# POLITECNICO DI TORINO

**Master's Degree in Computer Engineering**


**Master's Degree Thesis**

# Modeling and Control of Remotely Operated Underwater Vehicles


| Supervisor | Candidate |
| --- | --- |
| Prof. Carlo NOVARA | Angelo PETTINELLI |


**July 2022**

# Summary

Remotely Operated Underwater Vehicles (ROUVs, or just ROVs) belong to the larger group of Unmanned Underwater Vehicles (UUVs), submersible vehicles that operate underwater without the presence of a man on board, as they are remotely driven. Among the many different kinds of UUVs, ROVs are wire-guided by a human pilot which makes decisions and controls the vehicle. ROVs play an important role in a number of underwater missions for marine science, oil and gas extraction, exploration and salvage. For their various and important applications, in recent years there has been a significant increase in ROV's demand from industry, which has stimulated increasing interest from researchers.

Indeed, the marine environment, characterized by non-linear hydrodynamic effects and unknown disturbances, such as waves and currents, together with the uncertainty about the parameters and the lack of a precise model, make the ROV a complex system to identify and control.

In this thesis work, a control system has been designed and developed to be implemented on EVA, an ROV prototype developed by PoliTOcean, a Student Team of Politecnico di Torino, active in underwater robotic field. First, a 6 degree-of-freedom (6-DOF) mathematical model of the prototype has been designed. Then, a model-based control system has been developed and shown to work in simulations. Two different control techniques (PID and LQI/LQR), which are widely used in underwater applications, have been proposed and compared.

The comparison showed that regarding the speed controller, the LQI allows to reach the settling time faster than the PID controllers. However, PIDs, especially for angular velocities, stem better the oscillations. As for the position controller, the LQR performs better than the PIDs, if the saturation of the motors is avoided. Otherwise, the PIDs are able to better manage the non-linearities introduced by saturation, allowing faster settling times.

# Acknowledgements

I wish to thank my supervisor, prof. Carlo Novara, for guiding me in this thesis work with extreme competence and kindness, helping me to resolve all my doubts. Precious, for me, were our moments of "scientific conversation" in which prof. Novara has always shown patience, support, and attention to my needs.

I would like to thank prof. Claudio Sansoè, supervisor of the PolitOcean Team, for helping me in the various activities of the Team, from the organizational to the administrative aspect, to the scientific one, with evident enthusiasm and passion.

I would like to thank my teammates at PolitOcean, companions of a long journey that lasted, for me, four years. Some briefly, some all of the time, shared with me this unforgettable experience, which taught me the value of working together. I am fortunate to have been a part of this Team.

Thanks to my friends, the old ones, and the new ones, Francesco, Rocco, Guido, Mattia, who shared with me the joys and sorrows of life away from home. Thanks to Lorenzo, Alessandro and Francesco, companions of this university adventure, exam after exam.

Special thanks to Federica who was able to stay close to me even from afar. Thank you for encouraging me in every step of my life.

And last but not least, thanks to my family who supported me in everything and gave me the opportunity to get here and be who I am.

# Table of Contents

# List of Tables

# List of Figures

# Acronyms

**AI**
Artificial Intelligence

**AOA**
Angle of Attack

**ARE**
Algebraic Riccati Equation

**AUV**
Autonomous Underwater Vehicle

**DOF**
Degrees of Freedom

**ESC**
Electronic Speed Controller

**IMU**
Inertial Measurement Unit

**LQI**
Linear Quadratic Integral

**LQR**
Linear Quadratic Regulator

**LS**
Least Squares

**ML**

Machine Learning

**PCB**

Printed Circuit Board

**PID**

Proportional Integral Derivative

**ROUV**

Remoted Operated Underwater Vehicle

**ROV**

Remotely Operated Vehicle

**SID**

Systems Integration Diagram

**SMC**

Sliding Mode Control

**SPI**

Serial Peripheral Interface

**SVD**

Singular Value Decomposition

**UUV**

Unmanned Underwater Vehicle

# Chapter 1

# Introduction

In recent years, the interest in exploring and studying the underwater world has grown both for scientific/educational as well as for industrial purposes. This has prompted the scientific community to focus, in the last few decades, its attention on the design, development and implementation of unmanned underwater vehicles (UUVs) in order to meet the growing demand for systems suitable for reaching the sea depths and performing complex tasks in environments which are inhospitable and hazardous for humans [1].

UUVs are submersible vehicle that can operate underwater without the presence of human aboard. Indeed, they can play an important role in underwater missions for marine science, seabed exploration, oil and gas extraction, recovery, and rescue due to their flexibility and long working time in water. Based on their shape and operations, UUVs are classified into two kinds: Autonomous Underwater Vehicles (AUVs) and Remotely Operated Underwater Vehicles (ROUVs, or just ROVs). In Figure 1.1 a schematic classification of Underwater Vehicles is reported.

AUVs are unmanned, un-tethered underwater vehicles with on-board battery bank and relies on on-board circuits with preprogramed artificial intelligence and commands to execute a mission [2]. AUVs are out of scope of the topics covered by this thesis, which is instead focused on modeling and control of ROVs.

## 1.1 ROVs

ROVs were first introduced in the 1970s for work in the oil and gas industry and, since then, great efforts have been made for their improvement and implementation so that a rapid development of this class of vehicles occurred.

While AUVs are autonomous vehicles, ROVs require the control of an operator. They can be either tethered to the power source or battery supplied. Because of the difficulty of the underwater wireless communication, due to the physical

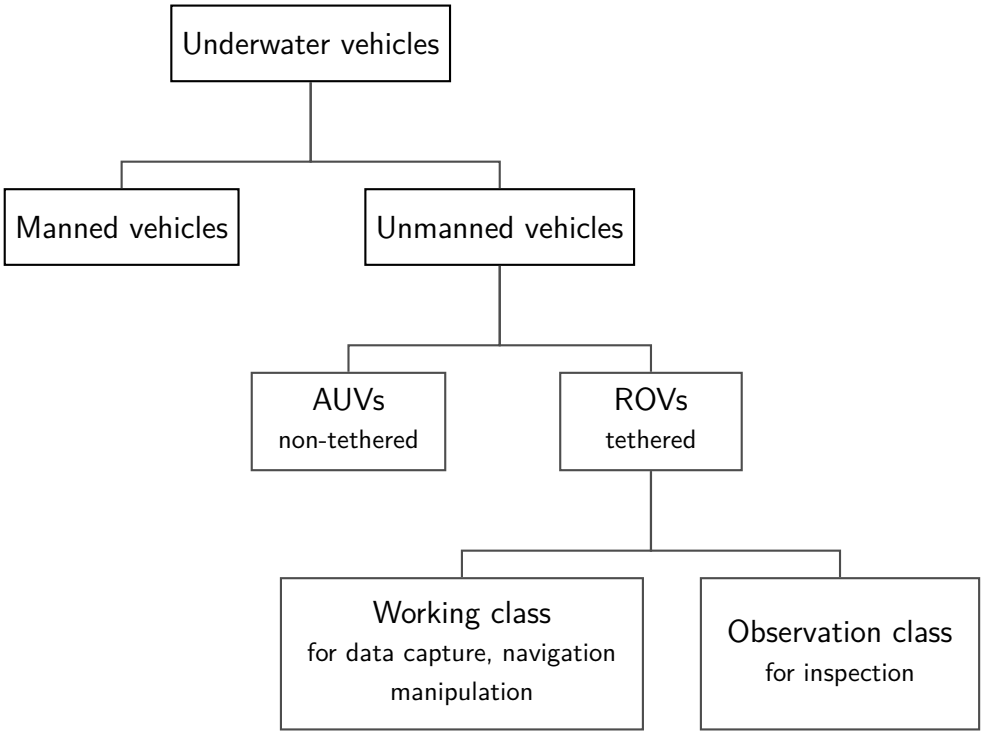



**Figure 1.1:** Classification of underwater vehicles

characteristics of the underwater environment which hinder EM signals deep penetration and propagation, ROVs are usually tethered to the control station over ethernet or linked to a proximate buoy communicating over Wi-Fi. The operator remotely controls the ROV, usually on a vessel or a proximate land.

Based on the required operation, ROVs may be equipped with video cameras to capture underwater images, sensors, and a robotic manipulator that allows to both transport objects and use auxiliary tools.

Because it is remotely controlled, ROVs are used to operate in areas that are difficult to reach or considered risky for humans and they have gradually replaced divers.

ROVs find application in several fields such as:

- Aquaculture
- Commercial and salvage diving
- Military
- Research
- Oil and energy

- Ocean safeguard


**(a)** Maintenance

**(b)** Repairing

**(c)** Exploration and research

**Figure 1.2:** ROVs operating underwater.

In particular, ROVs are commonly used for the maintenance and repair of underwater structures, mostly related to the offshore oil industry [3] (Figure 1.2a and 1.2b). Typical applications also include transportation and assembly of underwater structures and their inspection. ROVs used in marine industry are heavy, bulky, and highly expensive, due to the high operational and maintenance cost, as skilled operators are necessary. They are usually equipped with robotic manipulators capable of grasping targeted objects in underwater and require advanced sensors and control system [4].

ROVs used for research and exploration purposes (Figure 1.2c) are smaller in size and cheaper than ROVs used in marine industry. They are characterized by low-cost high-precision sensors and compact onboard computers. Such compact ROVs are essential tools for educators, engineers, scientists and environmental activists [4]. Indeed, the growing amounts of marine litter and the slow degradation rate result in its accumulation in the oceans, and this becomes one of the major environmental

threats of the twenty-first century [5]. ROVs can be used to monitor the water quality by using temperature sensors or pHmeters, to determine habitat diversity by using machine learning algorithms and to remove plastic pollution if equipped with a robotic manipulator, contributing to maintain a healthy environment.

## 1.2 Control system

A control system comprises several processes which stabilizes the underwater vehicle, so that it obeys the programmed instructions. Designing a control system which provides command signals in controlling the ROV in a multi-axis motion to obtain the desired position, the desired velocity and to follow the desired trajectory, is not trivial as the highly nonlinear hydrodynamic effect resulting from the interaction with the environment cannot be easily quantified. Indeed, the underwater scenario is very complex and highly dynamic: the pressure that increases with depth, underwater currents, interaction with waves and obstacles can cause turbulence which negatively affect the performance of the vehicle and make it difficult to stabilize the ROV [6]. The uncertainty of parameters such as the hydrodynamic coefficients and the added mass due to the water entering the pumps contributes to the difficulty to control the ROV [7].

Therefore, the major problem in designing the control system of an ROV is modeling the uncertainties of these parameters and developing the best control system to overwhelms these problems and obtain the best performance of the ROV is one of the fundamental goals to be achieved in the research field of UUVs. Figure 1.3 shows a feedback control system taking into account the environmental disturbances $G_d(s)$.

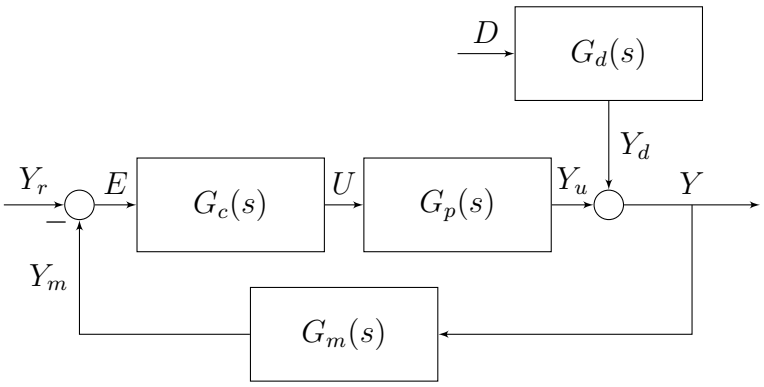



**Figure 1.3:** Closed loop (feedback) control system design.

A great number of works related to ROV control strategies are available in literature. The commonly used control methods for the underwater vehicles are

described below.

### 1.2.1 Proportional Integral Derivative (PID)

Proportional Integral Derivative (PID) is one of the most used control techniques due to its simplicity of design and implementation. It is especially used when the mathematical model of the system to be controlled is not (or only partially) known, through automatic calibration procedures. For 6-DOF systems, as in the case of ROVs, one widespread practice is to design 6 different PIDs, each for each degree of freedom [1].

### 1.2.2 Linear Quadratic Regulator (LQR)

Linear Quadratic Regulator (LQR) is an optimal control technique for linear systems. The control law is realized through a static state variable feedback control architecture. Because of the state variable feedback, all the state must be measurable (or observable). So, the mathematical model of the system to be controlled must be well-known, and it must be equipped with sensors to measure all the state variables involved.

### 1.2.3 Sliding Mode Controller (SMC)

Sliding mode control (SMC) is a nonlinear control method that provides robustness. In fact, it is largely used dealing with uncertainty. From a control point of view, modeling inaccuracies can be classified into two kinds:

- *structured* (or *parametric*) uncertainties
- *unstructured* uncertainties (or *unmodeled dynamics*)

The first corresponds to inaccuracies on the terms actually included in the model, while the latter corresponds to a simplified representation of the system's dynamics [8, Chapter 7].

A sliding surface is defined as a subset of the state space, on which the trajectory of the plant is desired to lie. A control law $u$ is designed to bring the plant trajectory towards the sliding surface and, once there, to stay close to this surface, so such that the sliding surface $S$ is an invariant set and attractive. The presence of a discontinuous term in the control law to ensures attractiveness, might cause chattering effect, an undesired phenomenon which can lead to high frequency modes causing oscillations around the sliding surface and destabilizing the system [1].

