbances, which makes PID Control sub-par candidate for accurate motion control of vehicles such as AUVs, with poorly characterised dynamic behaviour and susceptible to varying external disturbances, like waves and sea currents.

2.2.2 Gain Scheduling Control

Although Gain Scheduling includes a broad and unspecified range of controllers, one of the most general definitions is any controller whose coefficients continuously vary according to signals related to the plant whose controller is acting on, which are called scheduling variables [26].

As stated by Rugh et al. [26], this process envolves four main steps: computation of a linear model for the plant, design of linear controllers for the linear model implementation of the controllers with varying coefficients and performance assessment.

Nevertheless, this type of controller has its own disadvantages, as quipped by Shamma et al. in [27]. This includes potential loss of stability and loss of minimum-phase portrait, i.e., the system or its inverse either lose causality or stability. Also, the need for system linearisation limits the range of reference values admissible by the system.

2.2.3 Iterative Learning Control

The idea behind this controller is to improve the input control signal from trial to trial, based on the system's transient response to the current input signal [30]. In a sense, the controller "learns" its best parameters based on experience.

One advantage of this controller is that it is not needed to know the exact plant parameters, i.e., the system model does not need to be too similar to the real one. This is adequate for situations where the

plant model is unknown.

Nonetheless, this type of controller comes with various disadvantages, such as its incapability to know the best controller parameters on the initial trials (in order to have the best system response) and its lack of robustness when dealing with non-repetitive tasks. Also, since the plant model for marine vehicles is relatively well-known, this motion control solution does not seem adequate.

2.2.4 Sliding Mode Control (SMC)

In the past decades, a new type of control methodology called Variable Structure Control System (VSCS) has grown in popularity in the control research community and it has seen great development and applications ever since. VSCS is family of dynamical systems whose structure changes according with the present value of the state of the system it is controlling [31], making it a discontinuous nonlinear function of time. This can be seen as a set of independent structures with switching logics between each of them, which paves the way to taking advantage of the properties of each structure when driving the system to a desired state.

Generally speaking, a system's mathematical model is an idealised version of the real system. In the 1950s, Emelyanov, and other researchers from the Soviet Union like Utkins and Itkis, constructed a VSCS with Sliding Mode Control (SMC), which was meant to tackle modelling discrepancies and serve as a robust control approach [19].

SMC utilises a high-speed switching control law to ensure two main objectives. First, in the so-called *reaching phase*, the plant's state trajectory is driven onto a user-chosen hypersurface in the state space, called the sliding (or switching) surface. Second, the plant's state is kept in this hypersurface for all succeeding time (sliding phase). During this phase, the plant's state constantly switches from one side of the surface to the other, *sliding* along its vicinity until it reaches the desired equilibrium [19]. This behaviour is exemplified in figure 2.3.

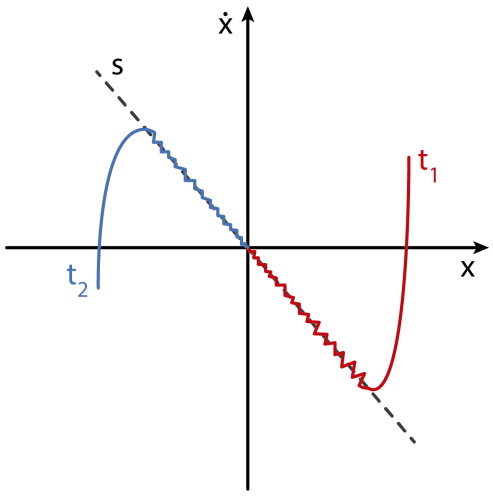


Figure 2.3: SMC 2D state-space $\{x\ \dot{x}\}$ phase portrait for two initial states; s is the sliding surface and t_1 and t_2 are state space trajectories.

Sliding Surface Design

As suggested by Liu et al. in [19], the sliding surface $s(\mathbf{x})$ can be described as

$$s(\mathbf{x}) = \mathbf{c}^T\mathbf{x} = \sum_{i=1}^{n} c_i x_i = \sum_{i=1}^{n-1} c_i x_i + x_n, \tag{2.8}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector and $\mathbf{c} = [c_1 \ \dots \ c_{n-1} \ 1]^T$. In this representation, the parameters in $\mathbf{c}$ should be selected such that the polynomial $p^{n-1} + c_n p^{n-2} + \cdots + c_2 p + c_1$ is a Hurwitz polynomial, where p is a Laplace operator, which means that the eigenvalues of the polynomial must have a negative real part. This ensures that, when the plant's state $\mathbf{x}$ reaches the sliding surface ($s(\mathbf{x}) = 0$), the equilibrium point is reached at least exponentially, i.e., the origin is exponentially stable. It is important to notice that, in an implementation phase, the state vector $\mathbf{x}$ would be replaced by the tracking error, such that the sliding surface drives the plant's state to the desired equilibrium.

In an improvement on (2.8), as suggested by Pan et al. in [23], the sliding surface s can be expressed as

$$s(\mathbf{x}) = s_0(\mathbf{e}) + z(\mathbf{e}, t), \tag{2.9}$$

where $s_0(\mathbf{e})$ is the conventional sliding surface and $z(\mathbf{e}, t)$ an added term which ensures an exponential reaching law. Reaching laws will be described in more detail in the sequel.

One simple expression for the conventional sliding surface $s_0(\mathbf{e})$ is

$$s_0(\mathbf{e}) = (\frac{d}{dt} + \lambda)^{n-1} e = [\Lambda^T \ 1]\mathbf{e}, \tag{2.10}$$

where $\mathbf{e}$ is the vector of the tracking error

$$e = x - x_d, \tag{2.11}$$

and its derivatives (considering x_d to be the desired state), up to the $(n-1)$th degree, such that

$$\mathbf{e} = [e \ e^{(1)} \ \dots \ e^{(n-1)}]^T, \quad e^{(i)} = \frac{d^i}{dt^i} e, \tag{2.12}$$

λ is a positive constant, n is the degree of the sliding surface and Λ is described as

$$\Lambda = [\lambda^{n-1} \ (n-1)\lambda^{n-2} \ \dots \ (n-1)\lambda]^T. \tag{2.13}$$

The matrix formulation for $s_0(\mathbf{e})$ is usually preferred to the binomial one (both described in (2.10)), since it simplifies the notation.

Stability

In order to ensure stability [31], all trajectories in the state space must be attracted by the sliding surface s, which can be summed up as:

- all trajectories starting on the surface s must remain there (s is an invariant set);

- all trajectories starting outside of the surface s tend to it at least asymptotically.

These constraints can be translated as the following set of conditions

$$\lim_{s \to 0^+} \dot{s} < 0 \quad \text{and} \quad \lim_{s \to 0^-} \dot{s} > 0, \tag{2.14}$$

which in turn can be simplified as the much simpler condition

$$s\dot{s} < 0. \tag{2.15}$$

Zak et al. [31] also offer a different perspective on the interpretation of (2.15), as $s\dot{s}$ can be seen as the time derivative of a Lyapunov function candidate $L = \frac{1}{2}s^2$. As $\frac{\partial L}{\partial t}$ is ensured to be negative definite by (2.15), then stability is assured.

Finally, Slotine et al. [28] have a different insight on the interpretation of (2.15). Essentially, $s\dot{s}$ is the time derivative of $\frac{1}{2}s^2$, the latter being the squared "distance" to the surface, measured by s^2, which decreases along all possible trajectories. Thus, it ensures that all trajectories point towards the surface and are contained in it once it is reached.

Reaching Laws

Another reason for SMC's popularity and general use in a broad range of problems and systems is its great versatility to enforce desired dynamics on the system, in comparison to PID control and other families of controllers. This is achieved by choosing the input control in a way that imposes the desired dynamic on the time derivative of the sliding surface, $\dot{s}$. Thus, this choice has a huge impact on the reaching phase of SMC. It is also important to notice that $\dot{s}$ should be chosen in a way that ensures stability, i.e., meets the condition in (2.15).

Some examples of reaching laws are described below [19], where sgn $(.)$ is the sign function:

- Constant Rate

$$\dot{s} = -\epsilon \operatorname{sgn}(s), \quad \epsilon > 0. \tag{2.16}$$

- Exponential

$$\dot{s} = -\epsilon \operatorname{sgn}(s) - ks, \quad \epsilon > 0, \quad k > 0. \tag{2.17}$$

- Power Rate

$$\dot{s} = -k|s|^{\alpha} \operatorname{sgn}(s), \quad k > 0, \quad 1 > \alpha > 0. \tag{2.18}$$

- General

$$\dot{s} = -\epsilon \operatorname{sgn}(s) - f(s), \quad \epsilon > 0, \tag{2.19}$$

  where $f(0) = 0$ and $sf(s) > 0$ when $s \neq 0$. It is important to state that the presence of the sign function is what actually imposes the *sliding* effect, which makes it essential. This discontinuity enforces the desired constant switching between both sides of the sliding surface s.

Moreover, the value of ϵ should be carefully chosen, such that it is larger than the upper bound of the absolute value of the disturbances affecting the system, which makes sure that it can oppose these current perturbations [23].

Furthermore, the main drawback of SMC concerns the discontinuous property of these reaching laws, which, if unaddressed, can lead to chattering, high amplitude and frequency control inputs, among others. A possible solution to these problems is proposed in Section 2.2.6, making use of a low-pass filter.

Control Law Definition

Considering a generic non-linear system with state $\mathbf{x}$ and input $\mathbf{u}$, the dynamics of one of its state variables $x_i(t)$ can be described as a function of the state itself and one of the input variables as

$$\ddot{x}_i(t) = g(\mathbf{x}(t), u_i(t)). \tag{2.20}$$

Also, considering $x_{i,d}(t)$ to be the reference value for $x_i(t)$, then the error and its derivatives are

$$\begin{cases} e_i(t) = x_i(t) - x_{i,d}(t) \\ \dot{e}_i(t) = \dot{x}_i(t) - \dot{x}_{i,d}(t) \\ \ddot{e}_i(t) = \ddot{x}_i(t) - \ddot{x}_{i,d}(t) \end{cases}. \tag{2.21}$$

Now, considering a simple sliding surface s of dimension 2, using (2.8), then

$$s(t) = \dot{e}_i(t) + c_i e_i(t) \tag{2.22}$$

and its derivative is

$$\dot{s}(t) = \ddot{e}_i(t) + c_i \dot{e}_i(t). \tag{2.23}$$

Moreover, it is also chosen a generic reaching law, such that, according to (2.19),

$$\dot{s}(t) = -\epsilon \operatorname{sgn}(s(t)) - f(s(t)). \tag{2.24}$$

Finally, equating (2.23) and (2.24), substituting the error derivatives according to (2.21) and $\ddot{x}_i(t)$ for (2.20), yields

$$g(\mathbf{x}(t), u_i(t)) - \ddot{x}_{i,d}(t) + c_i(\dot{x}_i(t) - \dot{x}_{i,d}(t)) = -\epsilon \operatorname{sgn}(s(t)) - f(s(t)), \tag{2.25}$$

such that, solving for the control input $u_i(t)$, yields the final control law.

2.2.5 Quasi-Sliding Mode

When applying SMC to actual physical systems, the use of a sign function such as

$$\operatorname{sgn}(s) = \begin{cases} \dfrac{s}{|s|}, & s \neq 0 \\ 0, & s = 0 \end{cases}, \tag{2.26}$$

as pictured in figure 2.4, in the controller's reaching law can lead to severe chattering as it is a discontinuous function. Moreover, its application, even in an approximate form, would severely strain the system's actuators, eventually leading to mechanical failures, due to the high-frequency commutations.

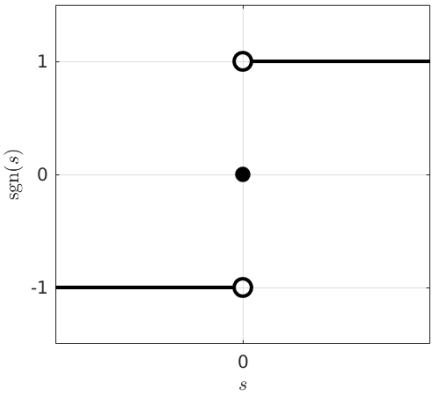


Figure 2.4: Sign function.

A proposed solution in [19] explains that substituting the sign function by a continuous one with similar characteristics could be a viable solution, such that the system still benefits from a quasi-sliding behaviour but has no chattering whatsoever.

One of these suggested solutions is a saturation function $\operatorname{sat}(.)$, as for instance

$$\operatorname{sat}(s) = \begin{cases} 1, & s > \Delta \\ ks, & |s| \leq \Delta \\ -1, & s < -\Delta \end{cases}, \quad k = \frac{1}{\Delta} > 0, \tag{2.27}$$

where Δ is called the boundary layer. As seen in figure 2.5, outside the boundary layer, the function behaves like a sign function switch control, while inside it it behaves as linear feedback control. It is crucial that Δ is small enough as to prevent the system from being constantly controlled in the linear zone, making it behave as a simple proportional controller and losing all benefits from a VSCS.

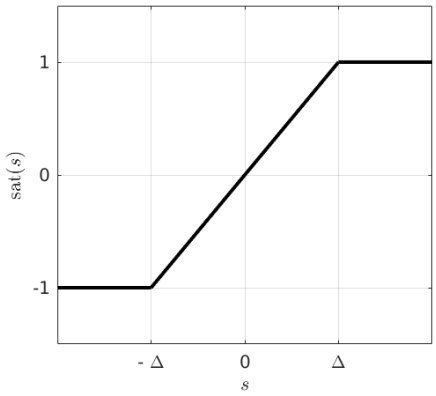


Figure 2.5: Saturation function.

Another solution suggested in [19] is to use a relay function, as described below

$$\theta(s) = \frac{s}{|s| + \delta}, \tag{2.28}$$

where δ is a very small positive constant. As pictured in figure 2.6, it can be seen that the value of δ changes the curvature of this function, such that the smaller it is, the better this function approximates a sign function like in (2.26).

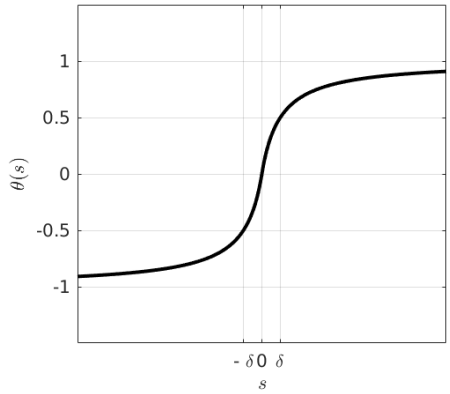


Figure 2.6: Relay function.

In fact,

$$\lim_{\delta \to 0} \theta(s) = \frac{s}{|s|}, \tag{2.29}$$

which mathematically proves what was visually inferred before. Also, this relay function has the advantage of being infinitely differentiable, in contrast to the saturation function in (2.27). Furthermore, other sigmoid-like functions would also reveal to be interesting contenders for quasi-sliding mode behaviour.

2.2.6 SMC Smoothing Solution

As stated previously, using a VSCS such as SMC has one main disadvantage, related to the discontinuity generated by the discontinuous reaching law — high frequency and amplitude chattering is transmitted

to the input controls, which is highly undesirable, as it would severely damage the vehicle's actuators, and infeasible, since actual physical actuators are rate-limited (they are not able to move instantaneously).

However, in order to preserve the intrinsic properties of SMC, namely the high-speed discontinuous switching law, quasi-sliding mode might not be a desirable solution. Thus, as stated in [23], another possible solution to the damaging chattering in the control inputs is using a low-pass first-order linear filter. As such, let the control input τ be defined as

$$\tau = \tau_0 + \tau_1, \tag{2.30}$$

such that τ_0 is the ideal control input (continuous term) and τ_1 is designed to reject perturbations (discontinuous term), i.e., the term which contains the sgn $(.)$ function. Hence, the low-pass filter is applied to the latter term, such that its output τ_{1av} is defined as

$$\mu\dot{\tau}_{1av}(t) = -\tau_{1av}(t) + \tau_1(t), \tag{2.31}$$

where $\mu \in [0, 1]$ is a constant such that $1/\mu$ is the filter's cutting frequency. Thus, τ is now finally redefined to

$$\tau = \tau_0 + \tau_{1av}. \tag{2.32}$$

2.3 Actuation on Underwater Vehicles

Actuation on underwater vehicles is related to their own actuators, which are devices capable of generating forces, which in turn can be defined by the system's design. Moreover, adding up all actuators' forces generates the overall force applied to the body of the vehicle. Also, taking into account the position of the actuators in relation to the vehicle's center of mass, then the overall torque can also be computed.

For some vehicles, the applied force and torque can be considered the system's control inputs, since it is possible to fully map the actuators in order to produce such force and torque. For other vehicles, it is not possible, as discussed in Section 2.3.1.

Typically, for underwater vehicles, the actuators are thrusters whose velocity can be controlled, which means the force produced by these thrusters can also be controlled. Additionally, it is important to notice that these thrusters are not instantaneous, having their own dynamic equations, though much faster than the whole vehicle itself.

2.3.1 Underactuated vs. Fully Actuated Underwater Vehicle

Marine crafts are generally characterised by their number of Degree-of-Freedom (DOF), which is, by definition, the set of independent displacements and rotations that completely specify the displaced position and orientation of the vehicle [7]. For instance, a ROV/AUV capable of freely moving in the three-dimensional space has a maximum of 6 DOFs, corresponding to three translational and three rotational components.

Furthermore, vehicles can be classified according to their ability to be controlled so as to achieve a certain control objective, which by itself can be defined by the number of DOFs it incorporates. On the one hand, Underactuated Vehicles are crafts whose actuators can not produce enough independent forces and moments correspondent to the craft's DOFs. On the other hand, Fully Actuated Vehicles are capable of achieving such task, which makes them adequate for control objectives involving all of the vehicle's DOFs [7].

As a matter of fact, there is also a type of fully actuated vehicles whose number of actuators exceeds the number of DOFs required to meet the control objective, which are often called Overactuated Vehicles. The extra actuators allow for a secondary objective to drive the control allocation and also to be robust to a failure in a particular actuator.

2.3.2 Allocation Methods for Underwater Vehicles

As previously mentioned, Overactuated Vehicles are crafts whose total of actuators is larger than the number of DOFs needed to achieve the control objective, which results in a non-trivial computation of the actuating forces.

When it comes to Overactuated Underwater Vehicles, following the notation used by Fossen et. al in [9], let $\boldsymbol{\tau}_i \in \mathbb{R}^n$ be the vector of forces and moments applied to the AUV produced by thruster i and $\mathbf{f}_i = [F_x \ F_y \ F_z]^T$ be the vector of forces produced by that same thruster along the three translational axis. It is possible to relate these physical quantities through (2.33),

$$\boldsymbol{\tau}_i = \begin{bmatrix} \mathbf{f}_i \\ \boldsymbol{\ell}_i \times \mathbf{f}_i \end{bmatrix} = \begin{bmatrix} F_x \\ F_y \\ F_z \\ F_z\ell_y - F_y\ell_z \\ F_x\ell_z - F_z\ell_x \\ F_y\ell_x - F_x\ell_y \end{bmatrix}_i, \tag{2.33}$$

where $\boldsymbol{\ell}_i = [\ell_x \ \ell_y \ \ell_z]_i^T$ is the vector of moment arms, i.e., the orthogonal distance between the thruster and each axis of rotation.

When dealing with more than one thruster, a more suitable notation is used. The vector of forces and moments applied to the AUV $\boldsymbol{\tau} \in \mathbb{R}^n$ can be written as

$$\boldsymbol{\tau} = T\mathbf{f}, \tag{2.34}$$

where $T \in \mathbb{R}^{n\times r}$ is the thruster configuration matrix and $\mathbf{f} \in \mathbb{R}^r$ is the vector of forces produced by all r thrusters present in the AUV. Also, $\mathbf{f}$ can be related to the actual control input $\mathbf{u} \in \mathbb{R}^r$ as

$$\mathbf{f} = K\mathbf{u}, \tag{2.35}$$

where $K \in \mathbb{R}^{r \times r}$ is called the force coefficient matrix, such that

$$K = \text{diag}\{k_1, \ldots, k_r\}. \tag{2.36}$$

Now, the problem lies with computing the control input $\mathbf{u}$ given a desired $\boldsymbol{\tau}$. For underactuated vehicles, it is physically impossible to generate arbitrary forces and torques, while in the case of fully actuated crafts there is only one solution for (2.34). For overactuated vehicles, there is an infinite number of solutions for $\boldsymbol{\tau}$, from which solutions that meet certain requirements can be searched for.