## 1.3 PoliTOcean

PoliTOcean is a Student Team of Politecnico di Torino, active since 2017 in underwater robotic field. In particular, the Team is dedicated to the design and development, through self-construction, of ROV prototypes for exploration of underwater environment. The realized prototype of ROV is tested with the participation to the *MATE ROV Competition* (Sec. 1.4), an international academic challenge addressed to a global community of students, that takes place every year, during the summer period, in the United States of America.

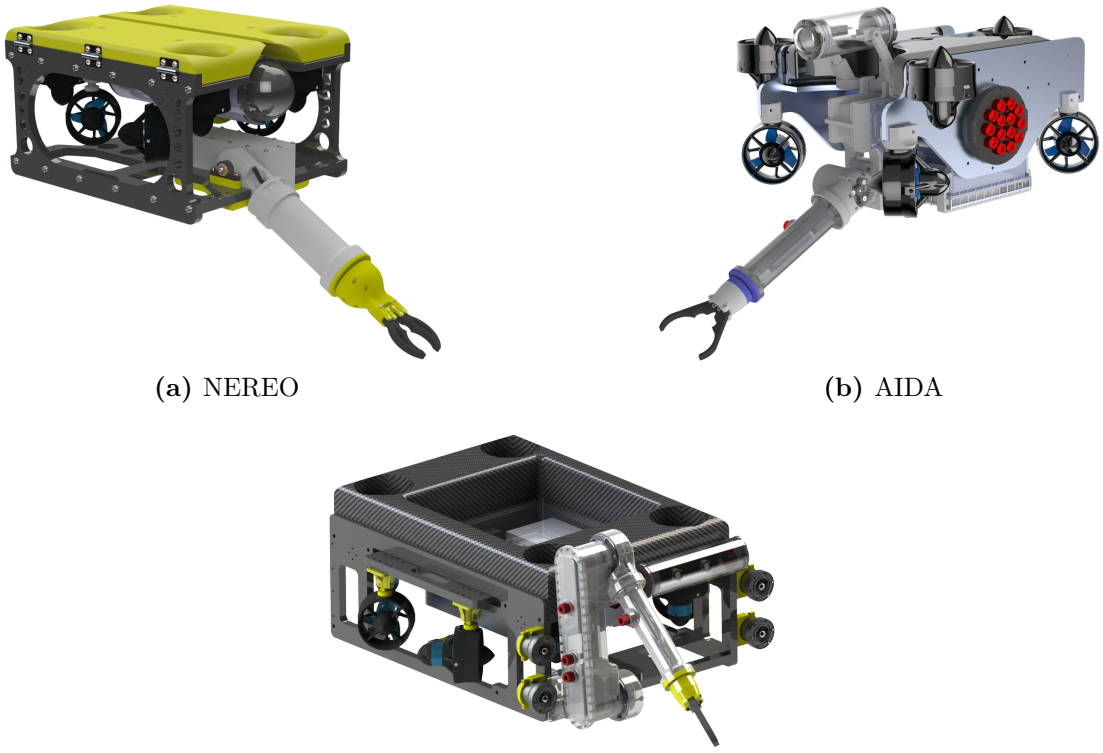
The studies are related to different topics such as mechanics, information technology and electronics, so they involve students coming from different areas of engineering, with different profiles and academic backgrounds, as well as from different countries and culture, resulting in a multidisciplinary and multi-ethnic team. The activity carried out within the Team allows the participating students to develop advanced knowledge on robotics and automation, gaining high professionalized skills.

PoliTOcean is made up of three sub-units: IT, Electronics and Mechanics. The activity of each sub-unit is supervised by a manager, while a Team Leader coordinates the three sub-units and takes care of administrative as well as organizational and scientific management of the Team. The tutor and scientific referent of PolitOcean is Prof. Claudio Sansoè and I filled the position of Team Leader and IT Manager.

During its activity, PoliTOcean team has designed and developed three different prototypes of ROV: *NEREO* (Figure 1.4a), *AIDA* (Figure 1.4b) and EVA (Figure 1.4c).

### 1.3.1 NEREO

NEREO (Fig. 1.4a) is the first ROV prototype designed and developed by Team PoliTOcean. It is equipped with eight thrusters, a 2-DOF manipulator with a shoulder, a rotating wrist and a nipper. All the elecrtonics (power and control) fits in the main tube and the software is based on the Robot Operating System (ROS). The power conversion electronics produces a lot of heat and putting it together the control electronics in a cast acrylic plastic tube does not allow sufficient cooling causing overheating. This problem will be solved in AIDA prototype (Sec. 1.3.2) by separating the power conversion electronics from the control electronics, placing it in an aluminum box which in contact with water allows greater heat dissipation. Also ROS-based software will be replaced by an ad-hoc `C/C++` firmware, reducing the resources used on hardware with limited memory and computational power, increasing its performance.



**(a)** NEREO **(b)** AIDA

**(c)** EVA

**Figure 1.4:** ROV prototypes designed and developed by Team PoliTOcean.

### 1.3.2 AIDA

AIDA (Fig. 1.4b) is the second ROV prototype designed and developed during the year 2019. It is equipped with seven thrusters and has a lightweight aluminium frame. With a weight of only 14 kg the ROV allows the operator to lift up to 10 N. This feature allows AIDA to perform the best in recovery missions. Moreover, AIDA is equipped with a mechanical manipulator, powered by three electrical motors actuating the shoulder, the wrist and the claw, and giving it 2 degrees of freedom. Thrusters are configured to have a fully actuated system. Figure 1.5 shows the mechanical drawing of AIDA.

The core of AIDA lives in the *Main Tube*. It consists of a Raspberry Pi communicating via *SPI* with an ATMega micro-controller. The first actuates the manipulator, the second actuates the thrusters and reads the main sensors.

The Raspberry Pi communicates with the control station using *MQTT*, a lightweight, publish-subscribe network protocol that transports messages between devices over TCP/IP. It allows the pilot to control the ROV by sending commands using a Joystick and receive the values of the sensors at diving time. The Raspberry

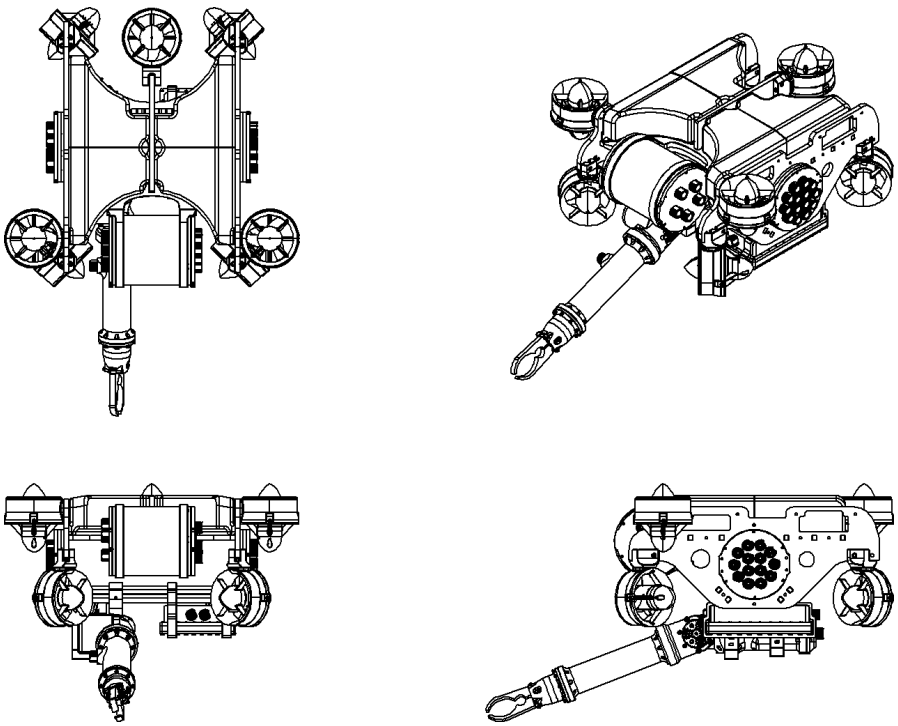


**Figure 1.5:** AIDA draft

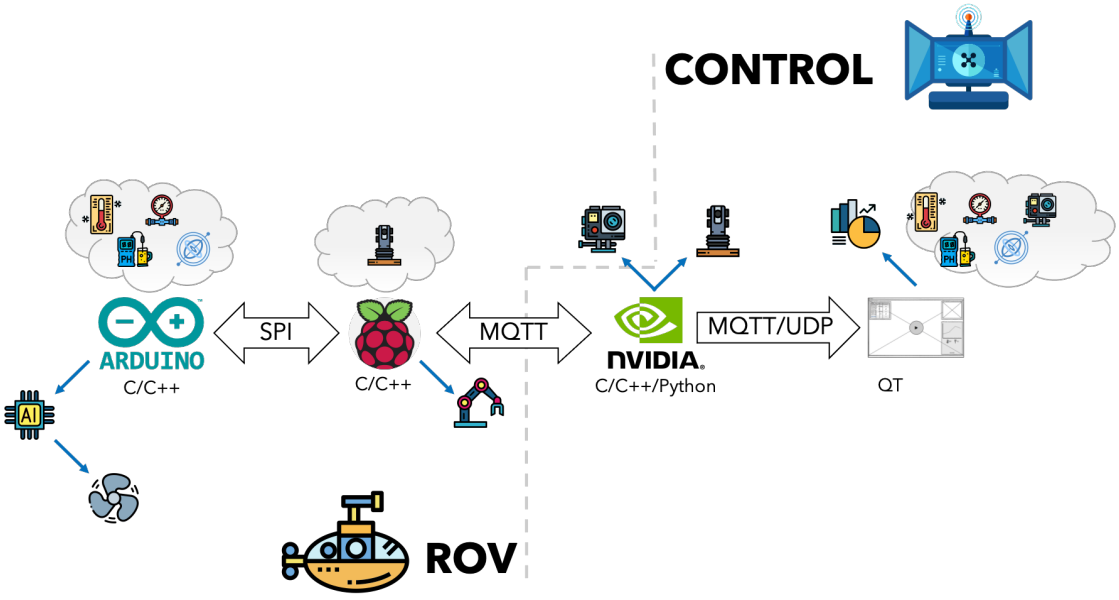



**Figure 1.6:** AIDA Systems Integration Diagram (SID)

Pi receives commands via MQTT from the control station and acts as a dispatcher: it routes to the ATMega commands related to the thrusters.

An aluminum box contains the power conversion electronics, which in contact with the cold sea water guarantees excellent heat dissipation.

The major problem of AIDA is its monolythic architecture with a single point of failure. Furthermore, the communication between the Raspberry and the ATMega micro-controller via SPI is neither trivial nor scalable: it needs a protocol to be designed and implemented for a specific communication. In our case, the Raspberry Pi (acting as the master) sends the reference signal to actuate the thrusters properly, while the ATMega (acting as a slave) sends back the sensors' values to be forwarded to the control station.

To better understand AIDA architecture, a *Systems Integration Diagram* (*SID*) is shown in Figure 1.6.

### 1.3.3 EVA

EVA (Fig. 1.4c) is the last prototype designed and developed by PoliTOcean Team. Its mechanical drawing is reported in Figure 1.7. The cuboid shape and the addition of an extra thruster guarantee a better mechanical stability than the prototype AIDA, compared to which EVA is heavier and bigger. In fact, the upper-back thruster of AIDA generated an unbalanced torque leading to a drift in the direction of motion. The additional thruster of EVA balances the torque and avoids to take the decoupling into account.

Moreover, EVA is equipped with a 9-axis IMU so, thanks to the magnetometer, we can control the yaw angle.

EVA is characterized by a decentralized and modular architecture, more expensive than the previous one, both in terms of budget and production time, but easier to maintain, debug and implement. Each device has its dedicated PCB with its own micro-controller, communicating each other over Ethernet, via MQTT. Code is smaller and updates on single device avoid to re-compile the whole source. Each PCB mounts an ESP-01, which allows to upload the firmware over Wi-Fi separately for every micro-controller.

To better understand EVA architecture, a *Systems Integration Diagram* (*SID*) is shown in Figure 1.8.

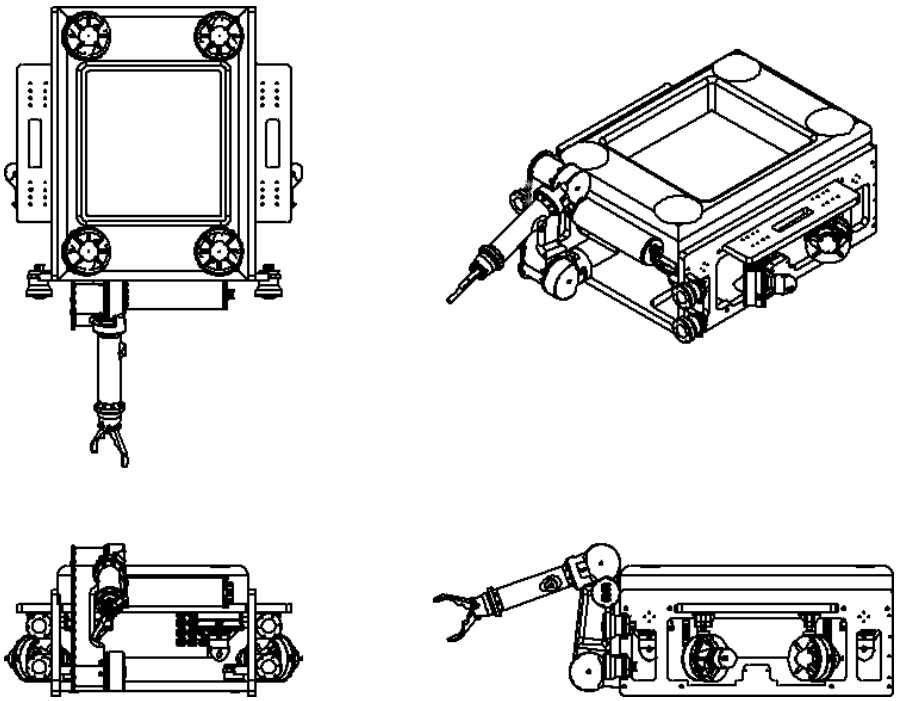


**Figure 1.7:** EVA draft

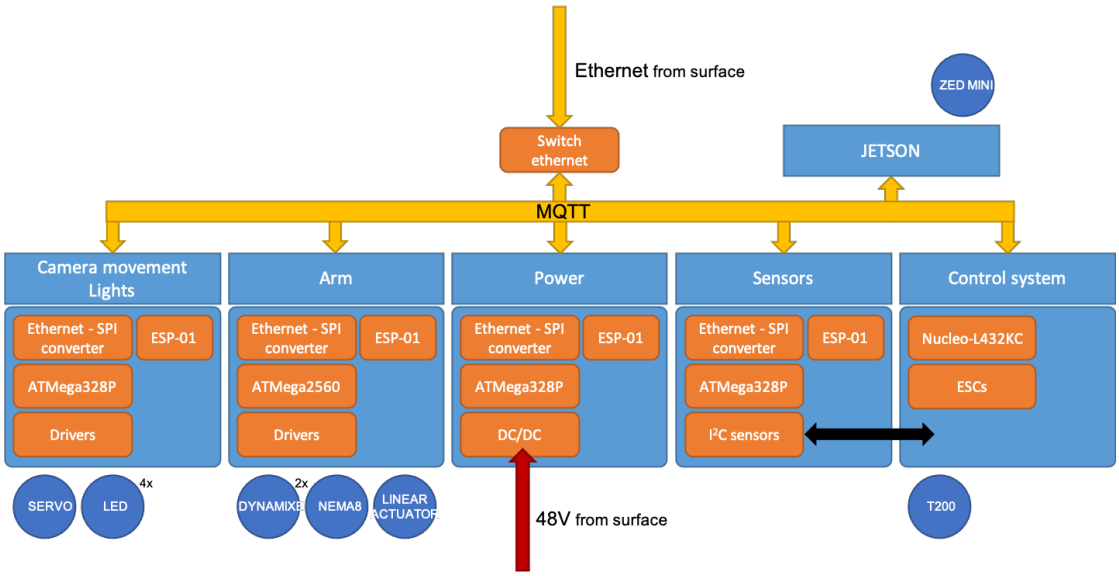



**Figure 1.8:** EVA System Integration Diagram (SID)

## 1.4 MATE ROV Competition

The MATE ROV Competition is an international underwater robotics challenge that engages students from all over the world to apply science, technology, engineering to solve real-world problems that impact the whole world, such as:

- contaminants in our rivers, lakes, waterways, and ocean
- climate change raising ocean temperatures, affecting the health of coral reef

following the invitation from the United Nations Decade of Ocean Science for Sustainable Development (2021-2030) to the global ocean community to plan in ocean science and technology towards creating improved conditions for sustainable development of the Ocean.

Indeed, the mission of the competition is to inspire young students to strength critical thinking, entrepreneurship, innovation and collaboration and embrace efforts to create a sustainable future for our oceans, channeling their skills towards goals such as fighting climate change, providing clean energy, monitoring ocean health.

On this basis, the MATE ROV Competition requires students to engineer ROVs to complete a set of underwater mission tasks, aligned with one or more the United Nation Sustainable Development Goals and, in addition to their ROVs, the student teams have also to prepare technical reports and deliver engineering presentations.

Some of the most important tasks the team faced during the 2021 edition of the MATE ROV Competition are briefly discussed below, as an example.

### 1.4.1 Flying a transect line over a coral reef

The task was to create a software that allowed ROV to autonomously fly a transect line over the coral reef. Successfully flying the transect line autonomously was defined as the control program moving the vehicle from one end of the transect to the other without any input from members. Two red lines are placed 50 cm far from both the long sides of the grid (Fig. 1.9). We succeed in the task if, during the fly, the ROV camera does not show the red lines.

To solve this task, some solutions for self-driving cars can be applied and, in particular, we implemented a CNN architecture with five convolutional layers previously developed by *NVIDIA* with *DARPA* [9], using *PyTorch* library. Since this method is used to steer the wheel of a car, its output is a single numeric value. Because our control system received a reference vector of the form $\mathbf{r} = \begin{bmatrix} x & y & z & roll & pitch & yaw \end{bmatrix}$ as command input, we adapted the network output to the reference signal, controlling y and yaw variables.

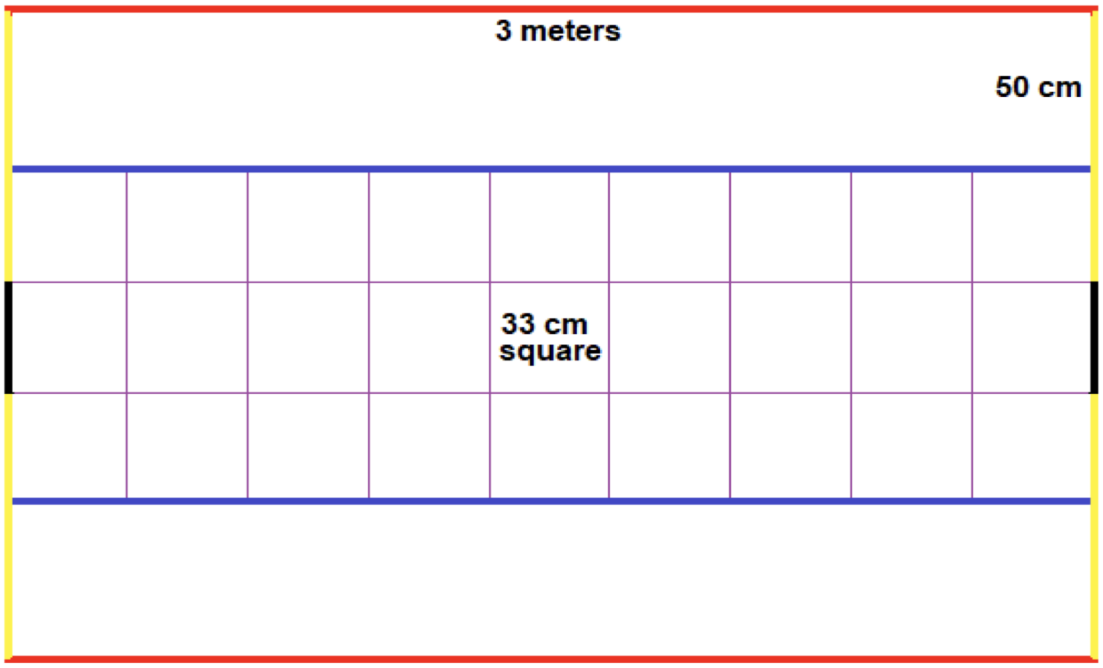
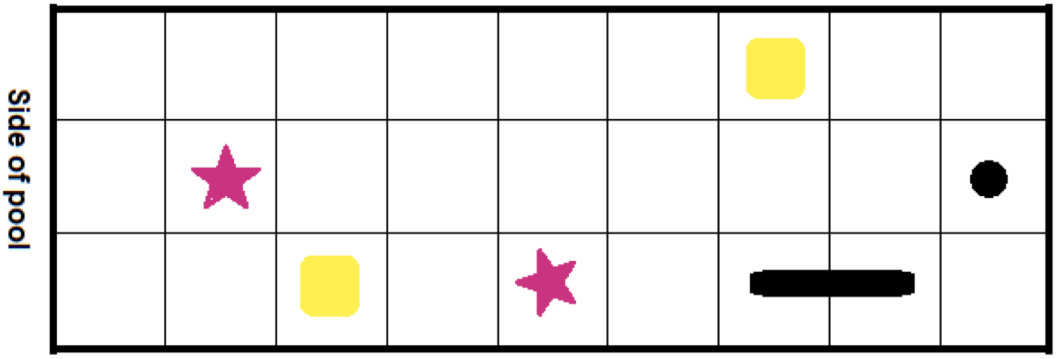



**Figure 1.9:** A diagram of the coral reef. The blue, red, yellow, and black lines are painted 1/2-inch PVC pipes. The purple lines are braided Mason's Line.

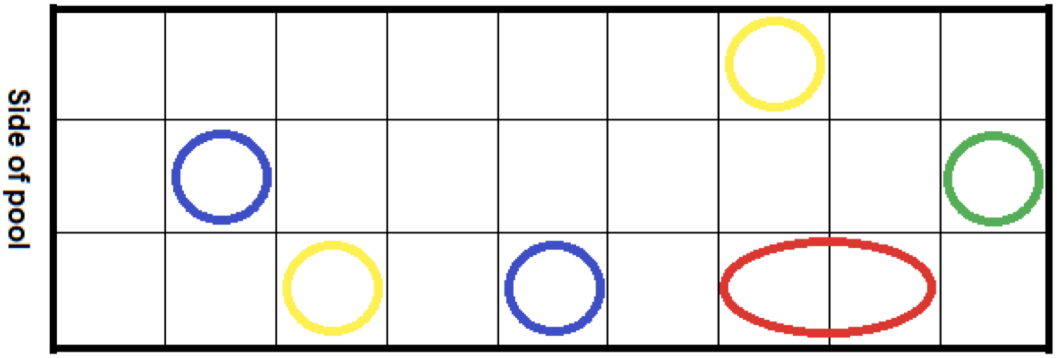
### 1.4.2 Mapping points of interest

Six points of interest located within the coral reef had to be mapped on a grid system. The intent for *autonomous mapping* was to develop a software program that used a photo/video screenshot image of the actual points of interests on the reef then created a map of grid squares and placed those points of interest in the appropriate grid squares autonomously. Successfully mapping the locations was defined as all circles appearing in the proper grid squares with no input from the team.

To autonomously map points of interest, we needed a bird-eye view of the grid. Because the view angle of the camera on board was limited, we approached to this problem creating a panoramic photo of the grid. To accomplish that, we searched for image-stitching algorithms. In particular, we used an Homography-based algorithm. Then, we used the **YOLO** (*You Only Look Once*) [10] object-detection system to locate objects in the grid, map their positions, and classify them.




**(a)** Reef with coral colony, two locations for outplanting coral fragments, two sea stars, and a sponge



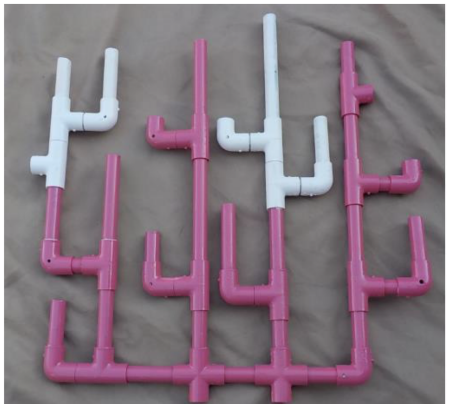

**(b)** Colored circle overlays in the proper squares to match the organisms on the reef

**Figure 1.10:** Mapping process.

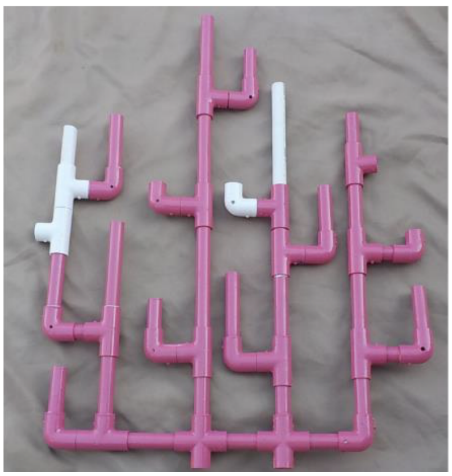
### 1.4.3 Coral Relief Detection

The task was to develop image recognition software to determine the health of the coral colony by comparing the current image (Fig. 1.11b) with an image taken a year earlier (Fig. 1.11a) and to identify and visualize areas of change on the coral colony (Fig. 1.11c).