In reality, minimising power consumption and taking into consideration actuator limitations is of huge importance, so taking advantage of extra control forces in an over-actuated problem seems to be the perfect way to take into account this priority. Thus, several methods have been proposed to approach this issue, where an optimal vector $\mathbf{f}$ is computed and $\mathbf{u} = K^{-1}\mathbf{f}$:

- **Linear Quadratic Unconstrained Control Allocation**

  This approach proposes to solve the unconstrained Least-Squares (LS) Optimisation Problem in (2.37), considering $\mathbf{f}$ and $\mathbf{u}$ are unbounded.

$$\begin{aligned} \min_{\mathbf{f}} \quad & \{J = \mathbf{f}^T W \mathbf{f}\} \\ \text{subject to} \quad & \boldsymbol{\tau} - T\mathbf{f} = \mathbf{0}. \end{aligned} \tag{2.37}$$

  In here, $W \in \mathbb{R}^{r \times r}$ is a positive definite matrix (usually diagonal) that serves the purpose of balancing the relative weight between control forces. In a situation where all actuators equally contribute to the overall expense of using those control forces, matrix W is the identity matrix I.

  The explicit solution for this problem is making use of the Lagrangian in (2.38),

$$L(\mathbf{f}, \boldsymbol{\lambda}) = \mathbf{f}^T W \mathbf{f} + \boldsymbol{\lambda}^T (\boldsymbol{\tau} - T\mathbf{f}), \tag{2.38}$$

  where $\boldsymbol{\lambda} \in \mathbb{R}^r$ is the vector of Lagrange multipliers. Differentiating $L(\mathbf{f}, \boldsymbol{\lambda})$ with respect to $\mathbf{f}$, equating it to $\mathbf{0}$ and solving for $\mathbf{f}$ yields

$$\frac{\partial L}{\partial \mathbf{f}} = 2W\mathbf{f} - T^T\boldsymbol{\lambda} = \mathbf{0} \implies \mathbf{f} = \frac{1}{2} W^{-1} T^T \boldsymbol{\lambda}. \tag{2.39}$$

  Then, assuming $TW^{-1}T^T$ is non-singular, $\boldsymbol{\tau}$ can be written as

$$\boldsymbol{\tau} = T\mathbf{f} = \frac{1}{2} TW^{-1}T^T\boldsymbol{\lambda}, \tag{2.40}$$

  from which $\boldsymbol{\lambda}$ can be described as

$$\boldsymbol{\lambda} = 2(TW^{-1}T^T)^{-1}\boldsymbol{\tau}. \tag{2.41}$$

Substituting (2.41) in (2.40), gives

$$\mathbf{f} = T_w^+ \boldsymbol{\tau}, \quad T_w^+ = W^{-1} T^T (T W^{-1} T^T)^{-1}, \tag{2.42}$$

where T_w^+ is the generalised inverse. In the cases where W is the identity matrix I, then T_w^+ is simplified as the Moore-Penrose pseudo inverse T^+,

$$T^+ = T^T (T T^T)^{-1}. \tag{2.43}$$

Finally, the control input vector $\mathbf{u}$ can now be expressed from (2.35) as

$$\mathbf{u} = K^{-1} T_w^+ \boldsymbol{\tau}. \tag{2.44}$$

- **Linear Quadratic Constrained Control Allocation**

  Besides taking into consideration the importance of minimising power consumption, there are also limitations that should be considered, such as actuator saturation, wear and tear and power system overload. Also, there are a myriad of other constraints that can be taken into account, as, for instance, forbidden sectors [9].

  Hence, a constrained optimisation problem (2.45) is formulated in order to accommodate this concern. Here, as mentioned before, $\mathbf{f} = [f_1\ f_2\ \cdots\ f_r]$ is the vector of forces of the actuators and W is a positive definite matrix weighting these forces. In addition, $\mathbf{s} \in \mathbb{R}^n$ is a vector of slack variables, $\bar{f} = \max_i |f_i|$ and $\mathbf{f}_{min}$ and $\mathbf{f}_{max}$ are, respectively, the vectors with the minimum and maximum values for each one of the actuators.

$$\begin{aligned} \min_{\mathbf{f},\mathbf{s},\bar{f}} \quad & \{J = \mathbf{f}^T W \mathbf{f} + \mathbf{s}^T Q \mathbf{s} + \beta \bar{f}\} \\ \text{subject to} \quad & T\mathbf{f} = \boldsymbol{\tau} + \mathbf{s} \\ & \mathbf{f}_{min} \leq \mathbf{f} \leq \mathbf{f}_{max} \\ & -\bar{f} \leq f_1, \ldots, f_r \leq \bar{f} \end{aligned} \tag{2.45}$$

  In (2.45), the first term of J corresponds to a classical LS problem, corresponding to (2.38). The second term ensures the minimisation of the slack variables $\mathbf{s}$, in order to maintain $\boldsymbol{\tau}$ as close as possible to $T\mathbf{f}$. This is made certain if it is considered that $Q \gg W > 0$, such that the second term's weight is made higher that the first one's. The third term accommodates the minimisation of the maximum value of the actuators' forces $\bar{f}$, which is weighted against the other terms through the constant β.

  Finally, this problem can be reconstructed as a Quadratic Programming (QP) problem, for which fast and computationally efficient solving methods exist. Nowadays, QP problems (of the order considered in this work) are solvable in real time, i.e., a QP problem is solved faster than the typical frequency at which the input control is computed (20 Hz). In [13], Johansen et al. reformulates the

optimisation problem in (2.45) as the following QP problem in (2.46),

$$\begin{aligned} \min_{\mathbf{z}} \quad & \{J = \mathbf{z}^T \Phi \mathbf{z} + \mathbf{z}^T R \mathbf{p}\} \\ \text{subject to} \quad & A_1 \mathbf{z} = C_1 \mathbf{p} \\ & A_2 \mathbf{z} \leq C_2 \mathbf{p} \end{aligned} \quad , \tag{2.46}$$

where $\mathbf{z} = [\mathbf{f}^T \ \mathbf{s}^T \ \bar{f}]^T \in \mathbb{R}^{r+n+1}$, $\mathbf{p} = [\boldsymbol{\tau}^T \ \mathbf{f}_{min}^T \ \mathbf{f}_{max}^T \ \beta]^T \in \mathbb{R}^{2r+n+1}$ and

$$\begin{gathered} \Phi = \begin{bmatrix} W & 0_{r\times n} & \mathbf{0}_{r\times 1} \\ 0_{n\times r} & Q & \mathbf{0}_{n\times 1} \\ \mathbf{0}_{1\times r} & \mathbf{0}_{1\times n} & 0 \end{bmatrix}, \quad R = \begin{bmatrix} 0_{(r+n)\times(2r+n)} & \mathbf{0}_{(r+n)\times 1} \\ \mathbf{0}_{1\times(2r+n)} & 1 \end{bmatrix}, \\ A_1 = \begin{bmatrix} T & -I_{n\times n} & \mathbf{0}_{n\times 1} \end{bmatrix}, \quad C_1 = \begin{bmatrix} I_{n\times n} & 0_{n\times(2r+1)} \end{bmatrix}, \\ A_2 = \begin{bmatrix} -I_{r\times r} & 0_{r\times n} & \mathbf{0}_{r\times 1} \\ I_{r\times r} & 0_{r\times n} & \mathbf{0}_{r\times 1} \\ -I_{r\times r} & 0_{r\times n} & -\mathbf{1}_{r\times 1} \\ I_{r\times r} & 0_{r\times n} & -\mathbf{1}_{r\times 1} \end{bmatrix}, \quad C_2 = \begin{bmatrix} 0_{r\times n} & -I_{r\times r} & 0_{r\times r} & \mathbf{0}_{r\times 1} \\ 0_{r\times n} & 0_{r\times r} & I_{r\times r} & \mathbf{0}_{r\times 1} \\ 0_{r\times n} & 0_{r\times r} & 0_{r\times r} & \mathbf{0}_{r\times 1} \\ 0_{r\times n} & 0_{r\times r} & 0_{r\times r} & \mathbf{0}_{r\times 1} \end{bmatrix}. \end{gathered} \tag{2.47}$$

Finally, Johansen et al. present a solution to this optimisation problem in their work [14]. These problems can be computationally solved using Nonlinear Programming (NLP) solvers, such as the primal-dual interior point method. This thrust allocation method is implemented in Section 5.2.3.

- **Nonlinear Constrained Control Allocation (Rotatable Actuators)**

  Nowadays, some crafts are equipped with azimuth thrusters, which are especially designed to be directed along arbitrary horizontal angles. In the case of vehicles equipped with azimuth thrusters, the optimisation problem to be solved tends to be non-convex, which increases its difficulty.

  As Fossen et al. describe in [9], a more complex optimisation problem is needed to compute the optimal forces and thrusters' azimuth angles that minimise a cost function respecting the constraints. Some of the new constraints considered in this approach are the following:

  – Feasible Sectors for the Azimuth Angles - every azimuth angle is limited by minimum and maximum values;

  – Limited Turning Rate - the difference between the new sample's and the previous sample's angles is lower and upper bounded;

  – Singular Configuration Avoidance - a term is included in the cost function such that singularities are prevented, avoiding in this way divisions by zero.

Chapter 3

Vehicle Model

3.1 Reference Frames and General Notation

The adopted notation for coordinates and reference frames follows [7]. The inertial frame $\{U\}$ is composed by three axes: x_U, y_U and z_U, while the body frame $\{B\}$, attached to the marine craft, is made up of its own three axes: x_B, y_B and z_B, such that it moves along with the vehicle.

On the one hand, the inertial frame $\{U\}$ follows the North-East-Down (NED) convention: the axis z_U is directed downwards, orthogonal to the surface of the Earth, while x_U is pointed towards the North and y_U towards the East. On the other hand, the body frame $\{B\}$ is oriented such that it coincides with the vehicle's main axes of rotation. Thus, x_B is oriented as a longitudinal axes, directed towards the front of the vehicle, y_B follows the transversal axis, directed to starboard, and z_B is directed vertically, pointing downwards.

Furthermore, it should be noticed that a general marine craft experiences motion in all 6 DOF, i.e., motion in all three axes and rotation around these same axes, as seen in figure 3.1.

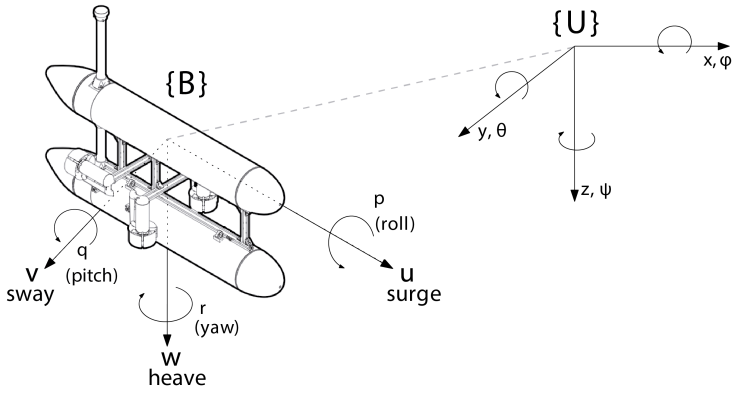


Figure 3.1: Reference Frames used, according to [7].

Hence, the adopted notation is the following:

- $\{U\}$ - body frame, usually attached to the geometric centre of mass of the vehicle;
- $\{B\}$ - inertial reference frame;

- $\boldsymbol{\eta}_1 = [x\ y\ z]^T$ - position of the origin of $\{B\}$ measured in $\{U\}$;
- $\boldsymbol{\eta}_2 = [\phi\ \theta\ \psi]^T$ - orientation of $\{B\}$ with respect to $\{U\}$;
- $\boldsymbol{\nu}_1 = [u\ v\ w]^T$ - linear velocity of the origin of $\{B\}$ with respect to $\{U\}$, expressed in $\{B\}$;
- $\boldsymbol{\nu}_2 = [p\ q\ r]^T$ - angular velocity of the origin of $\{B\}$ with respect to $\{U\}$, expressed in $\{B\}$;
- $F_{RB} = [X\ Y\ Z]^T$ - external forces measured in $\{B\}$;
- $N_{RB} = [K\ M\ N]^T$ - external torques measured in $\{B\}$;

such that $\boldsymbol{\eta} = [\boldsymbol{\eta}_1^T\ \boldsymbol{\eta}_2^T]^T$ and $\boldsymbol{\nu} = [\boldsymbol{\nu}_1^T\ \boldsymbol{\nu}_2^T]^T$.

Finally, table 3.1 summarises the notation used.

Table 3.1: Notation used (adapted from Fossen et al. [7]).

	Forces and moments	**Linear and angular velocities**	**Positions and Euler Angles**
motions in the x direction (surge)	X	u	x
motions in the y direction (sway)	Y	v	y
motions in the z direction (heave)	Z	w	z
rotation about the x axis (roll)	K	p	ϕ
rotation about the y axis (pitch)	M	q	θ
rotation about the z axis (yaw)	N	r	ψ

3.2 AUV Kinematics

The AUV kinematics geometrically describes the motion of the vehicle and the relation between the position and orientation of $\{B\}$ and the linear and angular velocity of $\{B\}$ with respect to $\{U\}$.

3.2.1 Euler Angles

The most common approach to orientation and rotation representation of a marine vehicle is making use of Euler Angles. As such, the position and orientation of $\{B\}$ is defined as $\boldsymbol{\eta}$ and the linear and angular velocity of $\{B\}$ with respect to $\{U\}$ as $\boldsymbol{\nu}$. According to the proposed notation, the kinematics equations are described by

$$\dot{\boldsymbol{\eta}} = J(\boldsymbol{\eta})\boldsymbol{\nu}, \quad J(\boldsymbol{\eta}) = \begin{bmatrix} {}^U_B R(\boldsymbol{\eta}_2) & 0_{3\times3} \\ 0_{3\times3} & T(\boldsymbol{\eta}_2) \end{bmatrix}. \tag{3.1}$$

Matrix ${}^U_B R(\boldsymbol{\eta}_2)$ is a rotation matrix, i.e., it is orthogonal (${}^U_B R(\boldsymbol{\eta}_2)^T {}^U_B R(\boldsymbol{\eta}_2) = I$) and $det({}^U_B R(\boldsymbol{\eta}_2)) = 1$. It represents the rotation from the body frame to the inertial frame and is described by successive rotations around all three axes, as follows.

$${}^U_B R(\boldsymbol{\eta}_2) = R_{z,\psi} R_{y,\theta} R_{x,\phi}, \tag{3.2}$$

where the individual rotations around each axis are defined as

$$R_{z,\psi} = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_{y,\theta} = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix}, \quad R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix}, \tag{3.3}$$

such that ${}^{U}_{B}R(\boldsymbol{\eta}_2)$ is characterised as

$${}^{U}_{B}R(\boldsymbol{\eta}_2) = \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix}, \tag{3.4}$$

given that $c(.)$ is $\cos(.)$ and $s(.)$ is $\sin(.)$.

Furthermore, $T(\boldsymbol{\eta}_2)$ is a transformation matrix defined by

$$T(\boldsymbol{\eta}_2) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix}, \tag{3.5}$$

such that $t(.)$ is $\tan(.)$. It is important to notice that this transformation is undefined for $\theta = \pm\frac{\pi}{2}$. As discussed later, this is not an issue for the case at study, since for typical manoeuvres the roll and pitch angles do not reach such high values, hence avoiding the singularity, as a 3 DOFs marine vehicle.

3.2.2 Unit Quaternions

In situations where crafts are operated such that their pitch may be equal to $\pm\frac{\pi}{2}$, the vehicle's orientation and rotation should have an alternate representation, for instance, using quaternions, since this representation avoids singularities.

A quaternion $\mathbf{q} \in \mathbb{R}^4$ is defined as a complex number with one real part σ and three imaginary parts given by $\boldsymbol{\epsilon} = [\epsilon_1 \ \epsilon_2 \ \epsilon_3]^T$, such that $\mathbf{q} = [\sigma \ \boldsymbol{\epsilon}^T]^T$ and $\mathbf{q}^T\mathbf{q} = 1$. Thus, $\boldsymbol{\eta} = [\boldsymbol{\eta}_1^T \ \mathbf{q}^T]^T$ and the kinematics equations are redefined as

$$\dot{\boldsymbol{\eta}} = J_q(\boldsymbol{\eta})\boldsymbol{\nu}, \quad J_q(\boldsymbol{\eta}) = \begin{bmatrix} {}^{U}_{B}R(\mathbf{q}) & 0_{3\times3} \\ 0_{4\times3} & T_q(\mathbf{q}) \end{bmatrix}, \tag{3.6}$$

where the rotation matrix ${}^{U}_{B}R(\mathbf{q})$ and transformation matrix $T_q(\mathbf{q})$ can be defined as

$${}^{U}_{B}R(\mathbf{q}) = \begin{bmatrix} 1-2(\epsilon_2^2+\epsilon_3^2) & 2(\epsilon_1\epsilon_2-\epsilon_3\sigma) & 2(\epsilon_1\epsilon_3+\epsilon_2\sigma) \\ 2(\epsilon_1\epsilon_2+\epsilon_3\sigma) & 1-2(\epsilon_1^2+\epsilon_3^2) & 2(\epsilon_2\epsilon_3-\epsilon_1\sigma) \\ 2(\epsilon_1\epsilon_3-\epsilon_2\sigma) & 2(\epsilon_2\epsilon_3+\epsilon_1\sigma) & 1-2(\epsilon_1^2+\epsilon_2^2) \end{bmatrix}, \quad T_q(\mathbf{q}) = \frac{1}{2}\begin{bmatrix} -\epsilon_1 & \sigma & \epsilon_3 & -\epsilon_2 \\ -\epsilon_2 & -\epsilon_3 & \sigma & \epsilon_1 \\ -\epsilon_3 & \epsilon_2 & -\epsilon_1 & \sigma \end{bmatrix}^T. \tag{3.7}$$

3.3 AUV Dynamics

As described in [7], the general dynamics of an AUV is characterised in vectorial form by

$$M_{RB}\dot{\boldsymbol{\nu}} + C_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_{RB}, \tag{3.8}$$

where M_{RB} is the rigid-body mass matrix, C_{RB} is the rigid-body Coriolis and centripetal matrix due to the rotation of the body frame $\{B\}$ about the inertial frame $\{U\}$ and

$$\boldsymbol{\tau}_{RB} = [X\ Y\ Z\ K\ M\ N]^T \tag{3.9}$$

is the vector of external forces and moments applied to the body frame $\{B\}$.

Furthermore, $\boldsymbol{\tau}_{RB}$ can be divided into the control input $\boldsymbol{\tau}$, the hydrodynamic forces and torques due to added mass and friction, drag, skin friction, etc., hydrostatic terms due to fluid density and gravity and, finally, external disturbances due to waves, sea currents and wind, for example. As such, (3.8) can be arranged as

$$\underbrace{M_{RB}\dot{\boldsymbol{\nu}} + C_{RB}(\boldsymbol{\nu})\boldsymbol{\nu}}_{\text{rigid-body forces}} + \underbrace{M_A\dot{\boldsymbol{\nu}} + C_A(\boldsymbol{\nu})\boldsymbol{\nu} + D(\boldsymbol{\nu})\boldsymbol{\nu}}_{\text{hydrodynamic forces}} + \underbrace{g(\boldsymbol{\eta}) + g_o}_{\text{hydrostatic forces}} = \boldsymbol{\tau} + \underbrace{\boldsymbol{\tau}_{\text{wind}} + \boldsymbol{\tau}_{\text{wave}}}_{\text{external disturbances}}, \tag{3.10}$$

where the rigid-body forces correspond to the body's inertia and Coriolis, centripetal and gyroscopic terms, the hydrodynamic forces relate to hydrodynamic added-masses and hydrostatic forces describe restoring forces due to gravity and fluid density.

3.4 AUV Simplified Motion

Taking into consideration the work to be developed, both the kinematics and dynamics of the AUV can be simplified. In this work's context, the vehicle is only manoeuvred at constant depth, such that movement in heave (along the z axis) is neglected. Also, rotations around the x and y axes, ϕ and θ respectively, are not considered. Thus,

$$\phi = 0, \quad \theta = 0. \tag{3.11}$$

As such, the vehicle is reduced to a 3 DOF craft, given by $\boldsymbol{\eta} = [x\ y\ \psi]^T$. Also, the linear and angular velocity of the body frame $\{B\}$ is also redefined, as $\boldsymbol{\nu} = [u\ v\ r]^T$. Thus, according to (3.1), $J(\boldsymbol{\eta})$ and the kinematics equations are simplified as

$$J(\boldsymbol{\eta}) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \begin{cases} \dot{x} = u\cos\psi - v\sin\psi \\ \dot{y} = u\sin\psi + v\cos\psi \\ \dot{\psi} = r \end{cases}. \tag{3.12}$$

Moreover, it is also important to define the expression for the vehicle's velocity in the body frame $[\dot{u}\ \dot{v}]^T$

given the vehicle's velocity and acceleration in the inertial frame, $[\dot{x}\ \dot{y}]^T$ and $[\ddot{x}\ \ddot{y}]^T$, respectively. The upper left 2×2 matrix in $J(\boldsymbol{\eta})$ from (3.12) is a rotation matrix R, i.e, $R^{-1} = R^T$ and $detR = 1$, such that

$$R(\boldsymbol{\eta}) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}, \quad \begin{bmatrix} u \\ v \end{bmatrix} = R^T \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}. \tag{3.13}$$

Hence, by computing the derivative of (3.13), yields

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \dot{R^T} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} + R^T \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix}, \tag{3.14}$$

where $\dot{R^T}$ yields

$$\dot{R^T} = -S(r)R^T, \quad S(r) = \begin{bmatrix} 0 & -r \\ r & 0 \end{bmatrix}, \tag{3.15}$$

such that $S(r)$ is a skew-symmetric matrix and $r = \dot{\psi}$ is the angular velocity about the z axis of the body frame.

Furthermore, considering that the control input only actuates through forces in the x and y axes and through the torque around the z axis, then $\boldsymbol{\tau} = [\tau_u\ \tau_v\ \tau_r]^T$. Hence, the dynamics equations are also simplified from (3.10) to

$$\begin{cases} m_u\dot{u} - m_v v r + d_u u = \tau_u \\ m_v\dot{v} + m_u u r + d_v v = \tau_v \\ m_r\dot{r} - m_{uv} u v + d_r r = \tau_r \end{cases}. \tag{3.16}$$

These coefficients can be computed by the following expressions

$$\begin{aligned} &m_u = m - X_{\dot{u}}, \quad m_v = m - Y_{\dot{v}}, \quad m_r = I_z - N_{\dot{r}}, \quad m_{uv} = m_u - m_v, \\ &d_u = -X_u - X_{|u|u}|u|, \quad d_v = -Y_v - Y_{|v|v}|v|, \quad d_r = -N_r - N_{|r|r}|r|, \end{aligned} \tag{3.17}$$

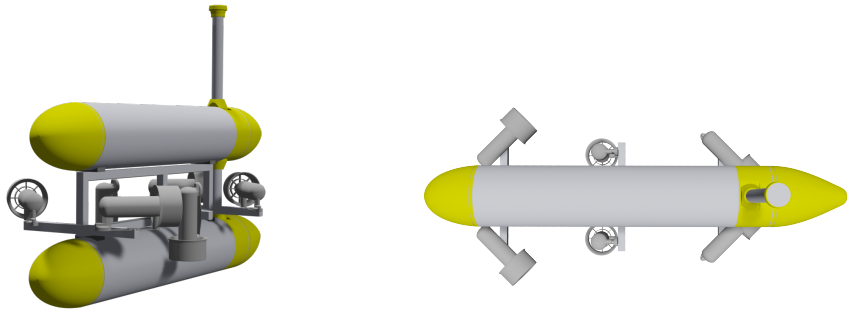
where $X_{\dot{u}}$, $Y_{\dot{v}}$ and $N_{\dot{r}}$ are hydrodynamic added masses due to the acceleration of the vehicle, since the motion of the fluid through which the craft passes is mathematically translated as an added mass. Also, X_u, Y_v and N_r are terms related to linear damping, $X_{|u|u}$ is the axial drag and $Y_{|v|v}$ and $N_{|r|r}$ are parameters associated with crossflow drag [17].

3.5 MEDUSA-class Vehicles

MEDUSA vehicles are ROV/AUVs owned by the Laboratory for Robotics and Engineering Systems (LARSyS)/Dynamical Systems and Ocean Robotics Laboratory (DSOR), from the Institute for Systems and Robotics (ISR), located at Instituto Superior Técnico (IST), in Lisbon, Portugal. These vehicles are developed by the research team at DSOR, where this work is integrated, thus making them easily accessible for the purpose of this endeavour.

The goal of this work is to adjust the previously described AUV model to this class of vehicles, so as to

simulate and test state of the art marine craft control techniques with a MEDUSA vehicle, such as the one in figure 3.2.



(a) Front view. **(b)** Top view.

Figure 3.2: Example of a MEDUSA vehicle (3D Model).

MEDUSA-class Vehicles main structure is composed of two longitudinal acrylic tubes, one above the other, such that a central aluminium frame holds them together. Also, the central frame holds the vehicle's thrusters (MEDUSA's actuators). The number and arrangement of thrusters is dependent on the type of MEDUSA vehicle.

Furthermore, MEDUSA's parameters have already been determined before by DSOR's research team, which can be examined in table 3.2.

Table 3.2: MEDUSA's parameters.

$X_{\dot{u}}$	-25 kg	X_u	-0.2 kg/s	$X_{\|u\|u}$	-19.5 kg/m
$Y_{\dot{v}}$	-2.325 kg	Y_v	-55.117 kg/s	$Y_{\|v\|v}$	-147.9 kg/m
$N_{\dot{r}}$	-8.690 kg · m^2	N_r	-4.14 kg · m/s	$N_{\|r\|r}$	-6.23 kg · m

The inertia about the z axis and the vehicle's mass are also defined, such that, respectively, $I_z = 4.14$ kg · m^2 and $m = 30$ kg.

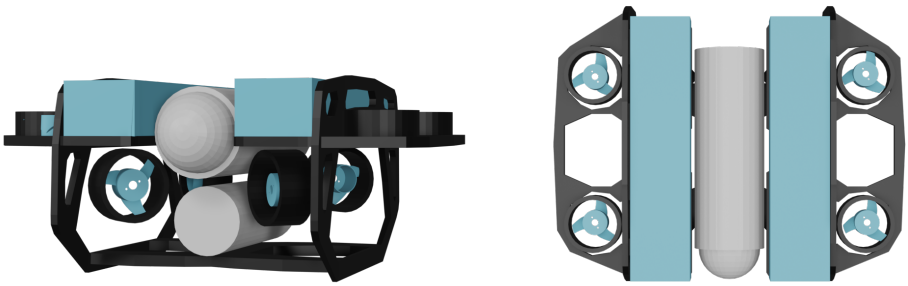
3.5.1 MEDUSA - Overactuated (3 DOF)

In recent years, the research group from DSOR developed a overactuated MEDUSA vehicle. This vehicle has four thrusters in the XY plane of the body frame for surface level manoeuvring, according to the notation previously stated. The craft also has two thrusters on each side of the craft acting on the direction of the body frame's z axis, such that the craft can also be manoeuvred vertically. As discussed later, the developed controller acts on surface level manoeuvring, while vertical movement is separately controlled using, for instance, a PID controller.

The main four thrusters referred are all oriented by 45 degrees, as seen in figure 3.2b. This configuration can be further analysed in A.1 in Appendix A.

3.6 BlueROV vehicle

The BlueROV vehicle is a ROV manufactured by BlueRobotics, a company focused on marine robotics. This vehicle comes in two different structures: a six-thruster (regular) and an eight-thruster configuration (heavy), the difference being the number of thrusters on the vertical axis (two and four thrusters, respectively). BlueROV's heavy configuration is the one used, as seen in Figure 3.3.



(a) Front view. **(b)** Top view.

Figure 3.3: BlueROV vehicle: heavy configuration (3D Model).

This vehicle has four thrusters in the vertical axis, evenly distributed as shown in Figure 3.3b, and other four thrusters in the horizontal plane, distributed in a 45 degrees configuration, similar to the ones described in the overactuated Medusa vehicle, in Section 3.5.1. Furthermore, this vehicle has two cylindrical tubes, dedicated to housing all electronic circuits and the main battery, isolating these from the outside environment. Finally, this vehicle's parameters have already been determined before by DSOR's research team, which can be consulted in Table 3.3.

Table 3.3: BlueROV's parameters.

$X_{\dot{u}}$	- 27.08 kg	X_u	-1.17 kg/s	$X_{\|u\|u}$	- 46.27 kg/m
$Y_{\dot{v}}$	- 25.95 kg	Y_v	-1.17 kg/s	$Y_{\|v\|v}$	- 46.27 kg/m
$N_{\dot{r}}$	1.00 kg · m^2	N_r	-0.5 kg · m/s	$N_{\|r\|r}$	-1.0 kg · m

The inertia about the z axis and the vehicle's mass are also measured, such that, respectively, $I_z = 0.245$ kg · m^2 and $m = 11.5$ kg.

3.7 Thrust Allocation

Recalling the relation between the vector of forces and torques in (2.34), it is possible to relate the vector of forces and torques $\boldsymbol{\tau}$ and the vector of forces produced by the craft's thrusters $\mathbf{f}$, as follows

$$\boldsymbol{\tau} = T\mathbf{f}.$$

3.7.1 6 DOF Vehicle

According to the expression for the forces and torques produced by one thruster given the force produced by that same thruster in (2.33), it is possible to sequentially determine every column of the thrust configuration matrix $T = T_{6\text{ DOF}} \in \mathbb{R}^{6\times4}$, considering that each column is determined by each actuator of the vehicle, such that

$$T_{6\text{ DOF}} = \begin{bmatrix} c(45^o) & c(45^o) & c(45^o) & c(45^o) \\ -s(45^o) & s(45^o) & -s(45^o) & s(45^o) \\ 0 & 0 & 0 & 0 \\ s(45^o)\ell_{1z} & -s(45^o)\ell_{2z} & s(45^o)\ell_{3z} & -s(45^o)\ell_{4z} \\ c(45^o)\ell_{1z} & c(45^o)\ell_{2z} & c(45^o)\ell_{3z} & c(45^o)\ell_{4z} \\ -s(45^o)\ell_{1x} - c(45^o)\ell_{1y} & s(45^o)\ell_{2x} - c(45^o)\ell_{2y} & -s(45^o)\ell_{3x} - c(45^o)\ell_{3y} & s(45^o)\ell_{4x} - c(45^o)\ell_{4y} \end{bmatrix} \tag{3.18}$$

where $s(.)$ and $c(.)$ is the sine and cosine functions, respectively, $\ell_i = [\ell_{ix}\ \ell_{iy}\ \ell_{iz}]^T$, for $i \in \{1,2,3,4\}$, are the vectors of moment arms for all four thrusters and $\boldsymbol{\tau} \in \mathbb{R}^6$.

It should be stated that only four thrusters are being considered in the thrust configuration matrix, since only horizontal movement is being considered. Also, this matrix is applicable to both the overactuated Medusa and the BlueROV vehicles, since both of them have the same thrust configuration in the horizontal plane (only the thruster locations change).

3.7.2 3 DOF Vehicle

In the case of this work, only motion in surge and sway and rotation in yaw are considered (3 DOF), alongside the four horizontal thrusters that affect them. Thus, $\boldsymbol{\tau} \in \mathbb{R}^3$ and $T = T_{3\text{ DOF}} \in \mathbb{R}^{3\times4}$ is

$$T_{3\text{ DOF}} = \begin{bmatrix} c(45^o) & c(45^o) & c(45^o) & c(45^o) \\ -s(45^o) & s(45^o) & -s(45^o) & s(45^o) \\ -s(45^o)\ell_{1x} - c(45^o)\ell_{1y} & s(45^o)\ell_{2x} - c(45^o)\ell_{2y} & -s(45^o)\ell_{3x} - c(45^o)\ell_{3y} & s(45^o)\ell_{4x} - c(45^o)\ell_{4y} \end{bmatrix}, \tag{3.19}$$

where, for the overactuated Medusa vehicle, for example,

$$\ell_{1x} = \ell_{2x} = \ell_{3x} = \ell_{4x} = 0m, \quad \ell_{1y} = \ell_{4y} = -0.20m, \quad \ell_{2y} = \ell_{3y} = 0.20m. \tag{3.20}$$

Finally, the vector of forces applied by the thrusters given the forces and torques desired for the vehicle can be computed, for example, using the Linear Quadratic Control Allocation solution stated in (2.37), with solution in (2.42) and (2.43).

Chapter 4

SMC Design and Application

In order to control surge velocity u, sway velocity v and the yaw angle ψ, three distinct SMC controllers are devised, such that the forces and torques $\boldsymbol{\tau} = [\tau_u \ \tau_v \ \tau_r]^T$ are computed. Thus, the vehicle's state is driven towards the input references u_d, v_d and ψ_d (desired surge and sway velocities and yaw angle, respectively), given the tracking error and its derivatives $\mathbf{e}$ for each DOF.

Two different approaches to this problem were taken into account: at first, a simple type of controller is designed, in order to prove some of the concepts previously stated. Then, in a second approach, the SMC controller is improved, such that some of the disadvantages of the first type of controller are mitigated.

4.1 Simple SMC Controllers

In this first approach, the sliding surface design was not the same for all three controllers, since one of them is controlling an angle (first order state variable) and the other two are controlling velocities (second order variable). As such, yaw's sliding surface s_ψ is of second degree ($n = 2$), while surge and sway velocities' s_u and s_v are of first degree ($n = 1$). In fact, a second order sliding surface for velocities would require knowledge of the states' velocities' second derivatives in order to compute the desired control, which is infeasible.

4.1.1 Surge Velocity u

For this controller, the sliding surface is designed as in (2.8) with $n = 1$, using the surge's tracking error as the sliding surface variable, such that

$$s_u = \tilde{u} = u - u_d, \tag{4.1}$$

where u_d is the reference value for u, computed from $\dot{x}_d$ and $\dot{y}_d$ using (3.13). Then, computing the derivative of s_u and applying the dynamics equation for $\dot{u}$, derived from (3.16), yields

$$\dot{s}_u = \dot{u} - \dot{u}_d = \frac{1}{m_u}\left(m_v v r - d_u u + \tau_u\right) - \dot{u}_d, \tag{4.2}$$

where $\dot{u}_d$ is computed using the equation in (3.14). Now, by choosing a constant rate reaching law (2.16) and substituting the sign function for a saturation function (2.27), follows

$$\dot{s}_u = -\epsilon_u \,\mathrm{sat}\,(s_u), \tag{4.3}$$

which ensures stability. Thus, substituting (4.3) in (4.2) and solving for τ_u yields

$$\tau_u = m_u \left[\dot{u}_d - \epsilon_u \,\mathrm{sat}\,(s_u)\right] - m_v v r + d_u u. \tag{4.4}$$

4.1.2 Sway Velocity v

Following the same thought process as before, the sliding surface s_v can be designed as

$$s_v = \tilde{v} = v - v_d, \tag{4.5}$$

such that v_d is the reference value for v, computed from $\dot{x}_d$ and $\dot{y}_d$ using (3.13). Moreover, computing the derivative of s_v and applying the dynamics equation for $\dot{v}$, derived from (3.16), yields

$$\dot{s}_v = \dot{v} - \dot{v}_d = \frac{1}{m_v}\left(-m_u u r - d_v v + \tau_v\right) - \dot{v}_d, \tag{4.6}$$

where $\dot{v}_d$ is computed using the equation in (3.14). Now, by choosing again a constant rate reaching law (2.16) and substituting the sign function for a saturation function (2.27), follows

$$\dot{s}_v = -\epsilon_v \,\mathrm{sat}\,(s_v), \tag{4.7}$$

which ensures stability. Thus, substituting (4.7) in (4.6) and solving for τ_v yields

$$\tau_v = m_v \left[\dot{v}_d - \epsilon_v \,\mathrm{sat}\,(s_v)\right] + m_u u r + d_v v. \tag{4.8}$$

4.1.3 Yaw Angle ψ

Finally, this controller is designed following (2.8) with $n = 2$, using yaw's tracking error as the sliding surface variable, yielding

$$s_\psi = c_\psi \tilde{\psi} + \dot{\tilde{\psi}} = c_\psi \left(\psi - \psi_d\right) + r - \dot{\psi}_d, \tag{4.9}$$

where ψ_d and $\dot{\psi}_d$ are the reference value for ψ and its derivative. Then, computing the derivative of s_ψ and applying the dynamics equation for $\dot{r}$, derived from (3.16), yields

$$\dot{s}_\psi = c_\psi \left(r - \dot{\psi}_d\right) + \dot{r} - \ddot{\psi}_d = c_\psi \left(r - \dot{\psi}_d\right) + \frac{1}{m_r}\left(m_{uv} u v - d_r r + \tau_r\right) - \ddot{\psi}_d, \tag{4.10}$$

where $\ddot{\psi}_d$ is the second derivative of the reference value for ψ. Now, by choosing then again a constant

rate reaching law (2.16) and substituting the sign function for a saturation function (2.27), follows

$$\dot{s}_\psi = -\epsilon_\psi \,\text{sat}\,(s_\psi), \tag{4.11}$$

which ensures stability. Thus, substituting (4.11) in (4.10) and solving for τ_r yields

$$\tau_r = -m_r \left[\epsilon_\psi \,\text{sat}\,(s_\psi) + c_\psi\,(r - \dot{\psi}_d) - \ddot{\psi}_d\right] - m_{uv} uv + d_r r. \tag{4.12}$$

4.2 Improved SMC Controllers

In this second approach, a solution to improve the control of surge and sway velocities is devised, since previously these state variables were controlled using first order sliding surfaces. As such, the variable of interest was changed, so as to, in each case, respectively, α and β are controlled instead of u and v, such that

$$\alpha = \int_0^t u(\rho)d\rho, \quad \beta = \int_0^t v(\rho)d\rho. \tag{4.13}$$

In addition, controlling the integral of the actual variable of interest has some advantages, namely the positive impact on the robustness of the controller against constant disturbances.

Moreover, instead of a constant reaching law, an exponential reaching law is used, in order to reduce the time taken to reach the control's sliding phase.

Also, the previously described smoothing solution is implemented, which prevents chattering and high frequency and amplitude control laws, eliminating the need to use a saturation function instead of the sign function in the reaching law design.

4.2.1 Surge Velocity u

As stated before, this controller uses α as variable of interest, as defined in (4.13). As such, the tracking error vector in (2.12) is expressed as

$$\mathbf{e}_\alpha = [\tilde{\alpha}\ \dot{\tilde{\alpha}}\ \ddot{\tilde{\alpha}}]^T, \quad \tilde{\alpha} = \alpha - \alpha_d, \quad \dot{\tilde{\alpha}} = u - u_d, \quad \ddot{\tilde{\alpha}} = \dot{u} - \dot{u_d} = \frac{1}{m_u}\left(m_v vr - d_u u + \tau_u\right) - \dot{u}_d, \tag{4.14}$$

where $\alpha_d = \int_0^t u_d(\rho)d\rho$. Now, from (2.9) follows

$$s_\alpha = s_{0\alpha} + z_\alpha, \tag{4.15}$$

where $s_{0\alpha}$ is computed with (2.10) such that

$$s_{0\alpha} = \left(\frac{d}{dt} + \lambda_\alpha\right)\tilde{\alpha} = \dot{\tilde{\alpha}} + \lambda_\alpha \tilde{\alpha} \tag{4.16}$$

and z_α is defined so as to ensure an exponential reaching law as follows

$$z_\alpha(t) = -s_{0\alpha}(\mathbf{e}_\alpha(0))e^{-k_c t} = -\left[u(0) - u_d(0) + \lambda_\alpha(\alpha(0) - \alpha_d(0))\right] e^{-k_c t}. \tag{4.17}$$

Furthermore, computing the derivative of s_α, $s_{0\alpha}$ and z_α yields

$$\dot{s}_\alpha = \dot{s}_{0\alpha} + \dot{z}_\alpha = \ddot{\tilde{\alpha}} + \lambda_\alpha \dot{\tilde{\alpha}} - k_c z_\alpha = \frac{1}{m_u}\left(m_v v r - d_u u + \tau_u\right) - \dot{u}_d + \lambda_\alpha(u - u_d) - k_c z_\alpha. \tag{4.18}$$

Finally, according to (2.30),

$$\tau_u = \tau_{u0} + \tau_{u1_{av}}, \tag{4.19}$$

where τ_{u0} is the ideal control input, such that

$$\tau_{u0} = -m_v v r + d_u u - m_u\left[-\dot{u}_d + \lambda_\alpha(u - u_d) + k_c s_{0\alpha}\right], \tag{4.20}$$

which implies that, according to (4.18),

$$\dot{s}_\alpha = -k_c(s_{0\alpha} + z_\alpha) + \frac{1}{m_u}\tau_{u1_{av}} = -k_c s_\alpha + \frac{1}{m_u}\tau_{u1_{av}}, \tag{4.21}$$

where $\tau_{u1_{av}}$, according to (2.31), is the result of low-pass filtering τ_{u1} with a cutting frequency $1/\mu_\alpha$ Hz and is ruled by

$$\mu_\alpha \dot{\tau}_{u1_{av}}(t) = -\tau_{u1_{av}}(t) + \tau_{u1}(t), \tag{4.22}$$

where τ_{u1} is

$$\tau_{u1} = -m_u \epsilon_u \,\mathrm{sgn}\,(s_\alpha). \tag{4.23}$$

Hence, applying (4.20) to (4.19), yields

$$\tau_u = -m_v v r + d_u u - m_u\left[-\dot{u}_d + \lambda_\alpha(u - u_d) + k_c s_{0\alpha}\right] + \tau_{u1_{av}}. \tag{4.24}$$

4.2.2 Sway Velocity v

As stated previously, this controller uses β as variable of interest, as defined in (4.13). Following the same mathematical process as for the computation of τ_u, it is possible to derive the following expression for τ_v

$$\tau_v = m_u u r + d_v v - m_v\left[-\dot{v}_d + \lambda_\beta(v - v_d) + k_c s_{0\beta}\right] + \tau_{v1_{av}}, \tag{4.25}$$

the sliding surface $s_{0\beta}$ is defined as

$$s_{0\beta} = v - v_d + \lambda_\beta(\beta - \beta_d) \tag{4.26}$$

and $\tau_{v1_{av}}$ is ruled by the following differential equation, which represents a low-pass filter,

$$\mu_\beta \dot{\tau}_{v1_{av}}(t) = -\tau_{v1_{av}}(t) + \tau_{v1}(t), \tag{4.27}$$

such that

$$\tau_{v1} = -m_v \epsilon_v \operatorname{sgn}(s_\beta). \tag{4.28}$$

Finally, the complete sliding surface s_β is defined as

$$s_\beta = s_{0\beta} - \left[v(0) - v_d(0) + \lambda_\beta(\beta(0) - \beta_d(0))\right] e^{-k_c t}. \tag{4.29}$$

4.2.3 Yaw Angle ψ

Following the same mathematical process as before, it is possible to derive the following expression for τ_r

$$\tau_r = -m_{uv} uv + d_r r - m_r \left[-\ddot{\psi}_d + \lambda_\psi(r - \dot{\psi}_d) + k_c s_{0\psi}\right] + \tau_{r1_{av}}, \tag{4.30}$$

the sliding surface $s_{0\psi}$ is defined as

$$s_{0\psi} = r - \dot{\psi}_d + \lambda_\psi(\psi - \psi_d) \tag{4.31}$$

and $\tau_{r1_{av}}$ is governed by the following differential equation, which represents a low-pass filter,

$$\mu_\psi \dot{\tau}_{r1_{av}}(t) = -\tau_{r1_{av}}(t) + \tau_{r1}(t), \tag{4.32}$$

such that

$$\tau_{r1} = -m_r \epsilon_\psi \operatorname{sgn}(s_\psi). \tag{4.33}$$

Finally, the complete sliding surface s_ψ is defined as

$$s_\psi = s_{0\psi} - \left[r(0) - \dot{\psi}_d(0) + \lambda_\psi(\psi(0) - \psi_d(0))\right] e^{-k_c t}. \tag{4.34}$$

4.3 SMC parameters' influence on controller performance

It is important to understand how theSMC parameters influence the performance of the controller itself, so that these parameters can be chosen to achieve the best possible dynamic behaviour. The impact of these parameters on controller performance is evaluated through mathematical analysis of the control expressions and trial and error testing in simulated and real environments.