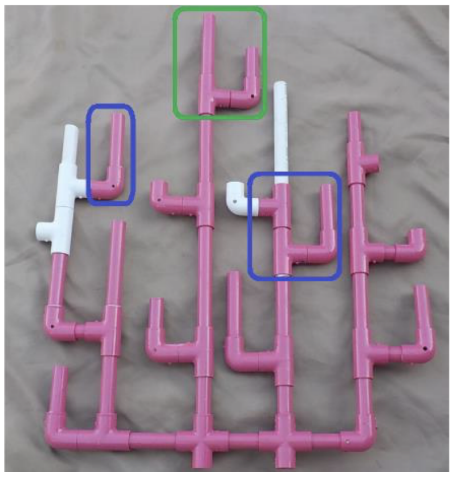
- Areas of growth should be outlined with a green overlay or marked with a green rectangle/circle around the affected area
- Areas of damage or death should be outlined with a yellow overlay or marked with a yellow rectangle/circle around the affected area
- Areas of bleaching/blotching should be outlined with a red overlay or marked with a red rectangle/circle around the affected area
- Areas that have recovered from bleaching/blotching should be outlined with a blue overlay or marked with a blue rectangle/circle around the affected area



**(a)** The coral colony photo from 1 year prior



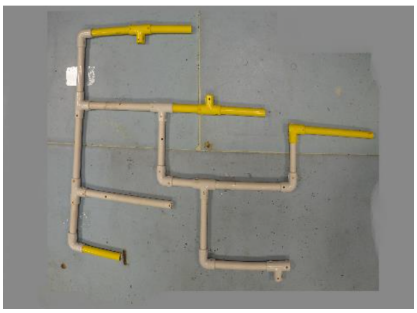
**(b)** The coral colony as it is today



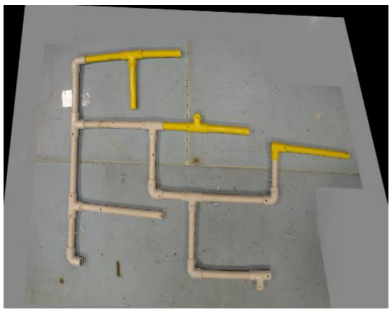
**(c)** The coral colony with all areas of change identified

**Figure 1.11:** Determine the health of a coral colony by comparing its current condition to past data

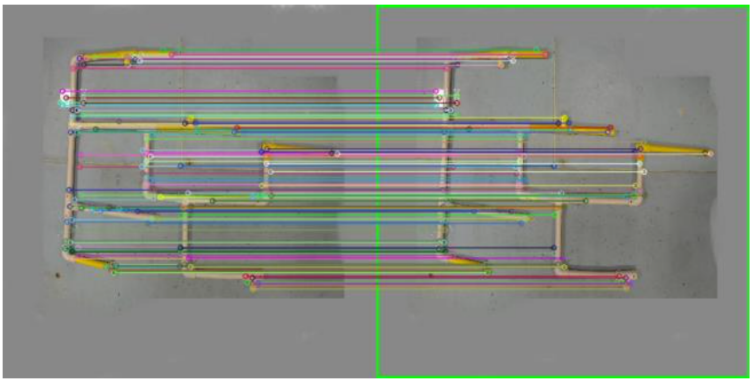
We worked on this task using a comparison method based on the image subtraction technique [11]. This technique requires images to be perfectly aligned. To overcome this issue, we removed the inclination difference between images and, the, the differences in the zooming. Finally, we aligned images both horizontally and vertically by using a Homography matrix generated by the *RANSAC* algorithm (Fig. 1.12). After this pre-processing step, we subtracted images and located the health of the coral reliefs by using object detection algorithms (Fig. 1.13).



**(a)** The coral colony photo from 1 year prior



**(b)** The coral colony as it is today



**(c)** Result of the RANSAC algorithm for matching process

**Figure 1.12:** Result of pre-processing to get *perfectly* aligned images for the matching process

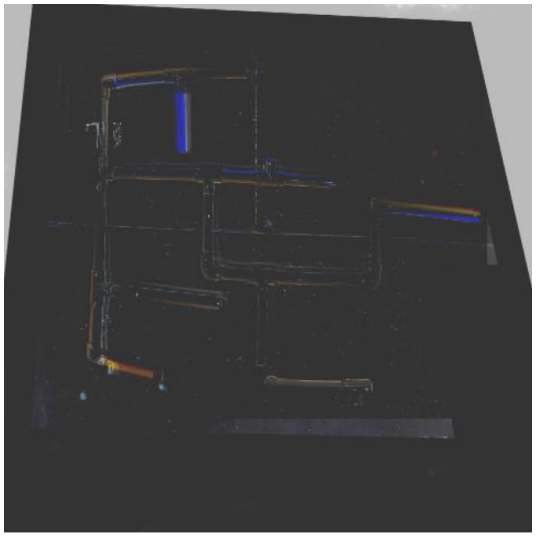


**Figure 1.13:** Result of the subtraction process

## 1.5 Objectives of this thesis

Underwater Vehicle Dynamics are multivariable, non-linear and strongly coupled due to many factors such as hydrostatic and hydrodynamic forces acting on the vehicle. For these reasons, an advanced control technique must be used in ROV for stable cruising, maintaining altitude and direction and so on.

This thesis is focused on the development of a control system that will allow pilot to control the speed of the prototype ROV *EVA* and to stabilize its altitude during precision operations. Indeed, the development of a good mathematical model of an ROV is very important for design of a good performance controller. For this reason, a 6 degree-of-freedom (6-DOF) mathematical model, suitable to describe the ROV, will be carried out and linearized. Then, a model-based control system will be developed and shown to work in simulations.

Two different control techniques (*PID* and $LQI/LQR$) will be used and controller comparison will be carried out based on non-linear model simulations.

The ROV equipped with the developed control system will be tested at the MATE ROV Competition 2022.

# Chapter 2

# Modeling

In this chapter an overview of fundamental theories of 6-DOF mathematical modeling of ROVs is presented, following the work of [12, 13].

Indeed, the determination of the position and the orientation of a rigid body requires 6 independent coordinates: the first three coordinates and their time derivatives describe the position and translational motion along the $x$-, $y$- and $z$-axes, while the last three coordinates and time derivatives describe orientation and rotational motion. The 6 different motion components are generally defined as: *surge*, *sway*, *heave*, *roll*, *pitch* and *yaw*, according to the *SNAME* notation commonly used in marine vehicles (Tab. 2.1).

| DOF | | forces and moments | linear and angular vel. | positions and Euler angles |
|---|---|---|---|---|
| 1 | motions in the $x$-direction (*surge*) | $X$ | $u$ | $x$ |
| 2 | motions in the $y$-direction (*sway*) | $Y$ | $v$ | $y$ |
| 3 | motions in the $z$-direction (*heave*) | $Z$ | $w$ | $z$ |
| 4 | rotation in the $x$-axis (*roll*) | $K$ | $p$ | $\phi$ |
| 5 | rotation in the $y$-axis (*pitch*) | $M$ | $q$ | $\theta$ |
| 6 | rotation in the $z$-axis (*yaw*) | $N$ | $r$ | $\psi$ |

**Table 2.1:** SNAME (1950) notation

## 2.1 Kinematics

Two reference frames are conveniently defined to describe the motion of a 6-DOF rigid-body, as shown in Fig. 2.1. The body-fixed frame ($X_0Y_0Z_0$) is attached to

and moves with the vehicle. The origin $O$ of the reference frame usually coincides with the center of gravity ($CG$). The motion of the body-fixed frame is described in relation to the earth-fixed frame ($XYZ$). The earth-fixed frame can be considered as an inertial frame because the accelerations of a point on the surface of the Earth can be neglected due to the fact that low speed marine vehicles, such as ROVs, are almost unaffected by the motion of the Earth.

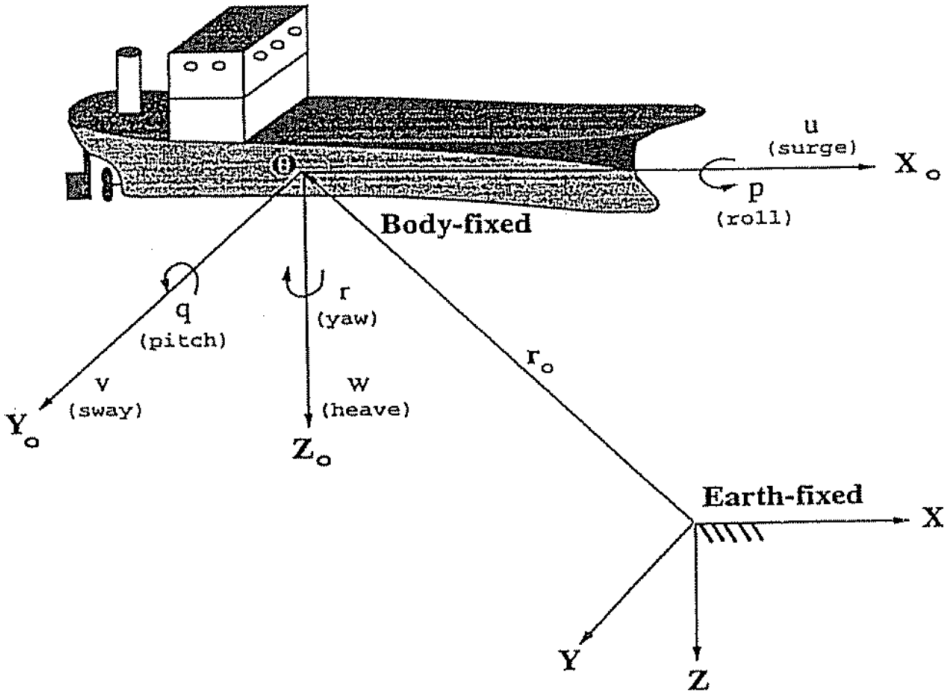



**Figure 2.1:** Body-fixed and earth-fixed reference frames

Based on the *SNAME* notation reported in Tab. 2.1, the general motion of a ROV in 6-DOF can be described by the following vectors:

$$\boldsymbol{\eta} = \left[\boldsymbol{\eta_1}^T, \boldsymbol{\eta_2}^T\right]^T \qquad \boldsymbol{\eta_1} = [x, y, z]^T \qquad \boldsymbol{\eta_2} = [\phi, \theta, \psi]^T$$
$$\boldsymbol{\nu} = \left[\boldsymbol{\nu_1}^T, \boldsymbol{\nu_2}^T\right]^T \qquad \boldsymbol{\nu_1} = [u, v, w]^T \qquad \boldsymbol{\nu_2} = [p, q, r]^T$$
$$\boldsymbol{\tau} = \left[\boldsymbol{\tau_1}^T, \boldsymbol{\tau_2}^T\right]^T \qquad \boldsymbol{\tau_1} = [X, Y, Z]^T \qquad \boldsymbol{\tau_2} = [K, M, N]^T$$

where $\eta$ represents the position and orientation vector with coordinates in the earth-fixed frame, $\nu$ represents the linear and angular velocity vector with coordinates in the body-fixed frame and $\tau$ describes the forces and moments acting on the vehicle in the body-fixed frame.

**Linear velocity transformation.** The transformation relating the linear velocity vector in the inertial reference frame to the velocity in the body-fixed reference frame can be expressed by the following equation:

$$\dot{\boldsymbol{\eta}}_{\mathbf{1}} = \boldsymbol{J}_{\mathbf{1}}(\boldsymbol{\eta}_{\mathbf{2}})\boldsymbol{\nu}_{\mathbf{1}} \tag{2.1}$$

where $\boldsymbol{J}_{\mathbf{1}}(\boldsymbol{\eta}_{\mathbf{2}})$ is described by

- a rotation of a *yaw* angle $\psi$ about $Z$ axis
- a rotation of a *pitch* angle $\theta$ about $Y^{'}$ axis
- a rotation of a *roll* angle $\phi$ about $X^{''}$ axis

where $Y^{'}$ and $X^{''}$ are the new axes obtained from each rotation.

$$\begin{aligned} \boldsymbol{J}_{\mathbf{1}}(\boldsymbol{\eta}_{\mathbf{2}}) &= \boldsymbol{C}^{\boldsymbol{T}}_{\boldsymbol{x},\boldsymbol{\psi}}\boldsymbol{C}^{\boldsymbol{T}}_{\boldsymbol{y},\boldsymbol{\theta}}\boldsymbol{C}^{\boldsymbol{T}}_{\boldsymbol{z},\boldsymbol{\phi}} \\ &= \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \end{aligned} \tag{2.2}$$

**Angular velocity transformation** The body-fixed angular velocity vector $\boldsymbol{\nu}_{\mathbf{2}} = [p, q, r]^T$ and the Euler rate vector $\dot{\boldsymbol{\eta}}_{\mathbf{2}} = [\dot{\phi}, \dot{\theta}, \dot{\psi}]^T$ are related through a transformation matrix $\boldsymbol{J}_{\mathbf{2}}(\boldsymbol{\eta}_{\mathbf{2}})$ according to:

$$\dot{\boldsymbol{\eta}}_{\mathbf{2}} = \boldsymbol{J}_{\mathbf{2}}(\boldsymbol{\eta}_{\mathbf{2}})\boldsymbol{\nu}_{\mathbf{2}} \tag{2.3}$$

where

$$\boldsymbol{J}_{\mathbf{2}}(\boldsymbol{\eta}_{\mathbf{2}}) = \begin{bmatrix} 1 & 0 & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} \end{bmatrix} \tag{2.4}$$

It is noteworthy that actual angular coordinates cannot be obtained by directly integrating the angular body velocity vector $\boldsymbol{\nu}_{\mathbf{2}} = [p, q, r]^T$. In fact, the resulting integral does not have any immediate physical interpretation and, moreover, $\boldsymbol{J}_{\mathbf{2}}(\boldsymbol{\eta}_{\mathbf{2}})$ is undefined for a pitch angle of $\theta = \pm 90°$. However, this does not represent a problem as underwater vehicles may operate close to this singularity.