4.3.1 Parameter ϵ

Parameter ϵ affects the scale of the discontinuous control input of the system, as stated in (4.23), (4.28) and (4.33). This parameter must be large enough so as to oppose external disturbances, keeping

in mind that too large values may cause the natural chattering effect to worsen, in case this control input is not filtered (as exemplified in Section 4.1).

4.3.2 Parameter Δ

As per (2.27), parameter Δ defines the boundary layer of the saturation function, which is used in Section 4.1 for the Simple SMC Controller. In this instance, Δ should be large enough to attenuate the discontinuities of the control law, while taking into consideration that a too large value will make the SMC revert to proportional control when $s \approx 0$. This corresponds to s being inside the boundary layer during the sliding phase of SMC control, erasing all benefits of using a switching control law, the main characteristic of a VSCS.

4.3.3 Parameters c and λ

Considering a sliding surface of second order, i.e., $n = 2$ (as in the majority of the controllers previously presented), (2.8) yields

$$s(\mathbf{x}) = cx_1 + x_2, \tag{4.35}$$

where the state vector $\mathbf{x} = [x_1 \ x_2]^T$ is the error and its derivative $\mathbf{e} = [e \ \dot{e}]^T$. This sliding surface definition is identical to the one described in (2.10), which in turn yields

$$s_0(\mathbf{e}) = \lambda e + \dot{e}. \tag{4.36}$$

Comparing the expressions in (4.35) and (4.36), it is clear that the parameters c and λ play the same role of defining the slope of the sliding surface.

4.3.4 Parameter k_c

In the previously mentioned improved controllers, in Section 4.2, an exponential reaching law is assigned to the derivative of the different sliding surfaces, as defined in (2.17), which accelerates the reaching phase of SMC.

The rate at which this acceleration takes place during reaching phase is balanced by k_c. A small value of this parameter leads to a slow convergence to the sliding surface, while too large values may lead to severe chattering and constant overshooting.

Chapter 5

Simulation

5.1 *MATLAB*/*Simulink* Implementation

In this implementation, both simple and improved solutions for the SMC controller are tested in order to freely manoeuvre an overactuated MEDUSA vehicle in 3 DOF, as the one previously described.

5.1.1 Software Simulation

This simulated environment is implemented using *MATLAB* and *Simulink* software. A block diagram for the implemented system can be visualised in figure 5.1.

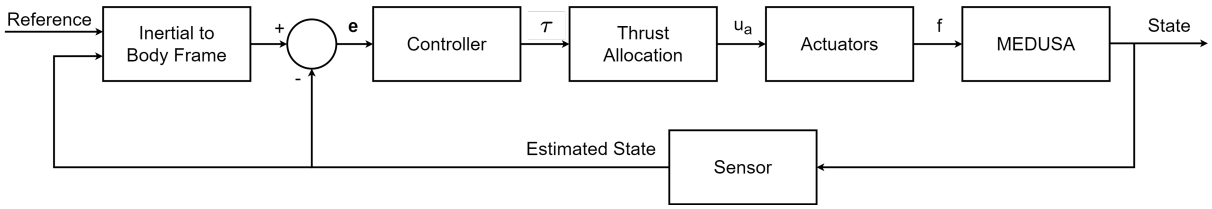


Figure 5.1: Simulated System block diagram.

In this figure, the Reference signal is a vector with the desired yaw angle ψ and velocities along the inertial frame's x and y axes for the entire duration of the simulation. Also, **e** is the vector of the tracking error and its derivatives up to the $(n-1)^{th}$ derivative, as explained previously. Moreover, $\boldsymbol{\tau}$ is a vector with the controller-generated forces and moments to be applied to the vehicle and $\mathbf{u}_a$ is the actuators' input. Finally, **f** is the vector of forces produced by the craft's actuators and the Estimated State is the vehicle's state as estimated by its sensors.

5.1.2 Inertial to Body Frame Transformation

The system's input consists of time parameterised values for ψ, $\dot{x}$ and $\dot{y}$, which are the reference that the vehicle must follow. Nevertheless, as discussed later, the controller also needs some time derivatives of these reference values, which means that $\dot{\psi}$, $\ddot{\psi}$, $\ddot{x}$ and $\ddot{y}$ are available as system inputs.

Hence, the Inertial to Body Frame block serves the purpose of computing u and v from (3.13) and $\dot{u}$ and $\dot{v}$ using (3.15) in (3.14).

5.1.3 Thrust Allocation and Actuator Dynamics

The Thrust Allocation block computes the control input vector $\mathbf{u}_a$ to be applied to the actuators, given the desired forces and torques $\boldsymbol{\tau}$ to be applied to the vehicle itself, following the solution for a Linear Quadratic Unconstrained Control Allocation problem. This relation is given by substituting T_w^+ for T^+, from (2.43), in (2.44), such that

$$\mathbf{u_a} = K^{-1}T^+\boldsymbol{\tau}.$$

The Actuator block has the purpose of simulating the craft's thrusters dynamic, as seen in figure 5.2. At first, it computes the forces $\mathbf{f}_i$ produced by the actuators given the control input $\mathbf{u}_a$, as stated by (2.35), and then filters them with a first-order filter, outputting the vector $\mathbf{f}$. This filter has a time constant $\Delta_a = 0.1s$.

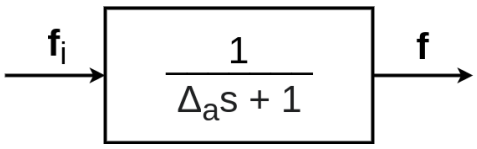


Figure 5.2: Actuator Dynamics block diagram.

5.1.4 MEDUSA Model

The MEDUSA block simply simulates MEDUSA vehicle's behaviour, mathematically implementing its kinematics and dynamics equations, making use of the expressions in (3.12) and (3.16), respectively. In this block, MEDUSA's real parameters from table 3.2 are used.

5.1.5 State Sensor

The Sensor block simulates the noisy perception of the vehicle's state. This is achieved by adding white noise to the state vector outputted by the MEDUSA model block. In this case, it is created by a White Noise block with noise power of 10^{-6}, which is reasonable for simulation purposes.

5.1.6 Results and Discussion

In this section, simulation results are discussed and different approaches are compared. The reference values used were the following:

$$\begin{cases} \dot{x}_d(t) = 0.2 \text{ [m/s]} \\ \dot{y}_d(t) = 0.2 \text{ [m/s]} \\ \psi_d(t) = 1.5\cos(0.4t) \text{ [rad]} \end{cases} \tag{5.1}$$

In order to properly simulate an actual environment as much as possible, some uncertainties were implemented in the system, modelling inaccurate characterisation of the system's parameters. Firstly, the

model's parameters used in the controller have a maximum of 25% error in comparison to the real values used in the simulation model. Secondly, white noise was added to the state's sensor.

The parameters used for the SMC controllers are shown in Figures 5.1 and 5.2.

Table 5.1: Simple SMC controllers - parameters for the *MATLAB* implementation.

	Yaw - ψ	Surge - u	Sway - v
c	0.5	-	-
ϵ	0.3	3	3
Δ	0.01	0.01	0.01

Table 5.2: Improved SMC controllers - parameters for the *MATLAB* implementation.

	Yaw - ψ	Surge - u	Sway - v
λ	0.5	0.5	0.5
ϵ	0.3	0.02	0.02
k_c	15	15	15
μ	0.8 s	0.2 s	0.2 s

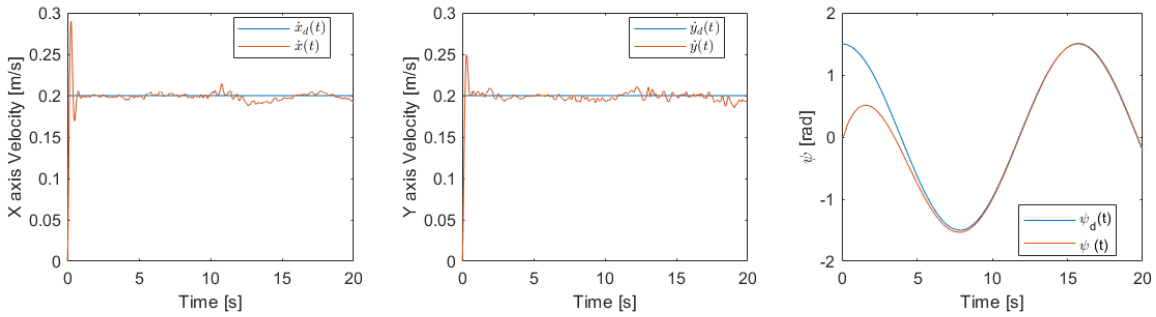
Improved SMC vs. Simple SMC

Here, the improved solution with SMC is compared to the simple SMC solution. First, the controllers' ability to follow the references is compared, as shown in figure 5.3.

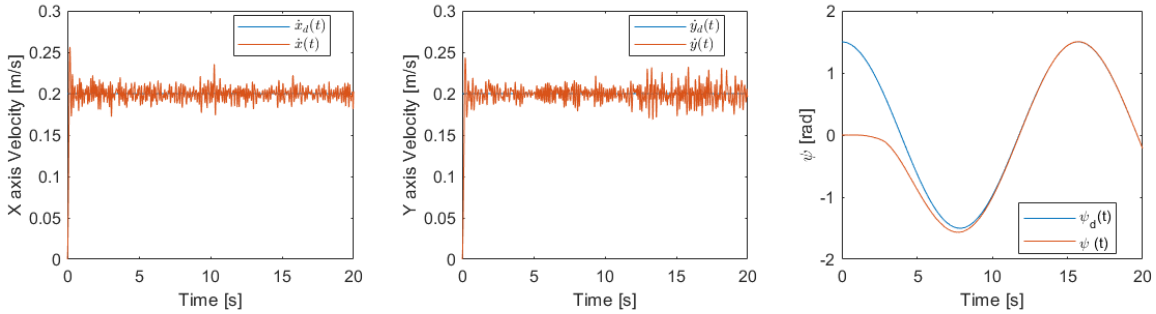
In this figure, it should be noticed that the vehicle's velocities $\dot{x}$ and $\dot{y}$ follow its reference with little deviation when using the improved SMC solution (5.3a), while with the simple solution the vehicle's velocities are much more noisy (5.3b), which is a clear influence of using the suggested smoothing solution for the improved SMC. It is also clear that the vehicle's yaw angle reaches the reference sooner in the improved solution, which results from using an exponential reaching law instead of a constant one.

Finally, the forces and torques $\boldsymbol{\tau}$ computed by the controller should also be compared, since these directly influence the control inputs of the actuators, as follows in figure 5.4.

It is clear that the obtained solution with the improved SMC solution is better (5.4a), due to the reduced amplitude of the high frequency components in the moments and torques signals, which directly influences the control inputs for the actuators. A smoothed solution such as this one is absolutely crucial in order to prevent extensive wear and eventually damaging the vehicle's actuators.

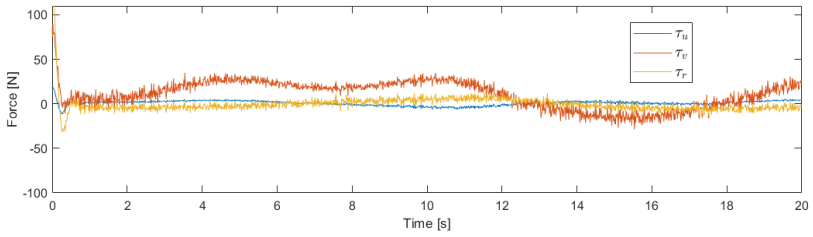


(a) Improved SMC solution

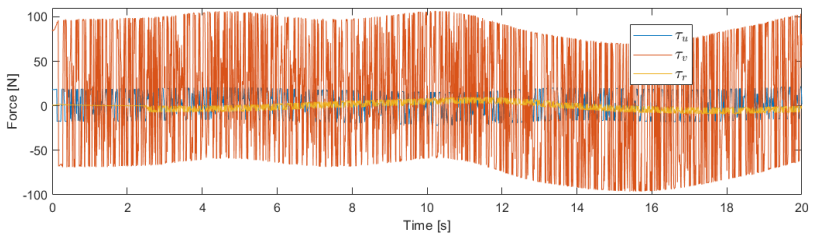


(b) Simple SMC solution

Figure 5.3: Comparison between improved and simple SMC solutions - reference tracking.



(a) Improved SMC solution



(b) Simple SMC solution

Figure 5.4: Comparison between improved and simple SMC solutions - controller generated forces and torques.

5.2 *C++*/ROS Implementation

The improved version of the SMC controllers developed is implemented using the *C++* language, within the Robot Operating System (ROS) framework. This allows to test the aforementioned controllers in a simulated environment much closer to reality with the BlueROV vehicle, as discussed later, and also in the actual vehicle, using the same code.

5.2.1 Farol Stack

The Farol stack is a set of ROS packages in *C++* and *Python* created and developed by the research team of DSOR, at ISR/IST, and by companies/developers of the robotics community. It serves the purpose of simulating and operating multiple robotic marine vehicles in an environment similar to the real world, where physical models of the vehicles and bodies of water are better simulated. This stack is public and available to the whole scientific community in `https://github.com/dsor-isr/farol`.

Moreover, the usage of ROS in the stack makes it possible to use the same code both for simulation and real experiments, as it is designed to communicate with low-level devices such as vehicle sensors and actuators. In essence, the ROS framework is based on a graph-like structure, where nodes communicate with each other through message subscribers and publishers, services and more.

Furthermore, the stack follows an inner-outer loop structure, which isolates the controller's kinematic and dynamic models, simplifying the development of the software, as follows in Figure 5.5.

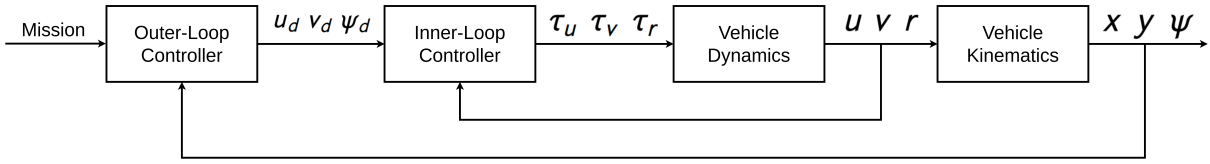
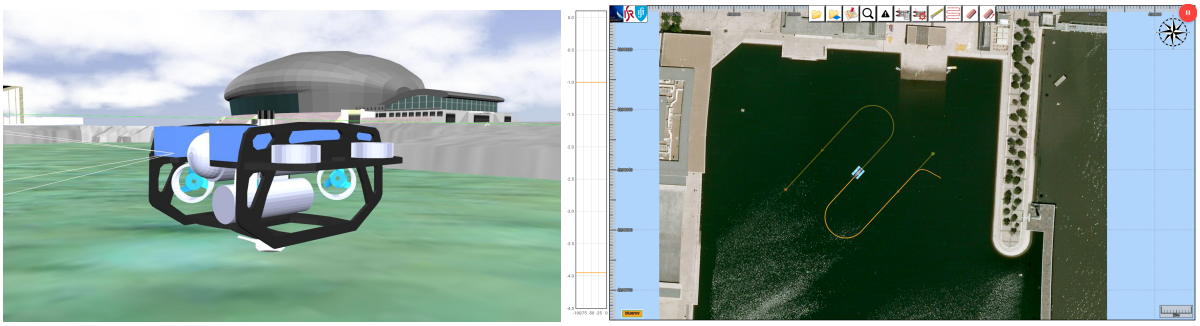


Figure 5.5: Inner-outer loop structure, considering Simplified Motion as in Section 3.4.

The inner-loop block computes the forces and torques required for the vehicle to follow given certain references, such as surge, sway and yaw. Also, the outer-loop block generates said references, given a certain mission to be accomplished.

Finally, the simulated environment where robotic marine vehicles are operated makes use of Gazebo, a 3D simulation software, as seen in Figure 5.6a. Also, a graphical console (Figure 5.6b) is used in order to upload missions to the vehicles in the simulation, which facilitates testing and development.



(a) BlueROV 3D model in Gazebo Simulator. **(b)** Lawnmower mission set through the console.

Figure 5.6: Graphical interfaces integrated with the Farol stack.

5.2.2 SMC

In order to test the advantages of SMC in an environment similar to the real world, the improved SMC controllers presented in Section 4.2 are implemented in the Farol stack as inner-loop controllers, such that this new type of controller is available to all the users in the scientific community in addition to the existing PID controllers.

Multiple Inner-Loop Controllers

Firstly, the SMC controllers are implemented such that they can be used in conformity with the already existent PID controllers, i.e., using the same interfaces and structure. This makes it possible to keep one type of controllers in charge of some DOFs, while other type of controllers can command some other DOFs. For instance, surge, sway and yaw may be controlled by SMC controllers, for horizontal movement, while the PID controllers could be in charge of maintaining the pitch and roll values near 0 degrees and the depth at a constant value. This is particularly important since, even though there are no SMC controllers developed for DOFs besides surge, sway and yaw, the vehicle's depth should be such that the craft is far enough from the sea bed and the remaining DOFs should not be left uncontrolled during vehicle operation. The latter is due to the fact that unmodelled dynamics might produce forces or torques in these DOFs, which would lead to unwanted behaviour.

Yaw Error

Secondly, amidst implementation of the yaw SMC controller, the yaw error computation must be reformulated, since in the Farol stack the yaw angle is always contained in a specific interval, i.e., $\psi \in [0\ 360]^\circ$. To deal with this discontinuity, the yaw error $\tilde{\psi}$ in degrees is redefined as follows

$$\tilde{\psi} = \begin{cases} \tilde{\psi}_0, & |\tilde{\psi}_0| < |\tilde{\psi}_0 - \operatorname{sgn}\tilde{\psi}_0 \times 360| \\ \tilde{\psi}_0 - \operatorname{sgn}\tilde{\psi}_0 \times 360, & |\tilde{\psi}_0| > |\tilde{\psi}_0 - \operatorname{sgn}\tilde{\psi}_0 \times 360| \end{cases}, \quad \tilde{\psi}_0 = \psi - \psi_d, \tag{5.2}$$

where ψ_d is the reference value for ψ.

Discrete Integral and Derivate in Real Time

Moreover, in order to implement the surge, sway and yaw controllers, there is a necessity to compute discrete derivatives and integrals in real time. As such, for a general time dependent discrete variable $a[t(i)]$, where $t(i)$ is the time in seconds and $i \in \mathbb{R}_0^+$, its integral is defined using the trapezoid law, such that

$$\int_{t=t(n)}^{t=t(m)} a[t] \approx \frac{1}{2}\big[a[t(n)] + a[t(m)]\big][t(m) - t(n)] \tag{5.3}$$

and its derivative is defined as follows

$$\dot{a}[t(m)] \approx \frac{a[t(m)] - a[t(m-1)]}{t(m) - t(m-1)}, \quad m > 0. \tag{5.4}$$

Integrator Anti-Windup

Furthermore, when using integral control a certain state variable, a phenomenon known as integral windup might happen; this term designation refers to when the integral term accumulates a significant error before this error reaches zero, i.e., the reference value is reached by the state variable, which leads to severe overshooting. Several anti-windup techniques have been developed, from which two are presented:

- **Integral Zeroing**: when the error crosses zero, the integral of the error is reset to zero.
- **Integral Saturation**: the integral of the error is saturated at a fixed value, preventing large accumulation of the error.

At first glance, Integral Zeroing seems like the perfect solution to prevent anti-windup, as it directly resets the integral term to zero when the error crosses zero. However, due to its discontinuous nature, doing this introduces unexpected dynamics to the system, which are not considered by the stability proof of SMC.

In opposition, Integral Saturation offers a better solution to the windup problem, as it reduces overshooting without introducing discontinuities through saturation of the integral term.

5.2.3 Thrust Allocation with Optimisation Methods

The Linear Quadratic Unconstrained Control Allocation method (mentioned in Section 2.3.2) is already implemented in the Farol stack by the research team at DSOR, providing a standard thrust allocation method, used in all subsequent simulations, unless stated otherwise.

Nonetheless, the Linear Quadratic Constrained Control Allocation method, presented in the aforementioned Section, is also implemented in the Farol stack, making use of CasADi, an open-source tool for nonlinear optimisation and algorithmic differentiation.

5.2.4 Results and Discussion - SMC and PID

Several tests in simulation were conducted in order to analyse SMC performance in comparison to PID control. Parameter tuning of the PID controllers used was performed by the research team at DSOR-ISR, which have been used for all vehicle control tests and have proved to be reliable. The parameters used for the SMC controllers are shown in Figure 5.3.

Table 5.3: SMC's parameters for the *C*++/ROS implementation.

	Yaw - ψ	Surge - u	Sway - v
λ	1.2	0.3	0.3
ϵ	0.1	0.02	0.02
k_c	1.8	0.8	0.8
μ	1 s	1 s	1 s

Constant Yaw Reference

In order to compare yaw control between both types of controllers, a yaw reference signal $\psi_d(t)$ was devised, as seen in Figure 5.7, where $\psi_{\text{PID}}(t)$ and $\psi_{\text{SM}}(t)$ are the vehicle's yaw when using PID control and SMC, respectively.

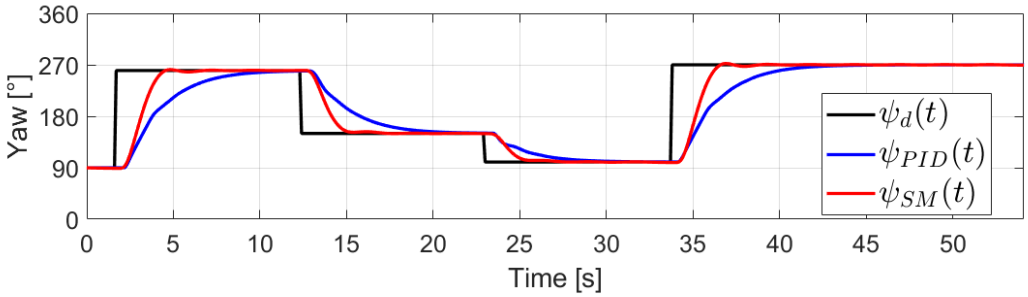


Figure 5.7: Yaw references - comparison between SMC and PID.

Through analysis of Figure 5.7, it is clear that both controllers succeed in driving the vehicle's yaw to the reference. However, SMC achieves the objective in less time with little to no overshooting, accomplishing better performance.

Also, both controllers evidence a delay between the change of reference and starting to act on the vehicle in order to follow the new reference. This is due to multiple causes, namely ROS process delay associated with stopping reference publishing and starting it again and simulated thruster delay. During this delay, the controllers stop working since no reference is being received (this behaviour is better demonstrated later), although this behaviour is not clear in this case, since, after reaching the yaw reference, the vehicle's thrusters operate at really low RPM values (similar to having the thrusters stopped when the controllers are idle). If handled correctly and with more time to study it, this unwanted behaviour could potentially be avoided.

Constant Surge Reference

Now, a surge reference signal u_d was created, as seen in Figure 5.8, where $u_{\text{PID}}(t)$ and $u_{\text{SM}}(t)$ are the vehicle's surge velocity when using PID control and SMC, respectively.

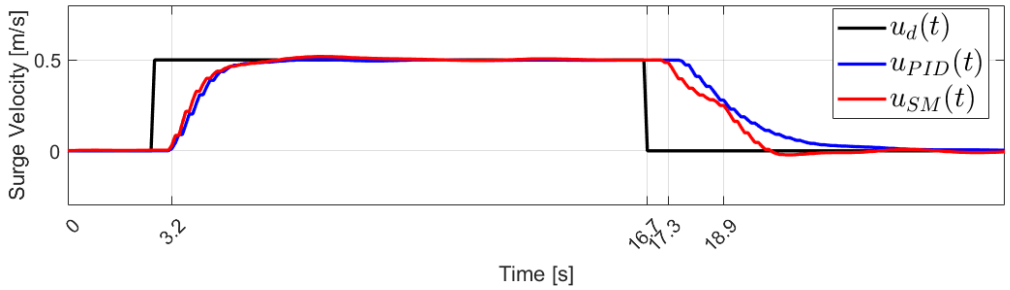


Figure 5.8: Surge references - comparison between SMC and PID.

After thorough analysis of the figure presented, it is clear that both controllers achieve the desired reference for surge velocity. Also, SMC and PID control show similar behaviour when the vehicle is accelerating from 0 m/s to 0.5 m/s, reaching this objective with little overshoot and in a similar time interval.

In this case, it is easier to visualise the idleness of the controller. At $t \approx 16.7$ s, the reference is changed, followed by a delay until the controller acknowledges the loss of a reference, at $t \approx 17.3$ s; during this period of time, the vehicle's surge velocity remains controlled at 0.5 m/s. From this instant of time until $t \approx 18.9$ s, the controller remains idle, not computing the desired force. Only after $t \approx 18.9$ s does the controller acknowledge the change in reference and resumes computing the desired force, which results in a noticeable change in the thrusters' RPM command, as seen in Figure 5.9.

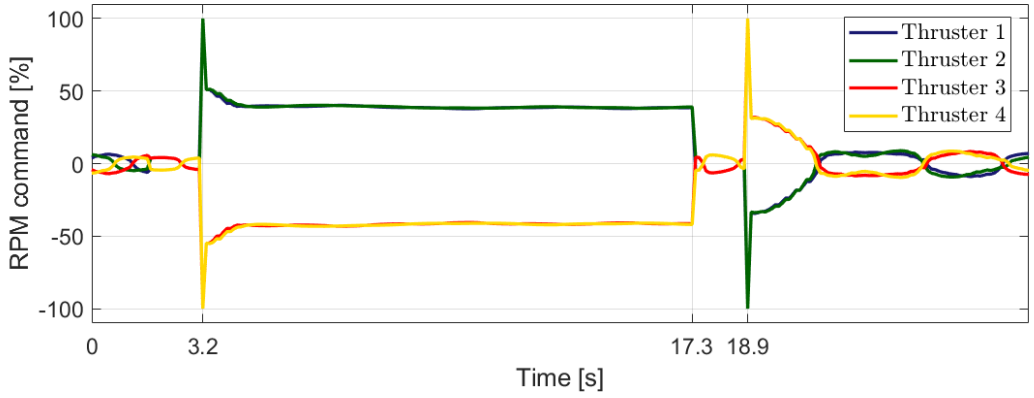


Figure 5.9: Surge references - thrusters' RPM command for SMC's case.

It is important to notice that, for the sake of performing the surge references test, the vehicle's yaw is constantly controlled at 0 degrees. As seen in Figure 5.9, this explains why, from $t \approx 17.3$ s to $t \approx 18.9$ s, the thrusters' RPM command has non-zero values. Furthermore, although in the time intervals between $t \approx 0$ s and $t \approx 3.2$ s and after $t \approx 18.9$ s the situation is the same (controlling surge velocity to 0 m/s and controlling yaw to be constant), in the latter time interval the surge velocity is still stabilising, resulting in higher amplitude oscillations in comparison.

Also, only 4 of the 8 thrusters in the BlueROV vehicle are displayed in this figure, since only these are responsible for horizontal movement.

Constant Sway Reference

Moreover, in order to compare the sway velocity controllers, a sway reference signal $v_d(t)$ was created, as seen in Figure 5.10, where $v_{\text{PID}}(t)$ and $v_{\text{SM}}(t)$ are the vehicle's sway velocity when using PID control and SMC, respectively.

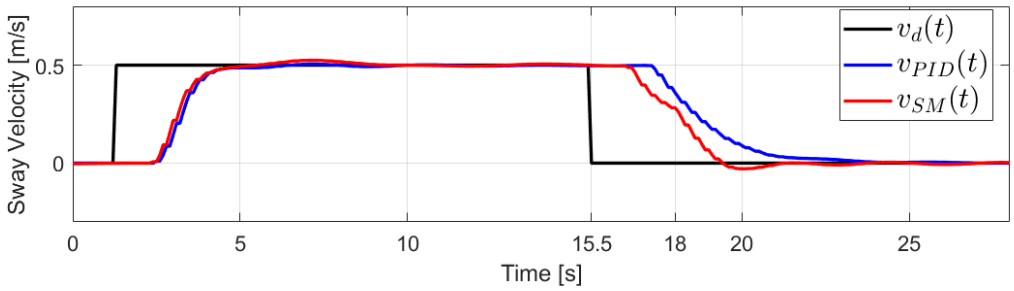


Figure 5.10: Sway references - comparison between SMC and PID.

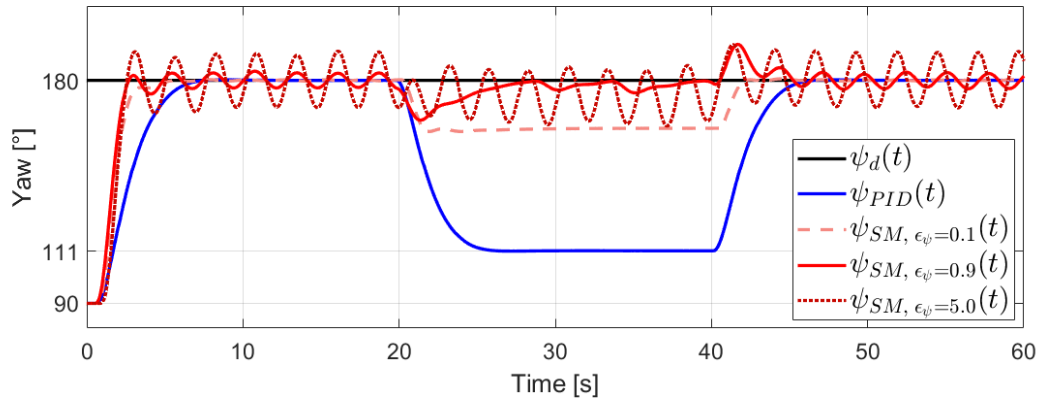
As expected, the sway controllers have similar performance to the surge controllers, since the dynamics of the BlueROV vehicle (specifically the linear and quadratic damping) are similar in the surge and sway axis, as seen in Table 3.3.

External Disturbance

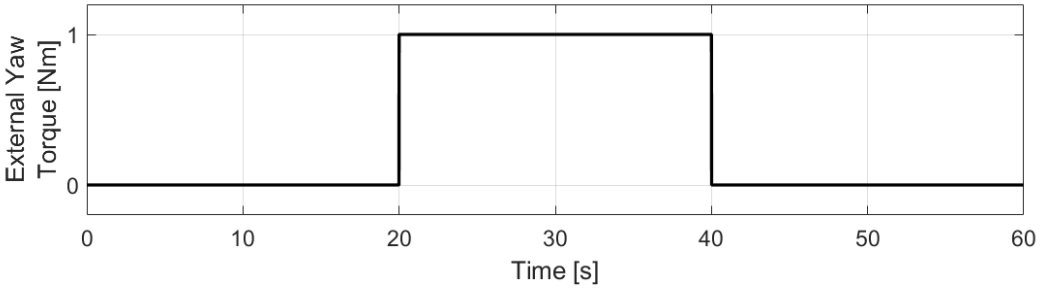
Furthermore, it is also important to assess both controllers' performance in the presence of external disturbances. Thus, as seen in Figure 5.11, the yaw controllers' ability to drive the vehicle's heading to a certain reference is tested, by applying an external torque of 1 Nm in the Z axis for 20 s, as seen in Figure 5.11b.

Through analysis of Figure 5.11a, it is clear that the PID controller is unable to maintain the vehicle's yaw near the desired reference while it is affected by the external disturbance, i.e., between $t = 20$ s and $t = 40$ s, leading to a constant error of 69 degrees. The integral term of the controller is not able to drive the vehicle's yaw to the reference due to the implemented saturation on this term.

When it comes to SMC's performance in this case, it is highly affected by its parameter ϵ. As explained in section 4.3.1, the parameter ϵ affects the scale of the discontinuous control input of the system, which is designed based on a trade-off between the ability to oppose external disturbances and SMC's natural chattering effect, since both increase with the parameter ϵ. In essence, a large value of ϵ leads to large chattering around the reference value, but guarantees good performance with respect to opposing external disturbances, while a small value of ϵ leads to small chattering but little to no ability to oppose external disturbances.



(a) Controlled yaw.



(b) External torque in the Z axis.

Figure 5.11: Yaw Controller in the presence of a constant disturbance - comparison between SMC for different values of ϵ_ψ and PID.

In the aforementioned figure, SMC's performance is illustrated with 3 different values for ϵ_ψ:

- $\epsilon_\psi = 0.1$

  When the vehicle is undisturbed, i.e., between $t = 0$ s and $t = 20$ s and between $t = 40$ s and $t = 60$ s, the normal chattering effect leads to a sinusoidal wave with ≈ 0.2 degrees of amplitude. When disturbed, the controller is unable to drive the vehicle's yaw to its reference in time, stabilising its error at ≈ 20 degrees.

- $\epsilon_\psi = 0.9$

  When the vehicle is undisturbed, the normal chattering effect leads to a sinusoidal wave with ≈ 2.8 degrees of amplitude. When disturbed, the controller is able to drive the vehicle's yaw to its reference in time, oscillating periodically near it.

- $\epsilon_\psi = 5.0$

  When the vehicle is undisturbed, the normal chattering effect leads to a sinusoidal wave with ≈ 10.9 degrees of amplitude. When disturbed, the controller is able to drive the vehicle's yaw to its reference in time, ending up oscillating sinusoidally near it, also with an amplitude of ≈ 10.9 degrees.

In order to summarise and compare all three cases, Table 5.4 is constructed, gathering the maximum

absolute error of ψ_{SM}, max $|\tilde{\psi}_{SM}|$, and the absolute error of the average of ψ_{SM}, $|\tilde{\bar{\psi}}_{SM}|$, both when the vehicle is undisturbed and disturbed.

Table 5.4: Performance of SMC for different values of ϵ_ψ, with and without a constant disturbance in yaw torque (values in degrees).

	Vehicle with no disturbance		Vehicle with disturbance	
	max $\|\tilde{\psi}_{SM}\|$	$\|\tilde{\bar{\psi}}_{SM}\|$	max $\|\tilde{\psi}_{SM}\|$	$\|\tilde{\bar{\psi}}_{SM}\|$
$\epsilon_\psi = 0.1$	0.2	≈ 0	20.7	19.3
$\epsilon_\psi = 0.9$	2.8	≈ 0	16.1	4.1
$\epsilon_\psi = 5.0$	10.9	≈ 0	18.0	3.9

After analysing all three cases, on the one hand, it is clear that $\psi_{SM,\epsilon_\psi=0.1}$ has the lowest max $|\tilde{\psi}_{SM}|$ when undisturbed, but it does not drive the vehicle to the reference while disturbed. On the other hand, $\psi_{SM,\epsilon_\psi=0.9}$ seems to be the best compromise between the vehicle's behaviour while affected and not affected by an external disturbance.

Lawnmower Mission

Additionally, the performance of both controllers is also tested through the execution of path following manoeuvres. Using the Aguiar algorithm, defined in (2.2), a lawnmower path $p(\gamma)$ is chosen to illustrate the efficiency of both controllers in this type of situation, where $p_{PID}(t)$ and $p_{SM}(t)$ are the vehicle's position when using PID control and SMC, respectively. Also, given the fact that the implemented Aguiar algorithm does not produce sway references for the inner-loop controller, the sway controllers are explicitly given a sway reference of 0 m/s during the whole manoeuvre, in order to prevent movement in the body's y axis.

Firstly, this lawnmower path is executed with a fixed surge velocity reference of 0.3 m/s, as seen in Figure 5.12.

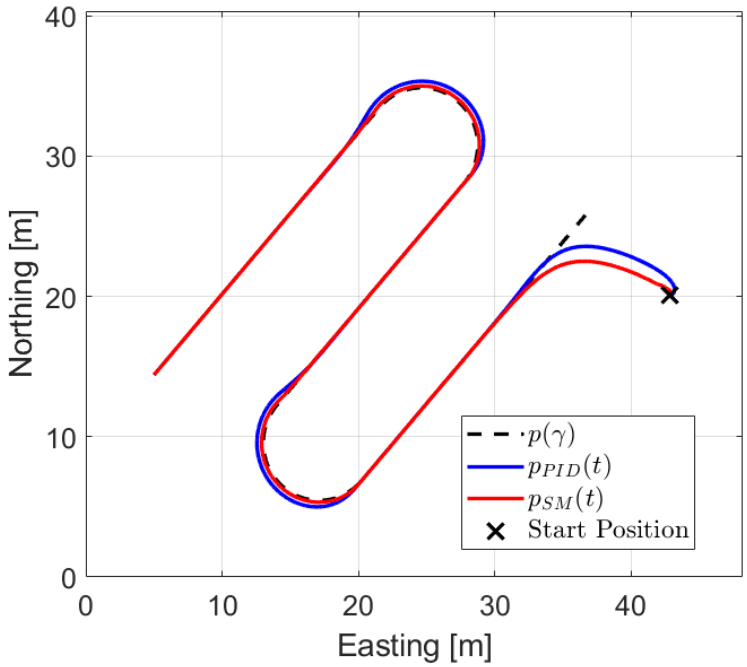
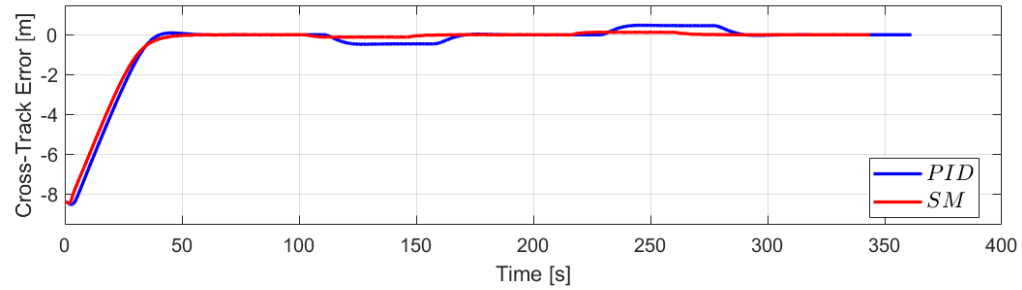


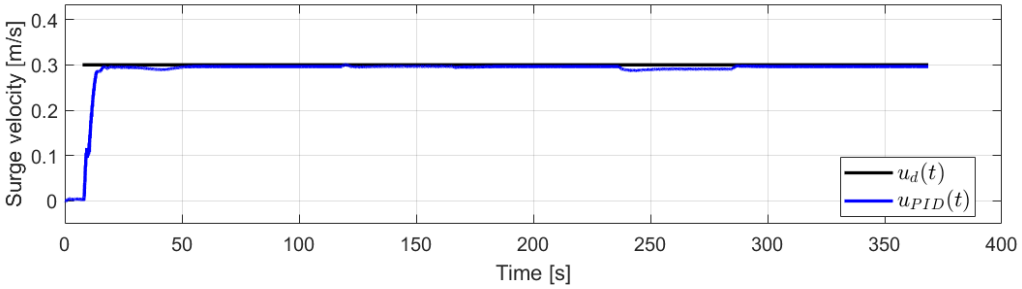
Figure 5.12: Lawnmower Path Following with u_d = 0.3 m/s - comparison between SMC and PID.

After analysing the presented figure, it is possible to see that both controllers have generally good performance. On the one hand, both controllers drive the vehicle with no overshoot when initially approximating to the desired path and, after this initial phase, the vehicle follows the desired path with low cross-track error (orthogonal distance between the vehicle and the desired path) until it reaches the end of it. On the other hand, SMC provides a faster initial approximation to the path, since its yaw controller drives the vehicle's heading to the desired reference faster than its PID counterpart, leading to finishing the path in a shorter period of time. Also, during the semicircular sections of $p(\gamma)$, it is clear that SMC drives the vehicle through the desired path with smaller cross-track error, in contrast to the PID controller.