Combining 2.1 and 2.3, they can be expressed in vector form as:

$$\begin{bmatrix} \dot{\boldsymbol{\eta}}_{\mathbf{1}} \\ \dot{\boldsymbol{\eta}}_{\mathbf{2}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_{\mathbf{1}}(\boldsymbol{\eta}_{\mathbf{2}}) & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \boldsymbol{J}_{\mathbf{2}}(\boldsymbol{\eta}_{\mathbf{2}}) \end{bmatrix} \begin{bmatrix} \boldsymbol{\nu}_{\mathbf{1}} \\ \boldsymbol{\nu}_{\mathbf{2}} \end{bmatrix} \quad \Longleftrightarrow \quad \dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{2.5}$$

## 2.2 Rigid-body Dynamics

Considering a body-fixed coordinate system $X_0Y_0Z_0$ which rotates with an angular velocity $\boldsymbol{\omega} = [\omega_1, \omega_2, \omega_3]^T$ about an earth-fixed coordinate system $XYZ$, the body's inertia tensor $\boldsymbol{I_0}$ referred to an arbitrary body-fixed coordinate system is described by the following relation:

$$\boldsymbol{I_0} \triangleq \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & -I_{zy} & I_z \end{bmatrix}; \qquad \boldsymbol{I_0} = \boldsymbol{I_0^T} > \boldsymbol{0} \tag{2.6}$$

where $I_x$, $I_y$ and $I_z$ are the moments of inertia about the $X_0$, $Y_0$ and $Z_0$ aces and $I_{xy} = I_{yx}$, $I_{xz} = I_{zx}$ and $I_{yz} = I_{zy}$ are the product of inertia defined as:

$$\begin{aligned} I_x &= \int_V (y^2 + z^2)\rho_A dV; & I_{xy} &= \int_V xy\rho_A dV = \int_V yx\rho_A dV = I_{yx} \\ I_y &= \int_V (x^2 + z^2)\rho_A dV; & I_{xz} &= \int_V xz\rho_A dV = \int_V zx\rho_A dV = I_{zx} \\ I_z &= \int_V (x^2 + y^2)\rho_A dV; & I_{yz} &= \int_V yz\rho_A dV = \int_V zy\rho_A dV = I_{zy} \end{aligned}$$

with $\rho_A$ as the mass density of the body. Furthermore the mass of the body is defined as:

$$m = \int_V \rho_A dV \tag{2.7}$$

and because it is assumed to be constant in time ($\dot{m} = 0$), the vehicle's center of gravity can be defined as:

$$\boldsymbol{r_G} = \frac{1}{m} \int_V \boldsymbol{r}\rho_A dV \tag{2.8}$$

By applying the *Newtonian* and *Lagrangian* formalism, a vectorial representation of the 6-DOF rigid-body equations of motion can be obtained and expressed in a compact form as:

$$\boldsymbol{M_{RB}\dot{\nu}} + \boldsymbol{C_{RB}(\nu)\nu} = \boldsymbol{\tau_{RB}} \tag{2.9}$$

While the parametrization of the rigid-body inertia matrix $\boldsymbol{M_{RB}}$ is unique and satisfies

$$\boldsymbol{M_{RB}} = \boldsymbol{M_{RB}^T} > 0; \qquad \boldsymbol{\dot{M}_{RB}} = \boldsymbol{0}$$

where

$$\boldsymbol{M_{RB}} = \begin{bmatrix} m\boldsymbol{I}_{3x3} & -m\boldsymbol{S}(\boldsymbol{r_G}) \\ m\boldsymbol{S}(\boldsymbol{r_G}) & \boldsymbol{I_0} \end{bmatrix} \tag{2.10}$$

a large number of parameterizations for the $\boldsymbol{C_{RB}}$ matrix, which consists of the Coriolis vector term $\boldsymbol{\omega} \times \boldsymbol{\nu}$ and the centripetal vector term $\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \boldsymbol{r_G})$, can

be found. An useful skew-symmetric representation can be derived from this expression:

$$\boldsymbol{C_{RB}}(\boldsymbol{\nu}) = \begin{bmatrix} \mathbf{0}_{3x3} & -m\boldsymbol{S}(\boldsymbol{\nu_1}) - m\boldsymbol{S}(\boldsymbol{\nu_2})\boldsymbol{S}(\boldsymbol{r_G}) \\ -m\boldsymbol{S}(\boldsymbol{\nu_1}) + m\boldsymbol{S}(\boldsymbol{r_G})\boldsymbol{S}(\boldsymbol{\nu_2}) & -\boldsymbol{S}(\boldsymbol{I_0}\boldsymbol{\nu_2}) \end{bmatrix} \tag{2.11}$$

To simplify the general rigid-body equations of motion the origin of the body-fixed coordinate system frame is chosen to match with the center of gravity, so that $\boldsymbol{r_G} = \mathbf{0}$ and consequently

$$\boldsymbol{M_{RB}} = \begin{bmatrix} m\boldsymbol{I}_{3x3} & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \boldsymbol{I_0} \end{bmatrix}$$

$$\boldsymbol{C_{RB}}(\boldsymbol{\nu}) = \begin{bmatrix} \mathbf{0}_{3x3} & -m\boldsymbol{S}(\boldsymbol{\nu_1}) \\ -m\boldsymbol{S}(\boldsymbol{\nu_1}) & -\boldsymbol{S}(\boldsymbol{I_0}\boldsymbol{\nu_2}) \end{bmatrix}$$

## 2.3 Hydrodynamic Forces and Moments

The right-hand side vector term of (2.9) represents the external forces and moments acting on the vehicle. These forces can be classified according to:

- *Radiation-induced forces*
  - added inertia
  - hydrodynamic damping
  - restoring forces
- *Environmental forces*
  - ocean currents
  - waves
  - wind
- *Propulsion forces*
  - thruster/propeller forces
  - control surfaces/rudder forces

Thus, the non-linear model representation of the dynamic equations of motion can be described by

$$\boldsymbol{M_{RB}}\dot{\boldsymbol{\nu}} + \boldsymbol{C_{RB}}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau_{RB}} \tag{2.12}$$

$$\boldsymbol{\tau_{RB}} = \boldsymbol{\tau_H} + \boldsymbol{\tau_E} + \boldsymbol{\tau} \tag{2.13}$$

where $\boldsymbol{\tau_H}$ represents the radiation-induced forces and moments and is expressed by

$$\boldsymbol{\tau_H} = -\underbrace{\boldsymbol{M_A}\dot{\boldsymbol{\nu}} - \boldsymbol{C_A}(\boldsymbol{\nu})\boldsymbol{\nu}}_{\text{added mass}} - \underbrace{\boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu}}_{\text{damping}} - \underbrace{\boldsymbol{g}(\boldsymbol{\eta})}_{\text{restoring forces}} \tag{2.14}$$

$\boldsymbol{\tau_E}$ describes the environmental forces and moments acting on the vehicle and $\boldsymbol{\tau}$ represents the propulsion forces and moments. The following 6-DOF dynamic equations of motion can be derived by substituting (2.13) into (2.12) together with (2.14):

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau_E} + \boldsymbol{\tau} \tag{2.15}$$

where

$$\boldsymbol{M} \triangleq \boldsymbol{M_{RB}} + \boldsymbol{M_A}; \qquad \boldsymbol{C}(\boldsymbol{\nu}) \triangleq \boldsymbol{C_{RB}}(\boldsymbol{\nu}) + \boldsymbol{C_A}(\boldsymbol{\nu})$$

In the following sections the added mass (Sec. 2.3.1), the hydrodynamics damping (Sec. 2.3.2) and the restoring forces and moments (Sec. 2.3.3), which appears in (2.14), are derived.

### 2.3.1 Added Mass and Inertia

When a rigid-body is moving in a fluid, the additional inertia of the fluid surrounding the body, that is accelerated by the movement of the body, has to take into consideration. While this effect is negligible in industrial robotics since the density of the air is much lighter than the density of a mechanical system which is moving, in underwater applications the density of water ($\rho \approx 1.000\,\mathrm{kg/m^3}$) is comparable with the density of the ROV and the effect must be considered.

The added mass is not a quantity of fluid to add to the system, resulting in an increased mass, but it is a reaction force the fluid exerts to accelerate with the body itself.

The hydrodynamic force $X_A$ along the $x$-axis due to an acceleration $\dot{u}$ in the $x-$direction is expressed by:

$$X_A = -X_{\dot{u}}\dot{u} \qquad \text{where} \qquad X_{\dot{u}} = \frac{\partial X}{\partial \dot{u}} \tag{2.16}$$

where $\boldsymbol{M_A} > \boldsymbol{0}$ for completely submerged bodies. Moreover, if the fluid is ideal, the body's velocity is low, there are no currents or waves and frequency independence it holds:

$$\boldsymbol{M_A} = \boldsymbol{M_A^T} > \boldsymbol{0} \tag{2.17}$$

The added mass has also an added Coriolis and centripetal contribution. It can be demonstrated that the matrix expression can always be parametrized such that:

$$\boldsymbol{C_A}(\boldsymbol{\nu}) = -\boldsymbol{C_A^T}(\boldsymbol{\nu}) \tag{2.18}$$

In many ROV applications, the vehicle is allowed to move at low speed and, if it has three planes of symmetry, the following structure of matrices $M_A$ and $C_A$ can therefore be considered:

$$\boldsymbol{M_A} = -\text{diag}\{X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}, K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}\} \tag{2.19}$$

$$\boldsymbol{C_A}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & 0 & 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v \\ 0 & 0 & 0 & Z_{\dot{w}}w & 0 & -X_{\dot{u}}u \\ 0 & 0 & 0 & -Y_{\dot{v}}v & X_{\dot{u}}u & 0 \\ 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v & 0 & -N_{\dot{r}}r & M_{\dot{q}}q \\ Z_{\dot{w}}w & 0 & -X_{\dot{u}}u & N_{\dot{r}}r & 0 & -K_{\dot{p}}p \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 & -M_{\dot{q}}q & K_{\dot{p}}p & 0 \end{bmatrix} \tag{2.20}$$

### 2.3.2 Hydrodynamic Damping

The viscosity of the fluid is also responsible of dissipative drag and lift forces on the body. Both forces are supposed to act on the center of mass of the body. The drag forces are parallel to the relative velocity between the body and the fluid and, for a spherical body moving in a fluid, can be modeled as:

$$F_{\text{drag}} = \frac{1}{2}\rho C_d(R_n)A|U|U \tag{2.21}$$

where $\rho$ is the fluid density, $U$ is the velocity of the sphere, $A$ is the frontal area of the sphere, $C_d$ is the adimensional drag coefficient and $R_n$ is the Reynolds number defined as

$$R_n = \frac{UD}{\mu}$$

where $D$ is the characteristic dimension of the body perpendicular to the direction of $U$ and $\mu$ is the kinematic viscosity coefficient of the fluid.

The lift forces are perpendicular to the flow direction. For an hydrofoil, these forces can be modeled as:

$$F_{\text{lift}} = \frac{1}{2}\rho C_l(R_n, \alpha)A|U|U \tag{2.22}$$

where $C_l$ is the adimensional lift coefficient. Notice that it also depends on the angle of attack (AOA) $\alpha$.

### 2.3.3 Restoring Forces and Moments

Restoring forces are the gravitational and buoyant forces, according to the hydrodynamic terminology. The gravitational force $\boldsymbol{f_G}$ acts through the center of gravity

$\boldsymbol{r_G} = [x_G, y_G, z_G]^T$ of the vehicle and the buoyant force $\boldsymbol{f_B}$ acts through the center of buoyancy $\boldsymbol{r_B} = [x_B, y_B, z_B]^T$.

Defining $m$ as the mass of the vehicle, $\Delta$ as the volume of the fluid displaced by the vehicle, $g$ as the acceleration of gravity (positive downwards) and $\rho$ as the fluid density, the submerged weight of the body is defined as $W = mg$, according to the *SNAME* notation, while the buoyancy force is defined as $B = \rho g \Delta$. By appling the results from Sec. 2.1, the weight and buoyancy forces transformed to the body-fixed coordinate system are defined as:

$$\boldsymbol{f_G}(\boldsymbol{\eta_2}) = \boldsymbol{J_1}^{-1}(\boldsymbol{\eta_2}) \begin{bmatrix} 0 \\ 0 \\ W \end{bmatrix} \qquad \boldsymbol{f_B}(\boldsymbol{\eta_2}) = -\boldsymbol{J_1}^{-1}(\boldsymbol{\eta_2}) \begin{bmatrix} 0 \\ 0 \\ B \end{bmatrix} \tag{2.23}$$

Consequently, the restoring force and moment vector in the body-fixed coordinate system is defined as:

$$\boldsymbol{g}(\boldsymbol{\eta}) = -\begin{bmatrix} \boldsymbol{f_G}(\boldsymbol{\eta}) + \boldsymbol{f_B}(\boldsymbol{\eta}) \\ \boldsymbol{r_G} \times \boldsymbol{f_G}(\boldsymbol{\eta}) + \boldsymbol{r_B} \times \boldsymbol{f_B}(\boldsymbol{\eta}) \end{bmatrix} \tag{2.24}$$

Notice that the $z$-axis is taken to be positive downwards. Expanding this expression yields:

$$\boldsymbol{g}(\boldsymbol{\eta}) = \begin{bmatrix} & (W-B)s\theta \\ - & (W-B)c\theta s\phi \\ - & (W-B)c\theta c\phi \\ - & (y_G W - y_B B)c\theta c\phi + (z_G W - z_B B)c\theta s\phi \\ - & (z_G W - z_B B)s\theta + (x_G W - x_B B)c\theta c\phi \\ - & (x_G W - x_B B)c\theta s\phi + (y_G W - y_B B)s\theta \end{bmatrix} \tag{2.25}$$

## 2.4 Equations of motion

The equations of motion can be represented in both the body-fixed and earth-fixed coordinate systems.