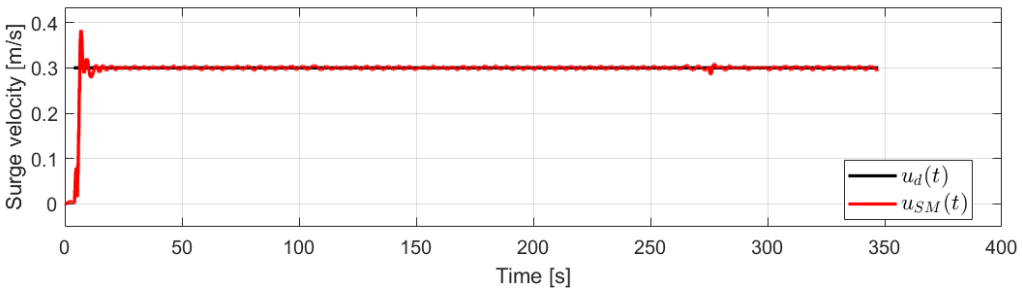
In order to better demonstrate this performance, the cross-track error and the surge velocity for both controllers throughout the path following are presented in Figure 5.13.



(a) Cross-track error.



(b) Surge velocity - PID controller.



(c) Surge velocity - SMC.

Figure 5.13: Lawnmower Path Following with u_d = 0.3 m/s: Cross-track error and surge velocity - comparison between SMC and PID.

Analysing Figure 5.13a, the maximum absolute cross-track error after the initial approximation to the desired path for SMC is 0.1 m, while for the PID controller is 0.5 m, proving the former's better performance. Also, looking at Figures 5.13b and 5.13c, both controllers succeed in driving the vehicle's surge velocity to the desired reference, having maximum steady-state absolute errors of 0.30 m/s and 0.32 m/s for SMC and PID control, respectively.

Secondly, these controllers' performance is also tested through the execution of the lawnmower manoeuvre with a fixed surge velocity reference of 1.0 m/s, a higher value than previously shown, as seen in Figure 5.14. This test is aimed to evaluate how differently does the surge velocity of the vehicle impact the performance of both controllers.

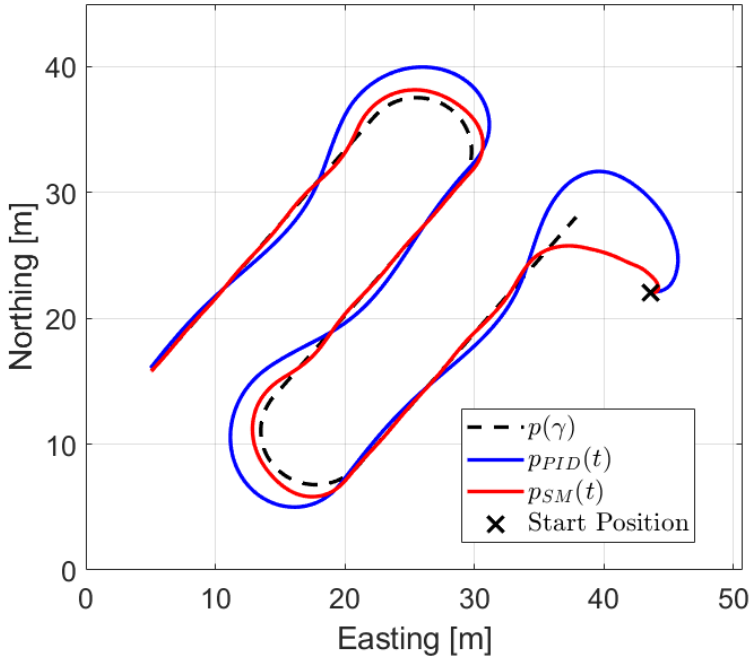
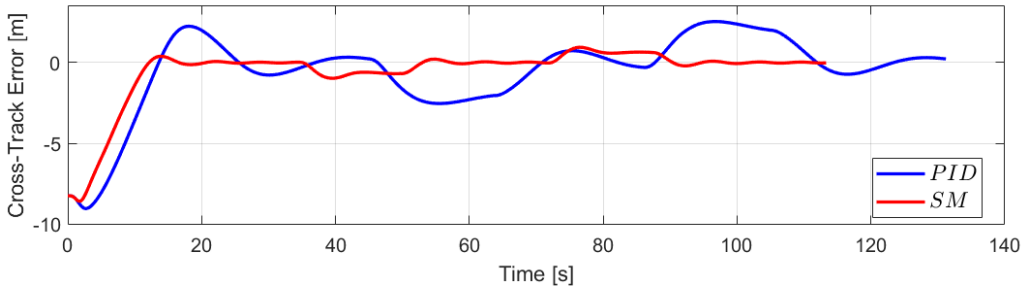


Figure 5.14: Lawnmower Path Following with u_d = 1.0 m/s - comparison between SMC and PID.

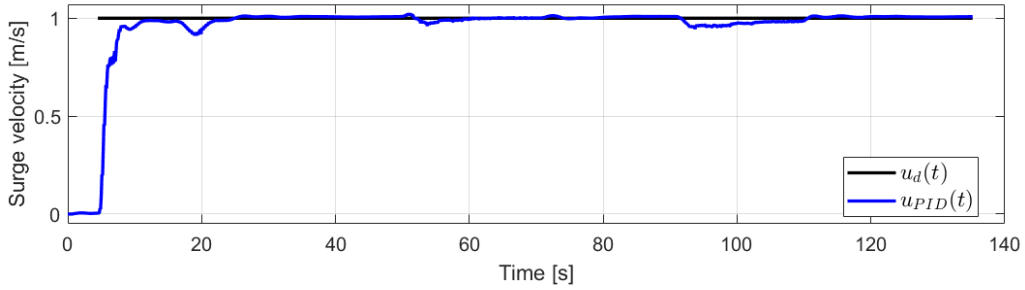
After analysing the presented figure, it can be seen that a higher surge velocity had a clear impact on the controllers' performance, while SMC was less affected than the PID controller. This proves SMC's effectiveness in counteracting the vehicle's dynamics effects on its movement, i.e., the way in which the vehicle's movement in one DOF affects the other DOFs.

It is observable that the PID controller drives the vehicle to the desired path with severe overshooting, evident in the initial approximation and after the semicircular parts of the path, while SMC executes this objective with little overshooting. In addition, during these semicircular sections of the path, it is perceivable that SMC drives the vehicle through the desired path with smaller cross-track error, in contrast to the PID controller.

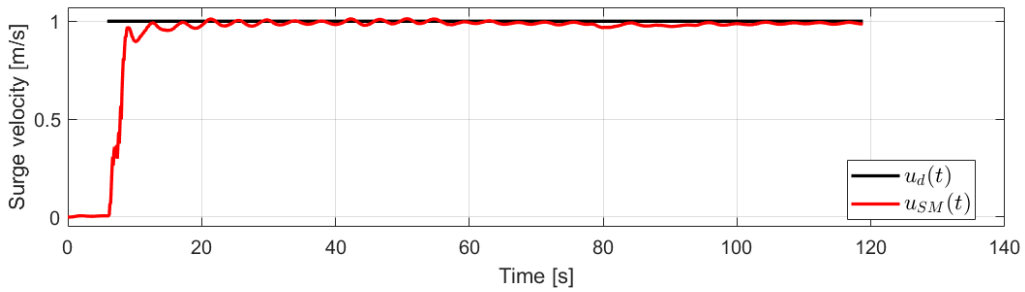
In order to better demonstrate this performance, the cross-track error and the surge velocity for both controllers throughout the path following are presented in Figure 5.15.



(a) Cross-track error.



(b) Surge velocity - PID controller.



(c) Surge velocity - SMC.

Figure 5.15: Lawnmower Path Following with u_d = 1.0 m/s: Cross-track error and surge velocity - comparison between SMC and PID.

After analysis of Figure 5.15a, it can be measured that the maximum absolute cross-track error after the initial approximation to the desired path for SMC is 0.9 m, while for the PID controller is 2.5 m, attesting the former's better performance. Also, through analysis of Figures 5.13b and 5.13c, it can be observed that both controllers succeed in driving the vehicle's surge velocity to the desired reference, having maximum absolute errors of 0.03 m/s and 0.09 m/s for SMC and PID control, respectively.

Star-shaped Mission

Furthermore, another type of manoeuvre is tested with the Aguiar algorithm, namely a star-shaped path $p(\gamma)$, where once again $p_{\text{PID}}(t)$ and $p_{\text{SM}}(t)$ are the vehicle's position when using PID control and SMC, respectively. Again, the sway controllers are explicitly given a sway reference of 0 m/s during the whole manoeuvre, in order to prevent movement in the body's y axis.

Firstly, this path is executed with a fixed surge velocity reference of 0.3 m/s, as seen in Figure 5.16.

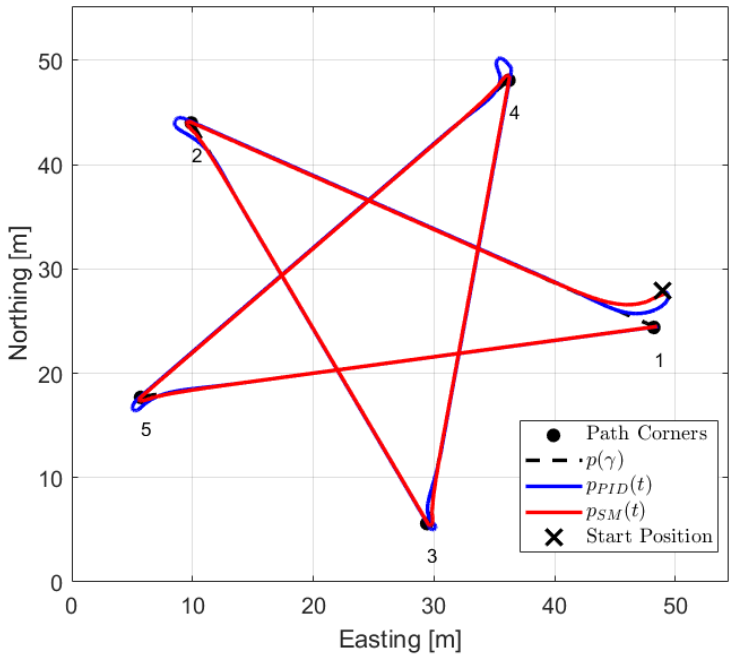
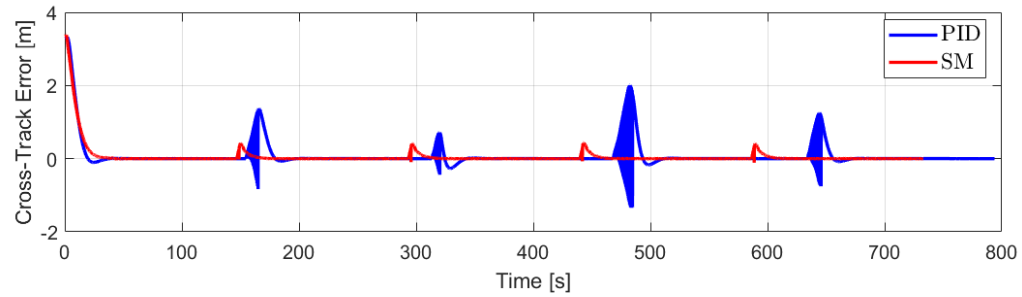


Figure 5.16: Star Path Following with $u_d = 0.3$ m/s - comparison between SMC and PID.

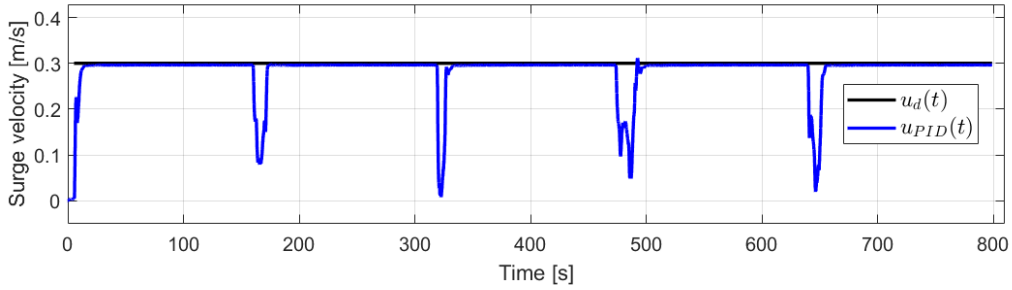
Similarly to the lawnmower path (with the same surge velocity), both controllers have good performance regarding the vehicle's initial approximation to the desired path, since it is done with no overshoot. Again, SMC leads to a faster approximation to the path, since the vehicle's yaw is driven faster to the reference when comparing to the PID controller.

Another measure of performance illustrated by this case is how easily the vehicle executes sudden changes of heading amidst path following, which take form as the five path corners (numbered from 1 to 5) seen in Figure 5.16. Generally, it can be seen that SMC demonstrates better performance for every path corner, since the vehicle is driven less further away from the desired path when "turning" each corner, when compared to the PID controller.

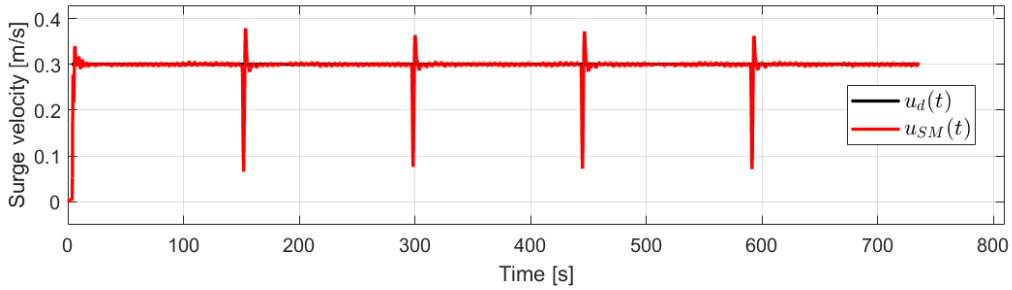
For further analysis, the cross-track error and the surge velocity for both controllers throughout the path following are shown in Figure 5.17.



(a) Cross-track error.



(b) Surge velocity - PID controller.



(c) Surge velocity - SMC.

Figure 5.17: Star Path Following with $u_d = 0.3$ m/s: Cross-track error and surge velocity - comparison between SMC and PID.

Analysis of the experimental data in Figure 5.17a reveals that the maximum absolute cross-track error after the initial approximation to the desired path for SMC is 0.4 m, while for the PID controller is 2.0 m, attesting the former's better performance.

It should be stated that the severe chattering present in the cross-track error for the PID controller corresponds to the four moments in time when the vehicle has to "turn" around star corners 2 to 5 and is caused by the way the path following is implemented in the Farol stack. In order to create complex paths like the ones presented, these are constructed by connecting different path sections to each other, such that the end of a path section is connected to the beginning of another path section. As such, a path section controller is implemented such that the vehicle can always receive references from path section to path section sequentially, which is critical when path sections intersect each other and near the points in space where path sections switch from one to the other (the star path's corners being an example of

these). However, in the latter case, this path section controller can end up switching back and forward between path sections, which results in the aforementioned behaviour of the cross-track error for the PID controller. This behaviour does not majorly affect the SMC case since the vehicle is driven to the yaw reference fast enough so as to avoid this problem.

Furthermore, by analysing figures 5.17b and 5.17c it can be seen that both controllers succeed in driving the vehicle's surge velocity to its desired reference. However, in the four times the vehicle "turns" around the path's corners, surge velocity drops given the vehicle's inability to keep its surge velocity at a constant non-zero value while performing such a sudden and sharp change of heading. Nonetheless, in SMC's case, the maximum absolute error of surge velocity after the initial approximation to the path is 0.23 m/s and the period of time with such a large error is on average 6 s, while these values for PID's case are 0.29 m/s and 14 s, respectively, showing SMC's better performance.

Secondly, this path is executed with a fixed surge velocity reference of 1.0 m/s, as seen in Figure 5.18. Once again, this test is aimed to assess how differently does the surge velocity of the vehicle impact the performance of both controllers during path following.

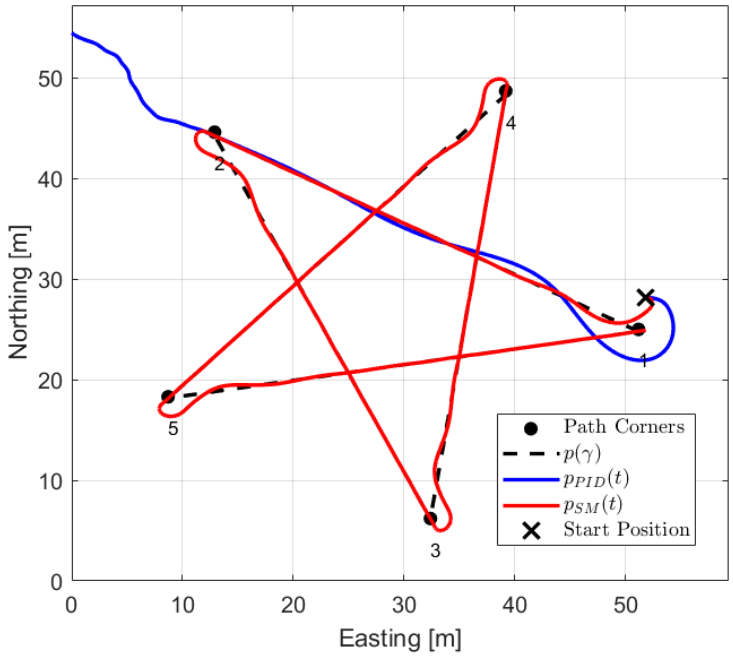


Figure 5.18: Star Path Following with u_d = 1.0 m/s - comparison between SMC and PID.

As seen in the presented figure, there is now a significant change of behaviour between both controllers. Besides the previously seen better approximation to the path manifested by SMC, it can now be observed that, after approaching the second corner of the path, the PID controller is not able to drive the vehicle to the desired path, leaving the vehicle uncontrolled, leading to the end of this mission. However, SMC secures its good performance, reaching the end of the path and "turning" around the path's corners with relative small cross-track error.

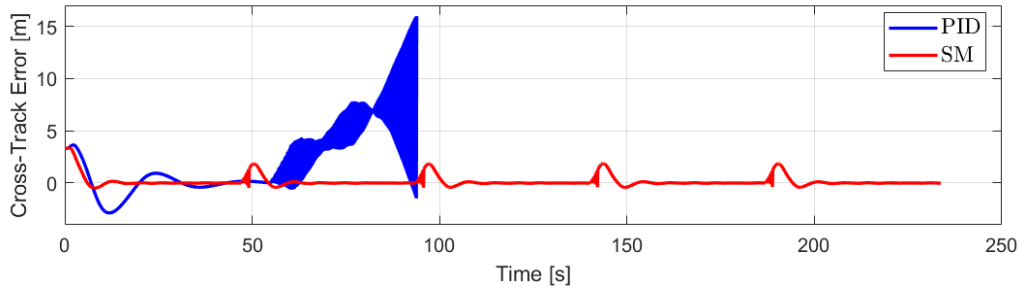
This difference in behaviour is explained by the fact that, as stated before, SMC takes into consideration the vehicle's dynamics in its control law, while PID control does not. For example, recovering the last

expression in (3.16) and solving for $\dot{r}$, yields

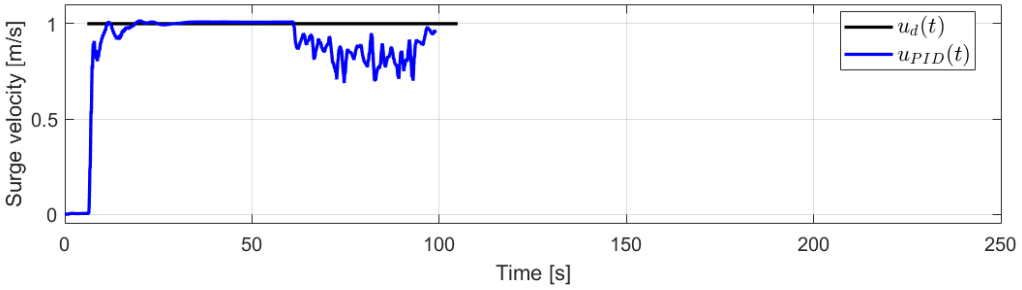
$$\dot{r} = \frac{1}{m_r}(m_{uv}uv - d_r r + \tau_r), \tag{5.5}$$

where it is clear that the derivative of the yaw rate r is influenced by the surge and sway velocity, u and v, respectively. Even though the sway velocity is constantly controlled to the reference 0 m/s, it is never exactly 0 m/s, especially when the vehicle is "turning" the path's corners, leading to the term $m_{uv}uv$ playing an ever bigger impact on the vehicle's yaw the larger its surge velocity u is. In essence, the larger the surge velocity, the more difficult it is to control the vehicle's yaw. As such, since SMC takes into account this kind of dynamics and PID does not, it is expected that the PID controller has worse performance when its surge velocity is large enough but SMC is, as previously seen in Figure 5.18.

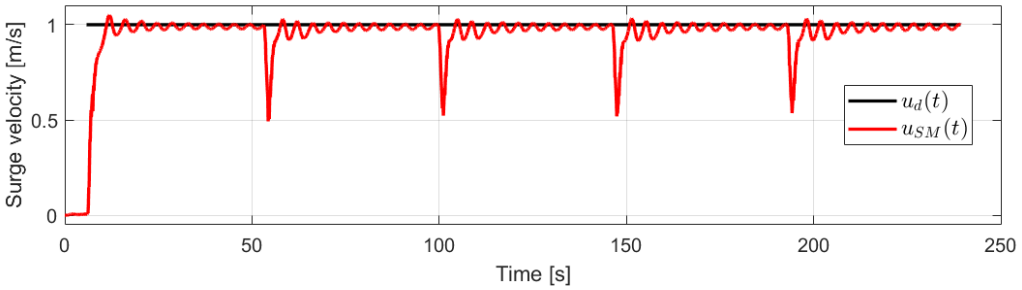
For further analysis, the cross-track error and the surge velocity for both controllers throughout the path following are shown in Figure 5.19. It should be noticed that, in PID's case, the mission is ended early since the vehicle ends up in an uncontrolled state, justifying PID's graphs for the cross-track error (Figure 5.19a) and surge velocity (Figure 5.19b) ending before $t = 100$ s.



(a) Cross-track error.



(b) Surge velocity - PID controller.



(c) Surge velocity - SMC.

Figure 5.19: Star Path Following with u_d = 1.0 m/s: Cross-track error and surge velocity - comparison between SMC and PID.

Once again, by analysing Figure 5.17a, the evidenced chattering is due to the aforementioned problem related to path section switching. In the case of SMC, the maximum absolute cross-track error is 1.9 m and the maximum absolute surge velocity error is 0.48 m/s, happening during the "turn" around of one of the path's corners, taking an average of 4 s for the vehicle to accelerate until its surge velocity reaches the reference again.

Such a large surge velocity error when "turning" around the path's corners can be explained by analysing the vehicle's thrusters' RPM commands, in percentage, as seen in Figure 5.20.

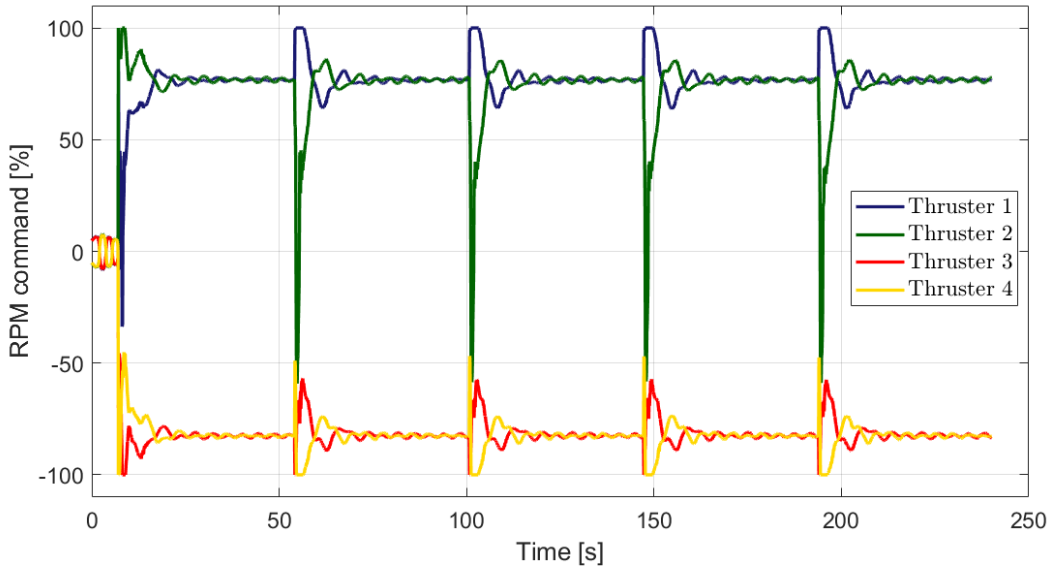


Figure 5.20: Star Path Following with u_d = 1.0 m/s - thrusters' RPM command for SMC's case.

Since the execution of path following, in this case, deals with surge and sway velocity and yaw, focus should be directed to Thrusters 1 to 4, which are the ones responsible for horizontal movement. After analysis, it is clear that Thrusters 1 and 4 saturate at 100 % at the four periods of time during which the vehicle is "turning" around the path's corners, which leads to a mechanical inability to produce the desired forces and torques necessary to reach the desired references in a smaller period of time similar to when this manoeuvre is performed at a slower surge velocity, as seen previously in Figure 5.16.

Circular Mission

Finally, the controllers are put to the test through the execution of a circular mission $p(\gamma)$ with the objective of inspecting an object of interest at the centre of the circular path. In order to perform this objective, the Rómulo algorithm is used, as defined in (2.3), where once again $p_{\text{PID}}(t)$ and $p_{\text{SM}}(t)$ are the vehicle's position when using PID control and SMC, respectively. Since this algorithm only computes references for surge and sway velocity, yaw references are computed externally, such that the vehicle is always headed towards the path's centre.

Firstly, this path is executed with a fixed profile velocity reference of 0.3 m/s, as seen in Figure 5.21.

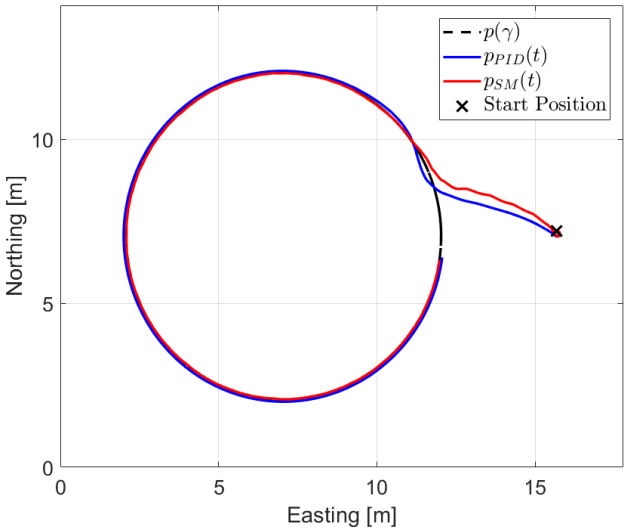


Figure 5.21: Circumference Path Following with profile velocity of 0.3 m/s - comparison between SMC and PID.

After analysis of the presented figure, it can be seen that both controllers drive the vehicle to the desired path, proving their ability to perform such task. However, SMC is able to achieve better results: on the one hand, the vehicle's initial convergence to the path is smoother and has no overshoot, as opposed to the PID controller; on the other hand, after convergence, the vehicle's maximum cross-track error is 0.02 m, while in the PID's case it is 0.10 m.

Secondly, the same path is executed, this time with a fixed profile velocity reference of 1.0 m/s, as observed in Figure 5.22.

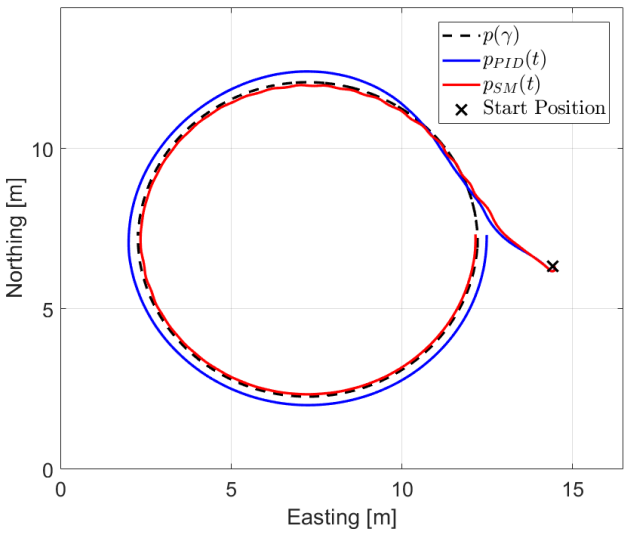


Figure 5.22: Circumference Path Following with profile velocity of 1.0 m/s - comparison between SMC and PID.

Through analysis of the presented figure, it can be seen that both controllers are still able to drive the vehicle towards the desired path, while this time the cross-track error for both of them is larger: in

the PID's case, its maximum value is 0.27 m, while for the SMC it is 0.11 m, proving the latter's better performance.

5.2.5 Results and Discussion - Unconstrained and Constrained Allocation

A test is developed to highlight the differences in behaviour between the Linear Quadratic Unconstrained Control Allocation and its constrained counterpart, as previously defined in Section 2.3.2. These two thruster allocation methods are referred to as *static* and *optimised* from now on, respectively, since the former has a closed-form solution and the latter results from an optimisation method.

The idea behind this test, in the case of the optimised method, is to understand how having knowledge of the thrusters' limitations can influence the RPM values sent to the vehicle's thrusters and its behaviour regarding reference tracking. This comes at the cost of an error between the forces $\mathbf{f}_{static}$ that produce the desired forces and torques on the vehicle's body, such that, as seen in (2.34),

$$T\mathbf{f}_{static} = \boldsymbol{\tau} \,, \tag{5.6}$$

and the forces that are actually computed $\mathbf{f}_{opt}$, such that

$$T\mathbf{f}_{opt} = \boldsymbol{\tau} + \mathbf{s} \,, \tag{5.7}$$

which is adapted from one of the constraints in the optimisation problem in (2.45).

As such, this test should induce one or more of the vehicle's thrusters to saturate at 100 %, which for the BlueROV corresponds to 3200 RPM. In the case of the static method, even if it computes a force for some thruster i that would cause an RPM value above 3200 RPM, it is simply saturated and the remaining thrusters' values stay unchanged, provided their values are lower than 3200 RPM. In the case of the optimised method, the force for thruster i would be preemptively computed to correspond to the saturation value of 3200 RPM, while the other thrusters' values would be optimally computed to minimise the cost function in (2.45), which minimises $\mathbf{s}$ through the term $\mathbf{s}^T Q\mathbf{s}$.

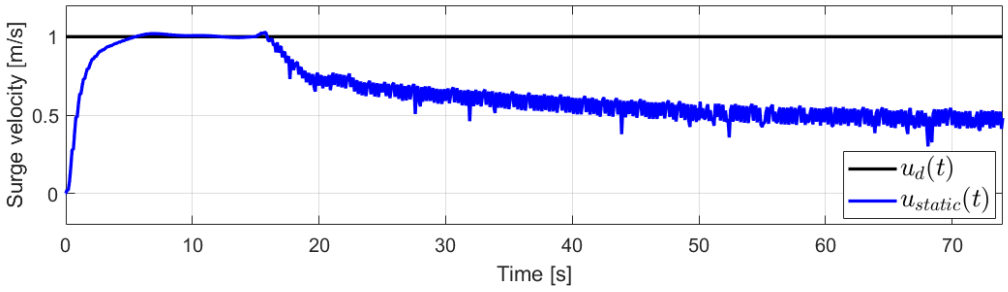
Moreover, it should be noticed that the hard limitation on the thrusters' maximum produced force (or maximum RPM value), in the case presented, inevitably leads to failing in achieving the desired forces and torques $\boldsymbol{\tau}$. The difference between both methods is that the optimised one provides a tunable parameter Q that allows the error $\mathbf{s}$ to be shaped. Fundamentally, this means that it is possible to choose which body forces and torques should be prioritised in the case of thruster saturation. For instance, in this situation, the accurate tracking of τ_r could be prioritised over τ_u, which would lead to relatively smaller errors regarding yaw or yaw rate control, in comparison to surge velocity errors. Using the static method, one error could not be minimised in detriment of another and neither of them could be easily predicted.

Test definition and results

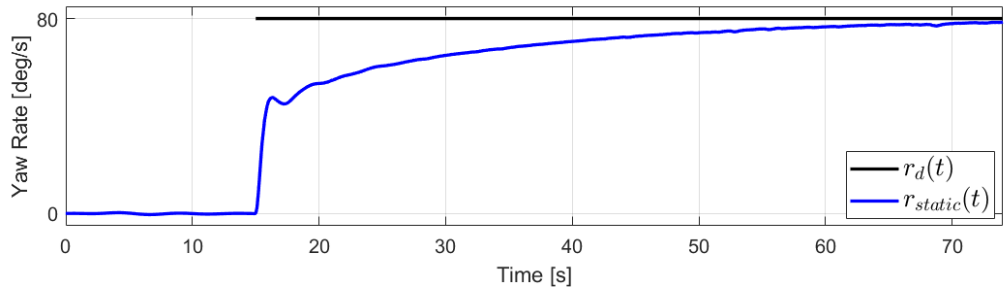
Taking into account the previously described behaviour, the developed test consists of using the SMC surge velocity controller and a yaw rate PID controller (since no SMC controller for yaw rate was

developed), the latter previously tuned by the research team at DSOR, in order to reach a surge velocity u of 1.0 m/s and a yaw rate r of 80 deg/s at the same time. Essentially, from $t = 0$ s to $t = 15$ s only the surge velocity is being controlled and from $t = 15$ s to $t = 75$ s both the surge velocity and the yaw rate are being controlled.

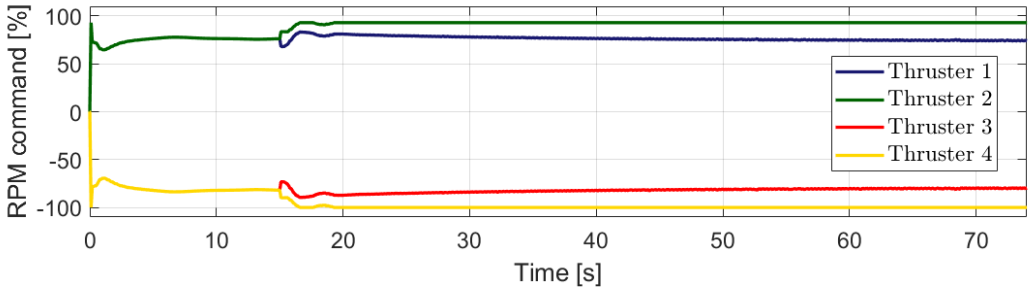
In Figure 5.23, both surge velocity and yaw rate can be observed when the static thrust allocation method is tested.



(a) Surge velocity.



(b) Yaw Rate.



(c) Thrusters' RPM command, in percentage.

Figure 5.23: Thruster saturation and its impact on the vehicle's behaviour - static thruster allocation.

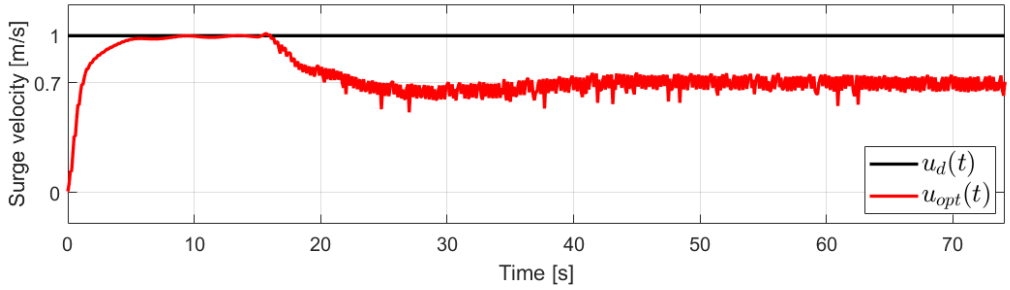
After analysis, it can be stated that, until $t = 15$ s, the surge velocity controller succeeds in driving the vehicle to the desired reference u_d, as seen in Figure 5.23a. After this instant of time, the yaw rate controller starts computing desired values for the Z axis torque, which leads to the saturation of thruster 4, as seen in Figure 5.23c. Ultimately, this ends up resulting in the vehicle's yaw rate reaching its desired value, as observed in 5.23b, and the surge velocity chattering with a mean value near 0.5 m/s. In conclusion, under these circumstances, the static method leads to prioritising yaw rate tracking

performance, in detriment of surge velocity.

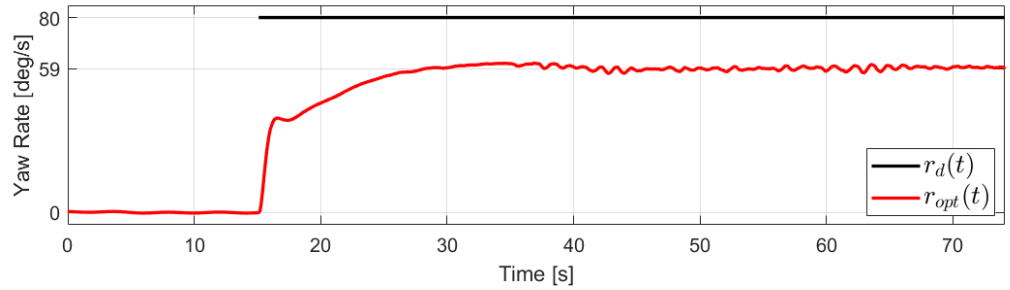
In order to take control of which vehicle DOF to prioritise, the optimised method is also tested under the same circumstances. With the purpose of illustrating this property, matrix $Q = Q_1$ is devised in order to prioritise the vehicle's surge velocity instead of its yaw rate. As such,

$$Q_1 = \text{diag}\{2000,\ 200,\ 200,\ 200,\ 200,\ 20\}\ , \tag{5.8}$$

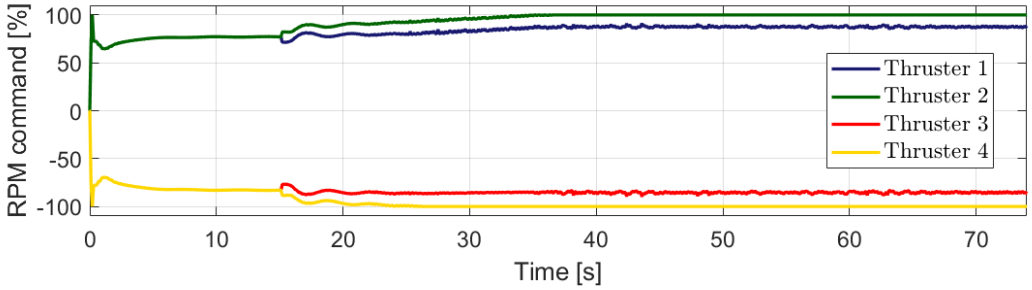
which weighs surge force error with a coefficient of 2000 and yaw torque error with a smaller value of 20, leading to the desired behaviour of reducing surge velocity error in prejudice of a larger yaw rate error. This test is illustrated in Figure 5.24.



(a) Surge velocity.



(b) Yaw Rate.



(c) Thrusters' RPM command, in percentage.

Figure 5.24: Thruster saturation and its impact on the vehicle's behaviour - optimised thruster allocation.

After thorough analysis, it can be seen that, once again, after $t = 15$ s, the vehicle's yaw rate starts being controlled and its surge velocity lowers. However, this time, yaw rate does not reach its desired

value, stabilising with an approximate error of 21 degrees (Figure 5.24b), and surge velocity ends up totalising an error of approximately 0.3 m/s (Figure 5.24a), which is lower compared to the static method. It can also be noticed that thrusters 2 and 4 end up saturating, as seen in Figure 5.24c.

Finally, it can be confirmed that optimised thrust allocation can lead to more predictable scenarios when it comes to situations where the vehicle's thrusters saturate. For instance, when a fully actuated vehicle is traversing a certain path with multiple obstacles, surge and sway velocity could be prioritised over yaw or yaw rate as seen before, since obstacle avoidance and keeping up with a specific speed profile is often easier when the vehicle keeps its yaw fixed and since preventing collision is related to the vehicle's position and not its heading angle.

Chapter 6

Experimental Results

6.1 Location

The Task Group for the Extension of the Continental Shelf (EMEPC) is a Portuguese endeavour to gather bathymetric, geological, geophysical and biological data of the continental shelf, beyond 200 nautical miles from the baselines from which the breadth of Portugal's territorial sea is measured. Its facilities include a large tank, where its own ROV, ROVLuso, is tested.

In order to assess SMC's performance in a real environment, several tests were conducted at EMEPC's facilities in Paço de Arcos, Oeiras, Lisbon, controlling the BlueROV inside the aforementioned tank. In Figure 6.1, a photograph of BlueROV during the tests at EMEPC's facilities can be observed.