### 2.4.1 Body-fixed vector representation

The nonlinear equations of motion in the body-fixed frame can be written as:

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\nu}) = \boldsymbol{\tau} \tag{2.26}$$

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{2.27}$$

where

$$\boldsymbol{M} = \boldsymbol{M_{RB}} + \boldsymbol{M_A} \qquad \boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C_{RB}}(\boldsymbol{\nu}) + \boldsymbol{C_A}(\boldsymbol{\nu}) \tag{2.28}$$

### 2.4.2 Earth-fixed vector representation

The earth-fixed representation is obtained by applying the following kinematic transformations (assuming that $\boldsymbol{J}(\boldsymbol{\eta})$ is non-singular):

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \quad &\Longleftrightarrow \quad \boldsymbol{\nu} &= \boldsymbol{J}^{-1}(\boldsymbol{\eta})\dot{\boldsymbol{\eta}} \\ \ddot{\boldsymbol{\eta}} &= \boldsymbol{J}(\boldsymbol{\eta})\dot{\boldsymbol{\nu}} + \dot{\boldsymbol{J}}(\boldsymbol{\eta})\boldsymbol{\nu} \quad &\Longleftrightarrow \quad \dot{\boldsymbol{\nu}} &= \boldsymbol{J}^{-1}(\boldsymbol{\eta})\dot{\boldsymbol{\eta}} \left[\ddot{\boldsymbol{\eta}} - \dot{\boldsymbol{J}}(\boldsymbol{\nu})\boldsymbol{J}^{-1}(\boldsymbol{\eta})\dot{\boldsymbol{\eta}}\right] \end{aligned} \tag{2.29}$$

to eliminate $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$ from (2.26). Defining:

$$\begin{aligned} \boldsymbol{M}_{\eta}(\boldsymbol{\eta}) &= \boldsymbol{J}^{-\boldsymbol{T}}(\boldsymbol{\eta})\boldsymbol{M}\boldsymbol{J}^{-1}(\boldsymbol{\eta}) \\ \boldsymbol{C}_{\eta}(\boldsymbol{\nu},\boldsymbol{\eta}) &= \boldsymbol{J}^{-\boldsymbol{T}}(\boldsymbol{\eta})\left[\boldsymbol{C}(\boldsymbol{\nu}) - \boldsymbol{M}\boldsymbol{J}^{-1}(\boldsymbol{\eta})\dot{\boldsymbol{J}}(\boldsymbol{\eta})\right]\boldsymbol{J}^{-1}(\boldsymbol{\eta}) \\ \boldsymbol{D}_{\eta}(\boldsymbol{\nu},\boldsymbol{\eta}) &= \boldsymbol{J}^{-\boldsymbol{T}}(\boldsymbol{\eta})\boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{J}^{-1}(\boldsymbol{\eta}) \\ \boldsymbol{g}_{\eta}(\boldsymbol{\eta}) &= \boldsymbol{J}^{-\boldsymbol{T}}(\boldsymbol{\eta})\boldsymbol{g}(\boldsymbol{\eta}) \\ \boldsymbol{\tau}_{\eta}(\boldsymbol{\eta}) &= \boldsymbol{J}^{-\boldsymbol{T}}(\boldsymbol{\eta})\boldsymbol{\tau} \end{aligned} \tag{2.30}$$

yields the earth-fixed vector representation:

$$\boldsymbol{M}_{\eta}(\boldsymbol{\eta})\ddot{\boldsymbol{\eta}} + \boldsymbol{C}_{\eta}(\boldsymbol{\nu},\boldsymbol{\eta})\dot{\boldsymbol{\eta}} + \boldsymbol{D}_{\eta}(\boldsymbol{\nu},\boldsymbol{\eta})\dot{\boldsymbol{\eta}} + \boldsymbol{g}_{\eta}(\boldsymbol{\eta}) = \boldsymbol{\tau}_{\eta} \tag{2.31}$$

# Chapter 3

# Control system

The control system, together with the sensor architecture, plays a crucial role in the quality and level of performance that an ROV can achieve. It allows ROV to perform complex operations with accuracy and, indeed, an excellent control is a fundamental requirement to enable ROV to effectively carry out its mission.

Generally, the control system receives a speed reference command as input and compares it with the values read by the sensors to compute the command for the actuators which, in the case of ROVs, are thrusters. The ROV must be able to move on the horizontal plane without changing its depth, so the controller system consists of two separate controllers working together: a *speed controller* and a *depth controller*. When the pilot interrupts the speed control on the $z$-axis, the depth controller stores the depth read by the pressure sensor and uses it as a reference.

Definitely, the system compares the vehicle's desired position, orientation and velocities with their current estimates and computes the required propulsion forces and moments.

This chapter describes the actuators and sensors mounted on board the ROV and the design and implementation of two distinct linear control techniques for both the speed controller and a more general position controller:

- PID and LQI for the speed controller
- PID and LQR for the position controller

## 3.1 Thrusters

Forces and moments are applied to the system through thruster. They generate a thrust via the rotation of the propellers, which determines the motion of the ROV by the Newton's third law. The rotation of the propellers also generates a moment that must be balanced by a correct arrangement of left- and right-hand propellers.

Starting from forces and moments in the body-fixed frame, it is necessary to know the forces to be assigned to each motor, so that the ROV can move in the desired direction and with the desired thrust.

### 3.1.1 Thruster allocation

In practical applications the vector of propulsion forces and moments $\boldsymbol{\tau}$ acting on the vehicle can be described as a function of the thrust vector $\boldsymbol{u}$ by the following expression [14]:

$$\boldsymbol{\tau} = \boldsymbol{B}(\boldsymbol{\alpha})\boldsymbol{P}\boldsymbol{u} \tag{3.1}$$

where $\boldsymbol{B}$ is the thruster configuration matrix, $\boldsymbol{\alpha}$ is the vector of thrust angles, $\boldsymbol{u}$ is the thrust vector and $\boldsymbol{P}$ is the diagonal matrix of the readiness of the thrusters.

$$p_{ii} = \begin{cases} 0 & \text{if the } i\text{-th thruster is off} \\ 1 & \text{otherwise} \end{cases}$$

$$\boldsymbol{B} = \begin{bmatrix} c\alpha_1 & c\alpha_2 & c\alpha_3 & c\alpha_4 \\ s\alpha_1 & s\alpha_2 & s\alpha_3 & s\alpha_4 \\ 0 & 0 & 0 & 0 \\ -z_1 s\alpha_1 & -z_2 s\alpha_2 & -z_3 s\alpha_3 & -z_4 s\alpha_4 \\ z_1 c\alpha_1 & z_2 c\alpha_2 & z_3 c\alpha_3 & z_4 c\alpha_4 \\ x_1 s\alpha_1 - y_1 c\alpha_1 & x_2 s\alpha_2 - y_2 c\alpha_2 & x_3 s\alpha_3 - y_3 c\alpha_3 & x_4 s\alpha_4 - y_4 c\alpha_4 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ y_5 & y_6 & y_7 & y_8 \\ -x_5 & -x_6 & -x_7 & -x_8 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

where $\boldsymbol{b}_i = [x_i\ y_i\ z_i]$ is the distance vector of the $i$-th thruster from the center of mass, $c\alpha_i = \cos\alpha_i$, $s\alpha_i = \sin\alpha_i$.

Assuming $\boldsymbol{\tau}_d$ such as $\boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max}$, the unconstrained thrust allocation problem can be formulated as the following Least-Squares (LS) optimization problem:

$$\begin{aligned} \min_{u} \quad & \frac{1}{2}\boldsymbol{u}^T \boldsymbol{W} \boldsymbol{u} \\ \text{s.t.} \quad & \boldsymbol{\tau} - \boldsymbol{B}\boldsymbol{u} = \boldsymbol{0} \end{aligned} \tag{3.2}$$

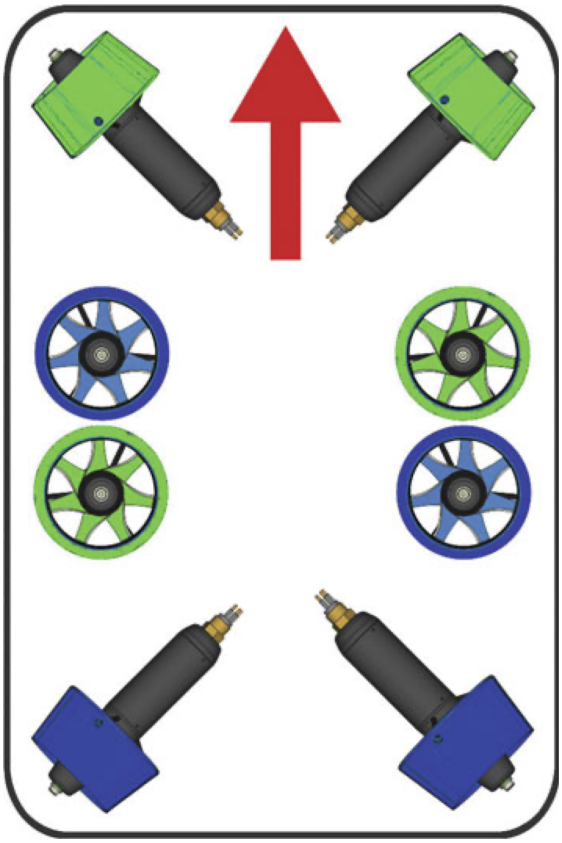


**Figure 3.1:** Thruster allocation for an underwater vehicle with 8 thrusters (blue clockwise, green counter-clockwise).

where $\boldsymbol{W}$ is a positive definitive matrix. For underwater vehicles equipped with both control surfaces and thrusters, the elements of $\boldsymbol{W}$ should be selected such that using the thrusters is more expensive than using the control surfaces, in terms of battery energy.

The solution to (3.2) can be calculated using the Lagrange multipliers, as described in [12] as

$$\boldsymbol{u} = \boldsymbol{B}^{\dagger}\boldsymbol{\tau} \tag{3.3}$$

where

$$\boldsymbol{B}_{W}^{\dagger} = \boldsymbol{W}^{-1}\boldsymbol{B}^{T}(\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^{T})^{-1} \tag{3.4}$$

Considering all inputs are equality weighted, that is $\boldsymbol{W} = \boldsymbol{I}$, (3.4) reduces to the *Moore-Penrose pseudo-inverse* (3.5).

$$\boldsymbol{B}^{\dagger} = \boldsymbol{B}^{T}(\boldsymbol{B}\boldsymbol{B}^{T})^{-1} \tag{3.5}$$

This approach provides a solution only if the thrusters are all available and fully operational, i.e. $\boldsymbol{P} = \boldsymbol{I}$. To increase the availability in case of one non-operational thruster, a solution using Singular Value Decomposition (SVD) is proposed [14].

$$\boldsymbol{B} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{3.6}$$

where

$$\boldsymbol{S} = [\boldsymbol{S}_T \quad \boldsymbol{0}], \qquad \boldsymbol{S}_T = \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_6 \end{bmatrix} \tag{3.7}$$

Considering all the thrusters available and fully operational,

$$\boldsymbol{B}^\dagger = \boldsymbol{V}\boldsymbol{S}^*\boldsymbol{U}^T = \boldsymbol{V}\begin{bmatrix} \boldsymbol{S}_T^{-1} \\ \boldsymbol{0} \end{bmatrix}\boldsymbol{U}^T \tag{3.8}$$

Suppose the $k$-th thruster is not available, i.e. $u_k = 0$ and $p_{kk} = 0$, due to a fault in the propulsion system. Substituting (3.6) into (3.1) leads to

$$\boldsymbol{\tau} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T\boldsymbol{P}\boldsymbol{u} \tag{3.9}$$

Defining

$$\begin{aligned} \boldsymbol{u}' &= [u_1, \ldots, u_{k-1}, u_{k+1}, \ldots, u_n]^T \\ \boldsymbol{V}^* = \boldsymbol{V}^T\boldsymbol{P} &= \left[\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_{k-1}^*, \boldsymbol{0}, \boldsymbol{v}_{k+1}^*, \ldots, \boldsymbol{v}_n^*\right] \\ \boldsymbol{V}_{\boldsymbol{u}}^* &= \left[\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_{k-1}^*, \boldsymbol{v}_{k+1}^*, \ldots, \boldsymbol{v}_n^*\right] \end{aligned}$$

the expression (3.9) can be written as

$$\boldsymbol{\tau} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}_{\boldsymbol{u}}^*\boldsymbol{u}' \tag{3.10}$$

and the vector $\boldsymbol{u}'$ can be computed as

$$\boldsymbol{u}' = \boldsymbol{B}^\dagger\boldsymbol{\tau}, \qquad \boldsymbol{B}^\dagger = \left( (\boldsymbol{S}\boldsymbol{V}_{\boldsymbol{u}}^*)^T \boldsymbol{S}\boldsymbol{V}_{\boldsymbol{u}}^* \right)^{-1} (\boldsymbol{S}\boldsymbol{V}_{\boldsymbol{u}}^*)^T \boldsymbol{U}^T \tag{3.11}$$

Hence, the thrust vector $\boldsymbol{u}$ can be obtained as

$$\boldsymbol{u} = \left[u_1', \ldots, u_{k-1}', 0, u_{k+1}', \ldots, u_n'\right]^T \tag{3.12}$$

### 3.1.2 Thruster dynamics

In [12] a 1st-order approximation of the developed thrust $T$ and torque $Q$ for a single-screw propeller is derived. Let $n$ denote the propeller revolution, $D$ the propeller diameter, $\rho$ the water density and $V_a$ the advance speed at the propeller.

$$T = \rho D^4 K_T(J_0)|n|n \tag{3.13}$$

$$Q = \rho D^5 K_Q(J_0)|n|n \tag{3.14}$$

where $J_0 = V_a/(nD)$ is the *advance number* and $K_T$ and $K_Q$ respectively the *thrust* and *torque coefficients*.