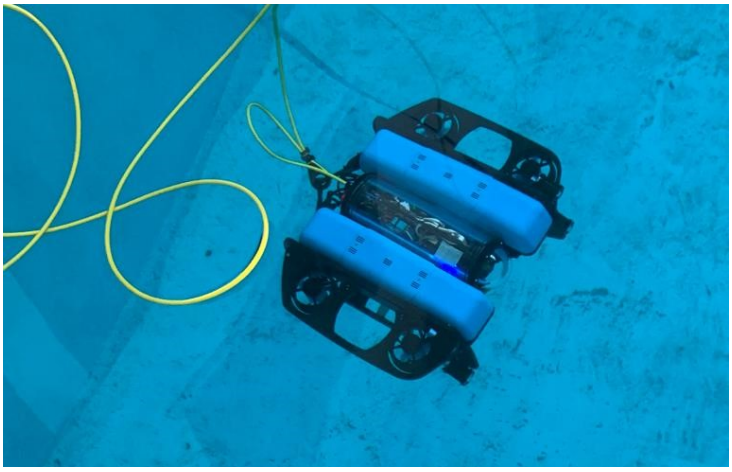


Figure 6.1: The BlueROV vehicle during experimental tests inside EMEPC's tank.

Since inside EMEPC's facilities GPS would not be sufficiently reliable, the vehicle's position is computed through the integration of the vehicle's velocity, measured by a Doppler Velocity Log (DVL), and the vehicle's attitude, measured by its Inertial Measurement Unit (IMU).

6.2 Results and Discussion - SMC

In the interest of experimenting as much as possible, a large variety of tests were conducted, with a focus on diversity, since time for testing was admittedly limited, resulting in reduced opportunities for perfect parameter tuning of SMC's controllers, namely yaw's and surge and sway velocities'.

6.2.1 Constant References

Several tests are conducted with multiple constant references for yaw and surge and sway velocities. Firstly, SMC's yaw controller is tested, as seen in Figure 6.2, where ψ_d is the yaw reference signal and ψ_{SM} is the vehicle's yaw.

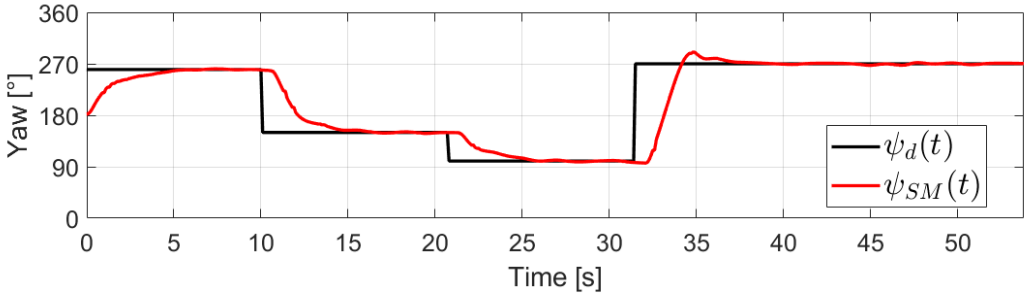
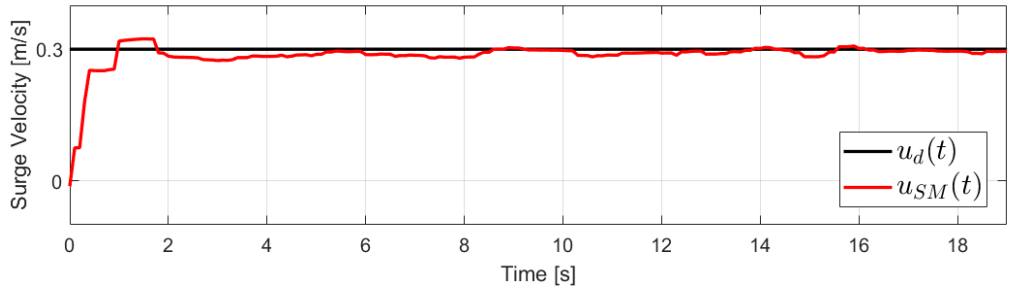


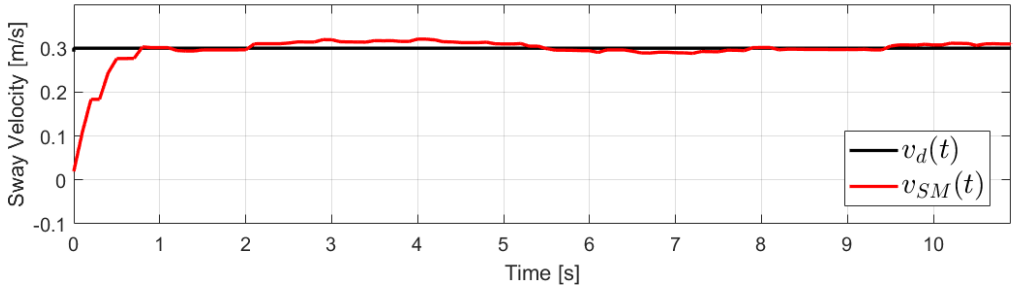
Figure 6.2: Yaw SMC - performance in response to constant references.

After analysis, it can be seen that the controller succeeds in driving the vehicle to the desired yaw reference, although the convergence is quite slow, taking around 3 to 4 s to reach it, due to the previously mentioned insufficient parameter tuning. Also, the overshoot at $t \approx 35$ s is significant, surpassing the desired reference by 20 degrees.

Secondly, the surge and sway controllers are individually tested, as seen in Figure 6.3, where u_d and v_d are the references and u_{SM} and v_{SM} are the vehicle's velocities, for surge and sway, respectively.



(a) Surge velocity.



(b) Sway velocity.

Figure 6.3: Surge and sway SMC - performance in response to constant references.

It can be observed that both controllers achieve the desired references for surge and sway in about 1 s. It is evident that there is some oscillation around the reference after the initial convergence, which can be associated to the natural SMC chattering, ineffective parameter tuning and environmental disturbances (such as the water circulation system of the tank). Moreover, an unexpected stabilisation of the surge and sway signals seems to occur frequently, which can be attributed to the sensor's filter. On the one hand, it processes sensor data at a lower frequency than the controller's, which leads to the occasional repetition of the signal's last value. On the other hand, sometimes the filter might consider some sensed data to be an outlier, which leads to it outputting the last output value.

6.2.2 Path Following Missions

Since all three controllers responsible for the system's inner loop demonstrate satisfactory performance, two path following missions are now devised, small enough to fit the facilities' tank, and executed using the Aguiar algorithm. Firstly, in Figure 6.4, a lawnmower path is executed by the BlueROV vehicle, where $p(\gamma)$ is the desired path and p_{SM} is the position of the vehicle itself.

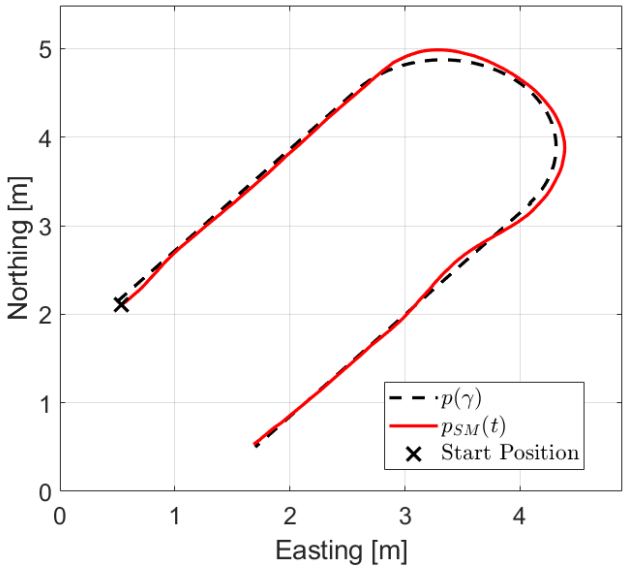


Figure 6.4: Lawnmower Path Following with profile velocity of 0.1 m/s.

It can be observed that the vehicle is able to follow the desired path with minor cross-track error during the straight sections of the path, while it has a maximum cross-track error of 0.1 m during its semicircular sections, resulting in a natural overshoot afterwards, since there is no sway control in this test. With it, it is predicted that the cross-track error during semicircular sections would be reduced and overshoot severely minimised.

Secondly, a circular path is created, as seen in Figure 6.5, where once again $p(\gamma)$ is the desired path and p_{SM} is the position of the vehicle itself.

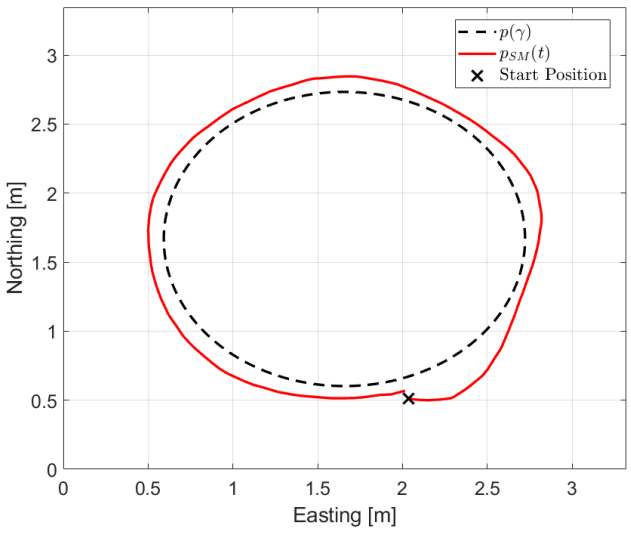


Figure 6.5: Circumference Path Following with profile velocity of 0.2 m/s.

It can be seen that the vehicle struggles to follow the desired path, having an average cross-track error of 0.11 m, which can be explained mainly by the absence of sway control to 0 m/s and the lack of enough parameter tuning of the SMC controllers. However, the cross-track error is kept around the same value during the whole mission, allowing the vehicle to perform the complete circumference.

Chapter 7

Conclusion

This work addressed the development of a nonlinear control method for Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs) called Sliding Mode Control (SMC), focusing on the craft's surge velocity, sway velocity and yaw. Initially, a simple version of this controller was developed for fast iteration, which led to promising preliminary results, which attested this control solution for underwater vehicles. Furthermore, an improved controller formulation was also devised, targeting SMC's main disadvantage (control input chattering) and fast convergence, which were both tackled successfully, since the new controller had in fact better performance in terms of faster convergence and lower chattering effect.

Initially, a general overview of the state of the art was undertaken, reviewing Trajectory Tracking and Path Following strategies, several control methods and different allocation methods for underactuated, fully actuated and overactuated vehicles. Secondly, the general notation and reference frames for the vehicle's model was presented, focusing on a general AUV's kinematics and dynamics. The specifications of the vehicles used (MEDUSA and BlueROV) was also introduced. Moreover, the SMC controllers were designed, initially providing a simpler approach and, later, an improved version. Parameter influence on performance was thoroughly analysed too.

Finally, a broad range of simulations was conducted, in order to test SMC controllers in comparison to PID's and equate the performance of two different thrust allocation methods. Firstly, the main objectives of SMC, when compared to the traditionally used Proportional-Integral-Derivative (PID) control, focuses on the rejection of vehicle modelling uncertainty and environmental disturbances, both of which have been proved to be successfully tackled through simulations in *MATLAB* and ROS/*C*++. The results obtained showed that SMC laws exhibit a good degree of robustness against model uncertainty, together with external disturbance attenuation. Likewise, it was also demonstrated that using Constrained Control Allocation over its unconstrained counterpart reflects on an ability to prioritise some vehicle's DOFs over others when its thrusters are saturated.

7.1 Future Work

Despite the good results obtained, some issues still remain as problems that have been left unattended and which still need to be addressed, both in the current improved SMC implementation as well as the Farol stack, from which the outer-loop guidance was derived. These include:

- Improving the path section switching logic and its chattering effect on the generated reference values, during multi-sectioned path following;
- Improving the path following algorithms, by lowering generated velocity references in order to account for path section transitions with discontinuous curvature or generate smoother paths;
- Preventing thrusters' deadzone delay effect on the vehicle's state variables transitioning between 0 to non-zero values. This could be implemented by leaving the thrusters rotating at a fixed speed, such that the sum of forces or torques is still 0;
- Analysing the effect of the time variable ϵ (SMC parameter) on SMC's proof of stability and on general performance, regarding disturbance rejection and sliding surface chattering;
- Evaluating other SMC's reaching laws, which could improve overall performance;
- Improving SMC parameter tuning for all three developed controllers in the real environment;
- Evaluating the importance of developing SMC controllers for other vehicle's DOFs, such as roll, pitch and heave or developing a vectorial SMC controller for all vehicle's DOFs.

References

[1] A Pedro Aguiar and Joao P Hespanha. 'Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty'. In: *IEEE transactions on automatic control* 52.8 (2007), pp. 1362–1379.

[2] A Pedro Aguiar, Joao P Hespanha and Petar V Kokotovic. 'Path-following for nonminimum phase systems removes performance limitations'. In: *IEEE Transactions on Automatic Control* 50.2 (2005), pp. 234–239.

[3] M Araki. 'Control Systems, Robotics and Automation-Vol. II'. In: *Encyclopaedia of Life Support Systems (EOLSS), Ch. PID Control* (2009).

[4] *Blueprint Lab and VideoRay in Partnership for New EOD Tool for US Navy*. URL: https://www.oceannews.com/news/subsea-and-survey/blueprint-lab-and-videoray-in-partnership-for-new-eod-tool-for-us-navy (visited on 23/03/2020).

[5] Matthew S Dodd et al. 'Evidence for early life in Earth's oldest hydrothermal vent precipitates'. In: *Nature* 543.7643 (2017), pp. 60–64.

[6] Ben Ford, Amy Borgens and Peter Hitchcock. 'The 'Mardi Gras' Shipwreck: Results of a Deep-Water Excavation, Gulf of Mexico, USA'. In: *International Journal of Nautical Archaeology* 39.1 (2010), pp. 76–98.

[7] Thor I Fossen. *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons, 2011.

[8] Thor I Fossen, Morten Breivik and Roger Skjetne. 'Line-of-sight path following of underactuated marine craft'. In: *IFAC proceedings volumes* 36.21 (2003), pp. 211–216.

[9] Thor I Fossen et al. 'A survey of control allocation methods for underwater vehicles'. In: *Underwater vehicles* (2009), pp. 109–128.

[10] H Gary Greene et al. 'APPLICATION OF A REMOTELY OPERATED VEHICLE IN GEOLOGIC MAPPING OF MONTEREY BAY, CALIFORNIA, USA.' In: (1993).

[11] Christopher Harrold, Karen Light and Susan Lisin. 'Distribution, Abundance, and Utilization of Drift Macrophytes in a Nearshore Submarine CanyonSystem'. In: *Diving for Science... 1993. Proceedings of the American Academy of Underwater Sciences (13th annual Scientific Diving Symposium). Retrieved*. 2008, pp. 07–11.

[12] Nguyen Hung et al. ‘A review of path following control strategies for autonomous robotic vehicles: theory, simulations, and experiments’. In: *arXiv preprint arXiv:2204.07319* (2022).

[13] Tor A Johansen, Thor I Fossen and Petter Tøndel. ‘Efficient optimal constrained control allocation via multiparametric programming’. In: *Journal of guidance, control, and dynamics* 28.3 (2005), pp. 506–515.

[14] Tor A Johansen et al. ‘Optimal constrained control allocation in marine surface vessels with rudders’. In: *Control engineering practice* 16.4 (2008), pp. 457–464.

[15] Clayton Kunz et al. ‘Deep sea underwater robotic exploration in the ice-covered arctic ocean with AUVs’. In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2008, pp. 3654–3660.

[16] L Lapierre, D Soetanto and A Pascoal. ‘Nonlinear path following with applications to the control of autonomous underwater vehicles’. In: *42nd IEEE international conference on decision and control (IEEE cat. no. 03ch37475)*. Vol. 2. IEEE. 2003, pp. 1256–1261.

[17] Tat-Hien Le et al. ‘Robust position control of an over-actuated underwater vehicle under model uncertainties and ocean current effects using dynamic sliding mode surface and optimal allocation control’. In: *Sensors* 21.3 (2021), p. 747.

[18] Anastasios M Lekkas and Thor I Fossen. ‘Line-of-sight guidance for path following of marine vehicles’. In: *Advanced in marine robotics* (2013), pp. 63–92.

[19] Jinkun Liu and Xinhua Wang. *Advanced sliding mode control for mechanical systems*. Springer, 2012.

[20] Thomas R Martin. *Ancient Greece: from prehistoric to Hellenistic times*. Yale University Press, 2013.

[21] Willy K Mojsznis and Terry Blevins. ‘Evolving PID tuning rules’. In: *Control Engineering* 60.3 (2013), pp. 1–6.

[22] *NOAA. Historical Maps and Charts audio podcast. National Ocean Service website*. URL: https://oceanservice.noaa.gov/podcast/july17/nop08-historical-maps-charts.html (visited on 13/08/2017).

[23] Yongping Pan, Young Hoon Joo and Haoyong Yu. ‘Discussions on smooth modifications of integral sliding mode control’. In: *International Journal of Control, Automation and Systems* 16.2 (2018), pp. 586–593.

[24] M Pidwirny. *Introduction to the Oceans. Fundamentals of Physical Geography, Date Viewed: 2013 March*. 2006.

[25] Francesco Pretagostini et al. ‘Survey on wheel slip control design strategies, evaluation and application to antilock braking systems’. In: *IEEE Access* 8 (2020), pp. 10951–10970.

[26] Wilson J Rugh and Jeff S Shamma. ‘Research on gain scheduling’. In: *Automatica* 36.10 (2000), pp. 1401–1425.

[27] Jeff S Shamma and Michael Athans. ‘Gain scheduling: Potential hazards and possible remedies’. In: *IEEE Control Systems Magazine* 12.3 (1992), pp. 101–107.

[28] Jean-Jacques E Slotine, Weiping Li et al. *Applied nonlinear control*. Vol. 199. 1. Prentice hall Englewood Cliffs, NJ, 1991.

[29] Robert J Stern. ‘Commentary on JGR-Sold Earth Paper “Deep Seismic Structure Across the Southernmost Mariana Trench: Implications for Arc Rifting and Plate Hydration” By Wan et al.’ In: (2019).

[30] Jian-Xin Xu and Ying Tan. *Linear and nonlinear iterative learning control*. Vol. 291. Springer, 2003.

[31] Stanislaw H Zak. *Systems and control*. Vol. 198. Oxford University Press New York, 2003.

Appendix A

MEDUSA Motor References

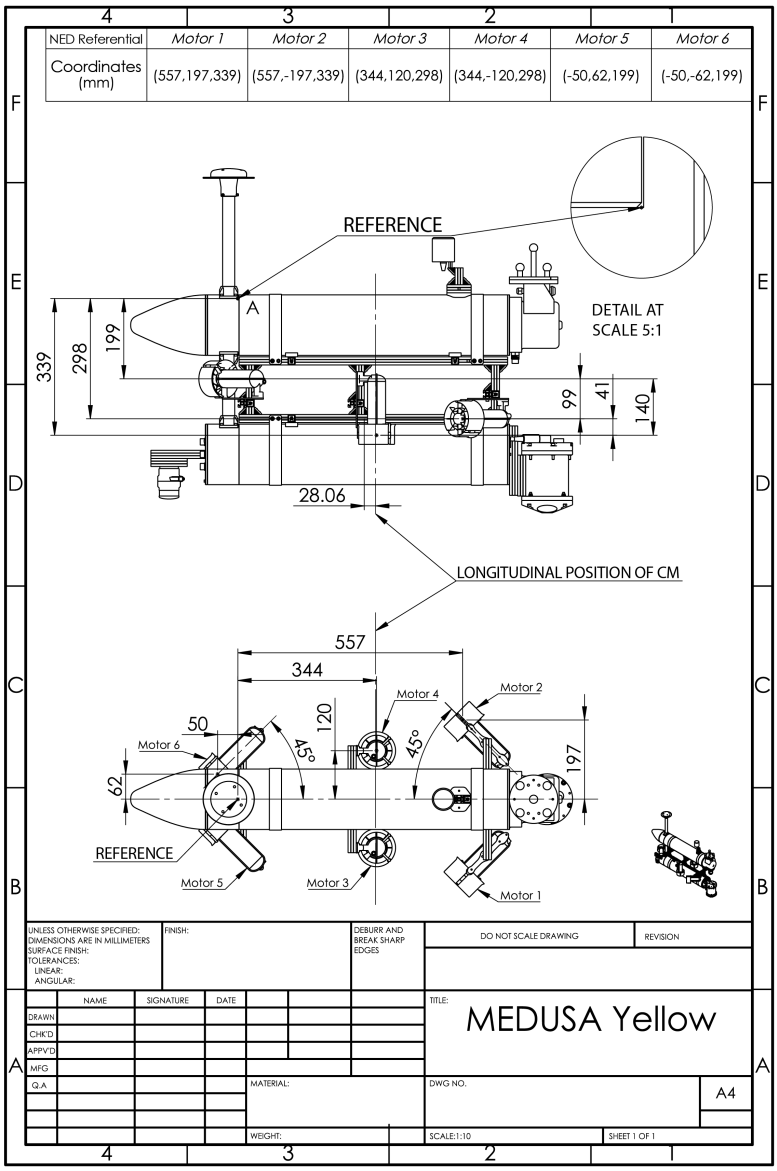


Figure A.1: MEDUSA Motor References; lengths in milimetres.