A third-order state-space model is presented and discussed in [15]

$$J_m \dot{n} + K_n n = \tau - Q \tag{3.15}$$

$$m_f \dot{u}_p + d_{f_0} u_p + d_f |u_p|(u_p - u_a) = T \tag{3.16}$$

$$(m - X_{\dot{u}})\dot{u} - X_u u - X_{u|u|} u|u| = (1-t)T \tag{3.17}$$

where $J_m$ is the moment of inertia of the motor, $K_n$ is the linear motor damping coefficient, $\tau$ is the motor control input, $Q$ is the propeller torque, $m_f$ is the mass of water in the propeller control volume, $d_{f_0}$ and $d_f$ are the linear and quadratic damping coefficients, respectively, $u_a$ is the ambient water velocity, $T$ is the propeller thrust, and $t$ is the thrust deduction number. In the case of steady-state motion, i.e., $\dot{u} = 0$, the ambient water velocity $u_a$ is related to the surge by the *wake fraction number* $w$ as

$$u_a = (1 - w)u \tag{3.18}$$

EVA is equipped with eight underwater BlueRobotics T200 thrusters, consisting of a fully flooded brushless motor. To run any three-phase brushless motor, an electronic speed controller (ESC) is necessary. It allows to control and adjust the ROV speed, following a speed reference signal and varying the switching rate of a network of field-effect transistors (FETs). The speed of the motors changes by adjusting the duty cycle. Both the BlueRobotics thruster and ESC are shown in Fig. 3.2.

The constructor does not provide any information about the T200 characteristics which could allow to reconstruct the thruster model or transfer function. Therefore, the thruster dynamics is approximated, in its simplest form, to a first-order transfer function, with time constant $T$

$$\tau = \frac{1}{Ts + 1} \tag{3.19}$$

As no specific values are reported in the datasheet [16], $T = 0{,}1\,\text{s}$ was chosen to achieve the maximum deliverable thrust in about $0{,}5\,\text{s}$. The choice of a first-order

**(a)** T200 thruster **(b)** Basic ESC

**Figure 3.2:** A BlueRobotics T200 thruster (left) with a BlueRobotics Basic ESC (right).

transfer function introduces a delay through the time constant to reach the steady-state value, but the advantage is that $T$ is the only parameter to set, making the model easy to tune. However, it is not possible to observe the overshoot as from a second-order model.

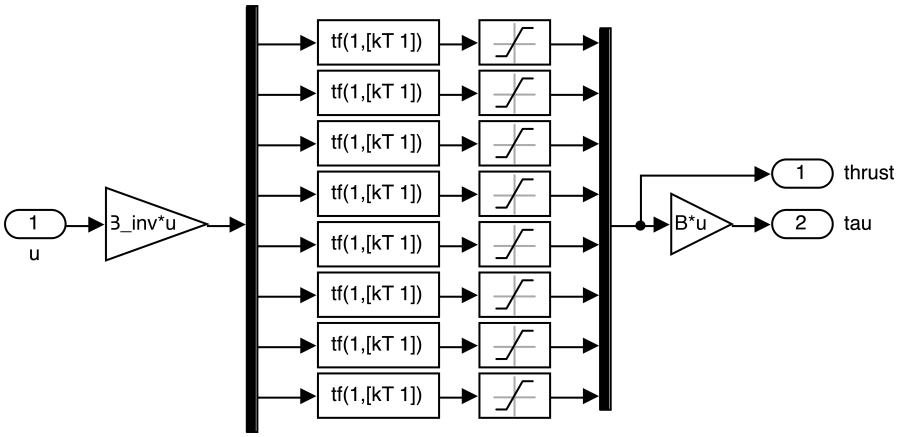



**Figure 3.3:** Simulink thrusters block

From a software implementation point of view, the ESC receives, downstream of the motor, a PWM signal to control the thruster. The model (or transfer function) that describes the ESC with its characteristics is not disclosed by the manufacturer. However, the performance charts are made public on the BlueRobotics website, including the force deliverable by the engine as the PWM changes to the ESC.

From the Basic ESC datasheet [17], the T200 thrusters to operate need a PWM signal from 1.100 µs to 1.900 µs

- 1.500 µs is the stop signal
- 1.500 µs to 1.900 µs for forward thrust
- 1.100 µs to 1.500 µs for reverse thrust

The chart reported in Fig. 3.4 shows that forward and backward thrusts are non-symmetrical. Furthermore, there is a range of PWM values $(1450 - 1550)$ providing zero thrust.

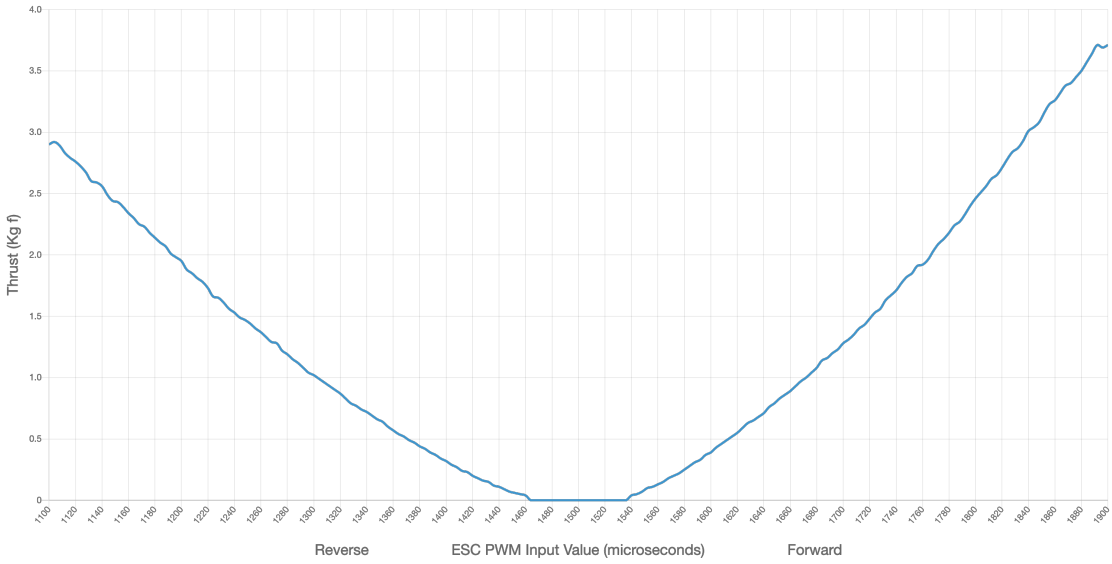



**Figure 3.4:** BlueRobotics T200 thruster (12 V) interpolated function mapping each ESC PWM input value to the produced thrust.

The inverse function expressed by (3.20) allows to predict the PWM value to be commanded by the ESC to the motor in order to obtain the desired thrust.

$$f(u) = \begin{cases} p_1(4)u^3 - p_1(3)u^2 + p_1(2)u + p_1(1), & u < 0 \\ 1500, & u = 0 \\ p_2(4)u^3 - p_2(3)u^2 + p_2(2)u + p_2(1), & u > 0 \end{cases} \tag{3.20}$$

where coefficients

$$p_1 = \begin{bmatrix} 7.3479 & 47.5779 & 199.2756 & 1.4608\text{e}3 \end{bmatrix} \tag{3.21}$$

$$p_2 = \begin{bmatrix} 3.8652 & -31.7639 & 160.9398 & 1.5391\text{e}3 \end{bmatrix} \tag{3.22}$$

are determined using `polyfit` MATLAB function, interpolating the datasheet values as shown in Appendix B.5. Figure 3.5 shows the chart of (3.20). The Simulink block (Fig. 3.6) acts as a *dispatcher* to obtain the thrust to each motor, starting from the cartesian forces and moments, and, finally, to obtain the PWM as input to each ESC. The constant gain of 0.102 is necessary to pass from Newton to KgF, accordingly to the datasheet of the thrusters.

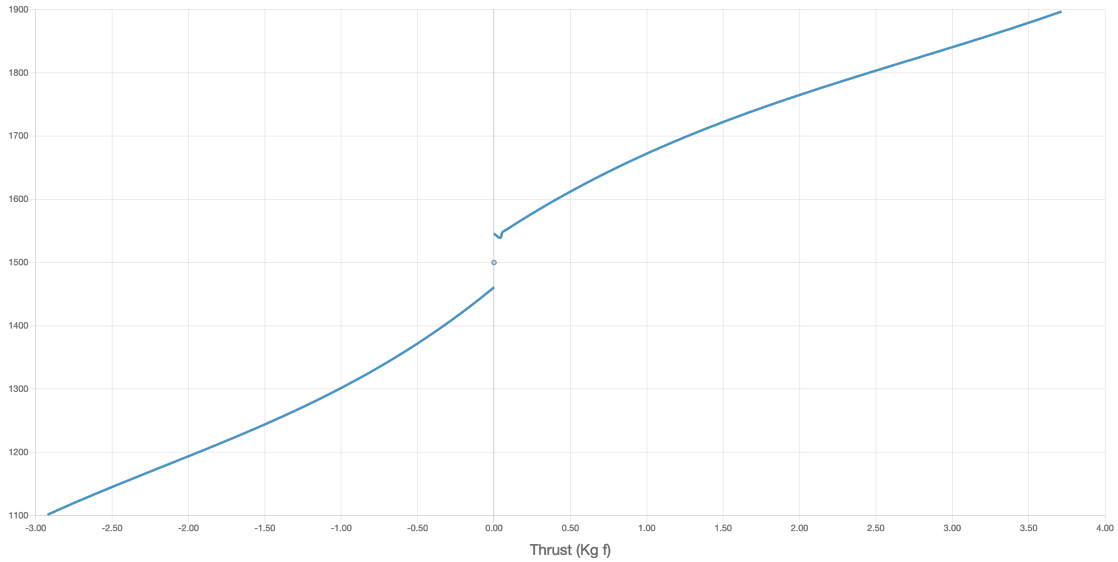



**Figure 3.5:** Interpolated function mapping BlueRobotics T200 (12 V) thrust to the corresponding ESC PWM input value.

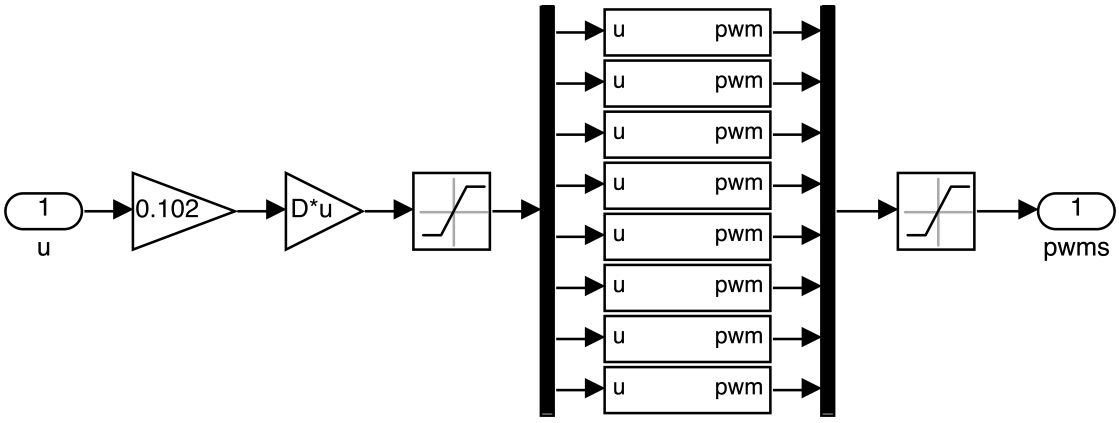



**Figure 3.6:** Simulink schema for code implementation

## 3.2 Sensors

The purpose of a navigation system is to compute in real time the best estimates of its position and velocity and these estimated values are then used by the ROV control system. To compute those estimates, the system receives information from high performance sensors.

### 3.2.1 Depth/Pressure

Maintaining the depth position is one of the fundamental issue for a good performance of an ROV. For this purpose, a sensor that can measure depth is needed. In this work, a depth sensor using a small size pressure sensor, which monitor depth changes on the basis of the pressure changes, has been preferred to a proximity sensor, which work is based on the time measured to the reflecting light. Indeed, proximity sensor has the disadvantages that the light propagation in water is different than in air.

In particular, the BlueRobotics Bar30 High-Resolution 300m Depth/Pressure Sensor has been mounted on board the ROV. It includes a depth/pressure and a temperature sensor, both communicating over $I^2C$. The pressure sensor can measure up to 30 bar (300 m depth) with a depth resolution of 2 mm. The temperature sensor is accurate to $\pm 1\,^{\circ}C$

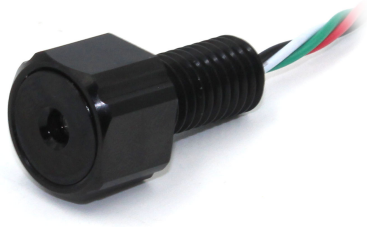


**Figure 3.7:** Bar30 High-Resolution 300m Depth/Pressure Sensor by BlueRobotics

### 3.2.2 IMU

The IMU supplies measurements of linear accelerations and angular velocities at high data rates.

In particular, the 9DOF Click by MIKROE mounts the ST's `LSM9DS1` inertial measurement module that combines a 3D accelerometer, a 3D gyroscope and a 3D magnetometer to measure 3-axis acceleration, angular velocity and heading (9-DoF data) in 16-bit resolution.

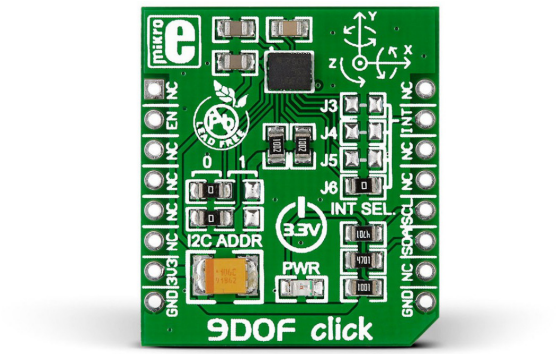



**Figure 3.8:** 9DOF Click by MIKROE

### 3.2.3 Others

The ROV mounts on board other sensors to measure and check pressure and temperature inside the add-ons (manipulator, camera, ecc...) and voltage and current in the Power Conversion System.

## 3.3 Control techniques

This section illustrates two different linear control system techniques, widely used in underwater robotics, especially in ROV or AUV applications: *PID* and *LQR*.

### 3.3.1 Proportional Integral Derivative (PID)

PID is the simplest and one of the most used linear control strategies. Because it allows on-line tuning on the real system to control, it is widely used when the transfer function of the system is unknown or not-well defined.

The PID control law (3.23) comprises

- a proportional action $K_p$, to reduce the rise time, the steady-state error, but increases the overshoot

- an integral action $\frac{K_i}{s}$, to eliminate the steady-state error, but increases the overshoot
- a derivative action $K_d s$, to reduce the overshoot, but increases the phase margin,

A best practice is to use a different PID controller for every degree of fredom of the ROV, so that the ROV is controllable in every direction to enable the autopilot [1]. However, even if simple to implement, this technique requires extensive manual tuning procedures.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{3.23}$$

### 3.3.2 Linear Quadratic Regulator (LQR)

LQR is a linear optimum control strategy. It aims to find the state-feedback law

$$u(t) = -Kx(t)$$

that solves the following optimization problem

$$\begin{aligned} \min_{u(t)} \quad & \int_0^\infty \left( x^T(t)Qx(t) + u^T(t)Ru(t) \right) dt \\ \text{s.t.} \quad & \dot{x}(t) = Ax(t) + Bu(t) \end{aligned} \tag{3.24}$$

The optimal solution of (3.24) is given by

$$u^*(t) = \arg\min_{u(t)} J(u) = -Kx(t)$$

where

$$K = R^{-1}B^T P \tag{3.25}$$

and $P$ is obtained solving the following *Algebraic Riccati Equation* ($ARE$)

$$A^T P + PA + Q - PBR^{-1}B^T P = 0 \tag{3.26}$$

Weight matrices $Q = Q^T >= 0$ and $R = R^T >= 0$ in the cost function are the design parameters chosen according to the desired performance trade-off. Tuning LQ regulators implies choosing the weight matrices $Q$ and $R$. They are usually chosen as diagonal matrices, with diagonal values $q_{jj} > 0$ and $r_{jj} > 0$ according to the relative importance of each state and control variable.

For control tracking problems, zero steady state tracking error can be achieved by introducing the integral of the tracking error as an additional system state ($x_i(t)$).

$$x_{aug}(t) = \begin{bmatrix} x_i(t) \\ x(t) \end{bmatrix} \Longrightarrow u(t) = -K x_{aug}(t) \tag{3.27}$$

## 3.4 Speed controller

The purpose of the speed controller is to regulate and keep the speed of the ROV constant to a reference velocity signal, assigned by the pilot through a joystick. During a mission execution, the ROV must be reactive, i.e., fast in reaching a new reference signal, and robust, i.e. able to reject disturbances.

In many ROV applications it is reasonable to assume that the ROV is moving in the longitudinal plane with non-zero velocity components $u_0$ and $w_0$ in the $x$- and $z$-directions, respectively and can rotate about the $z$-axis with non-zero angular velocity component $r_0$ [12].

The speed controller assigns the *free* velocities ($u$, $w$ and $r$), and regulates the remaining components.

The tuning parameters have been chosen in order to

- obtain appreciable settling time (5%) and rise time in the orders of tenths of a second
- reach the stady-state in about a second
- avoid the saturation of the actuators.

The latter point is crucial, because the saturation of the actuators introduces an unmodeled non-linear effect, causing unwanted behavior from the linear controllers. The datasheet [16] and the transfer function reported in Fig. 3.4, show that thrusters may deliver a reverse thrust di 2,9 kg F (28,43 N) and a forward thrust di 3,71 kg F (36,38 N).

### 3.4.1 PID

As described in Sec. 1.2.1 and 3.3.1, PID is the most used control technique due to its easy implementation and auto-tuning techniques, which allow to calibrate the controller parameters even in the absence of a well-defined mathematical model. In underwater applications, the widely used technique is to design one PID for each of the six degrees of freedom. As three are the controllable components of the linear and angular velocities ($u$, $w$ e $r$), three PI will act as controllers, the other three as regulators. The resulting control system is shown in Fig. 3.9. From simulation results, the derivative component of the PID does not introduce any advantage in terms of control, so $K_d = 0$.

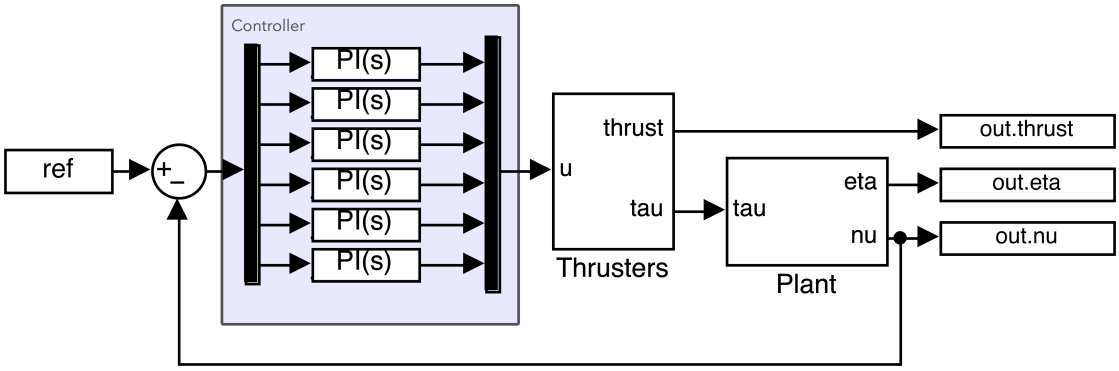



**Figure 3.9:** Control system design using 6 PIs one for each degree of freedom

The PID Tuner App by Simulink PID Block was used to tune the PID-based controller. By default, PID Tuner linearizes the plant and designs a controller at the operating point specified by the initial conditions in your Simulink model [18]. It is possible to tune the contrller by choosing a faster or slower response time, in combination with an more aggressive or more robust transient behavior. From the response time and transient behavior, MATLAB computes the parameters $K_p$, $K_i$, $K_d$ e $n$ which characterize a PID controller, so that, once the input has been given, the system has a behavior similar to that reported in Fig. 3.10. By the way, it rarely happens and a manually fine-tuning to adjust the parameters is almost always required.

PID tuning has been performed by applying the principle of superposition of effects. PIDs have been tuned singularly, one at a time, turning off the others. The principle of superposition of effects worked very well, even if the model is highly non-linear.

Controllers for linear velocity have been tuned using a step reference signal of 1 m/s, and $\frac{\pi}{6}$ rad/s for angular velocity, representing the maximum speed reachable by EVA.

### 3.4.2 LQI

As described in Sec. 3.3.2, the LQI, as well as the LQR, is an optimal control technique, in which the control law is computed as the solution of an optimization problem. Like PIDs, it is a linear control technique, and it needs the system to be linearized to solve the Algebraic Riccati Equation (ARE) and find the gain matrix $\boldsymbol{K}$. The `linmod` MATLAB function has been used for the linearization. It computes a linear state-space model by linearizing each block in a model individually to obtain linear models from systems of ordinary differential equations described as Simulink models [19]. The Simulink model shown in Fig. 4.1a was given to `linmod`

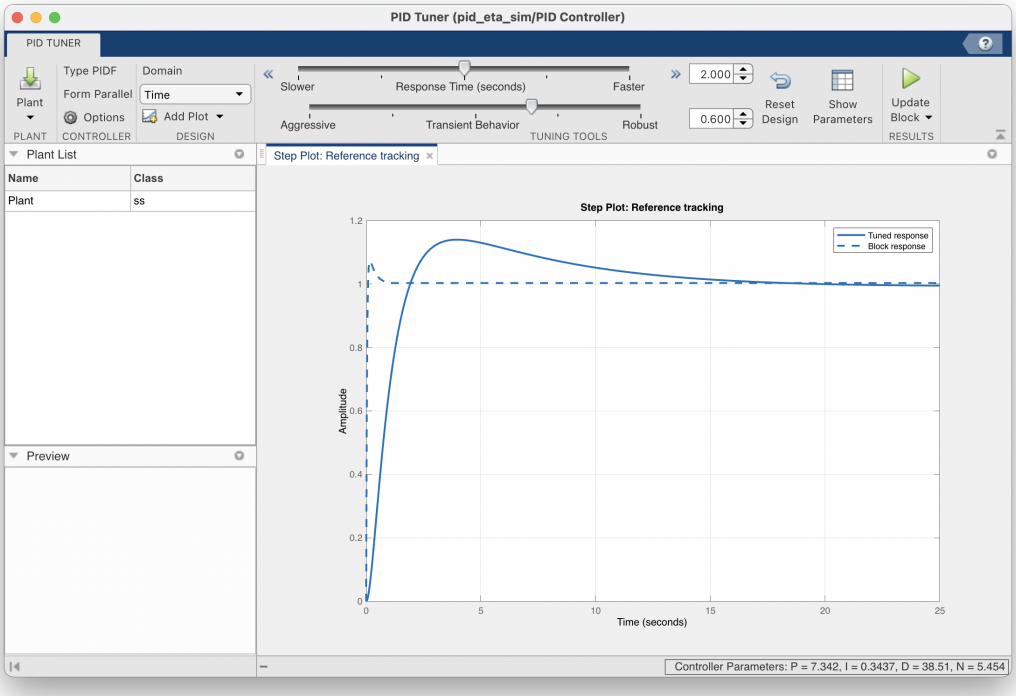



**Figure 3.10:** PID Tuner software

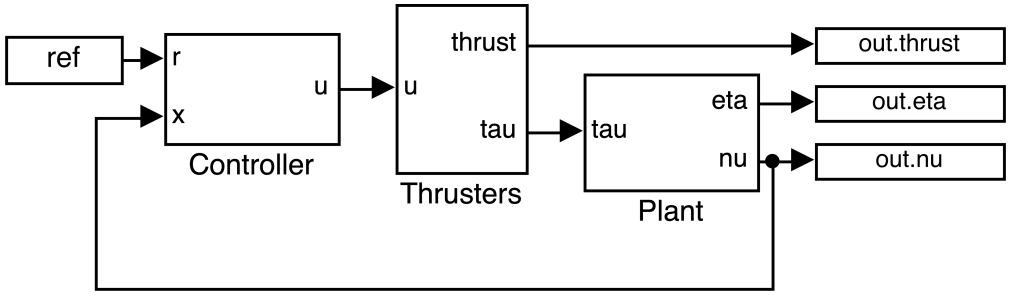
to perform the linearization.

The design of the LQI differs from the design of the LQR (described in 3.5.2) for the presence of an integrator whose purpose is to track a reference trajectory (or signal). The integral states are introduced in the system in an augmented state, so the matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are bigger then the LQR's ones, and there are more parameters to tune.

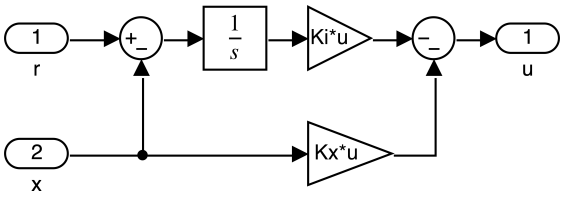
The resulting system is shown in Fig. 3.11a while Fig. 3.11b shows the LQI implementation in detail.

## 3.5 Position controller

Position controller is fundamental to define a reference trajectory for the motion of the ROV. In our case, it is enough to design a regulator for both position and attitude to allow the ROV to move on the horizontal plane keeping its depth constant and without changing its attitude, rotating or deviating from its original trajectory.




**(a)** Complete design




**(b)** Controller subsystem

**Figure 3.11:** Control system design using an LQI controller

### 3.5.1 PID

As for the speed controller, the PID-based position controller consists of six PIDs, one to regulate each degree of freedom. Tuning and linearization methods are the same described in Sec. 3.4.1, except for the reference signal. Indeed, being a regulator, the reference is the constant zero. To tune the characteristic parameters $K_p$, $K_i$, $K_d$ and $n$, the initial state for the position integrator has been set to values different from zero. In particular, the initial value of 0,5 m has been set for linear positions, and of $\frac{\pi}{6}$ rad for angular positions. They represents limit values the ROV must correct, because of inertia or external disturbances.

Each PID controller has been tuned individually, keeping the others off. The parameters have been chosen considering the best trade-off between the command activity and the convergence time. Finally, the principle of superposition has been applied. Unfortunately, it did not work as well as for the PI-based speed controller, so the parameters of each PID have been manually adjusted, considering the effect of the other PIDs, starting from the results previously obtained.

### 3.5.2 LQR

LQR is an optimal control strategy. Unlike the LQI, the integral with its added states is not present. The control law is obtained from the solution of the optimization problem described in (3.24), and the gain matrix $\boldsymbol{K}$ is obtained as in

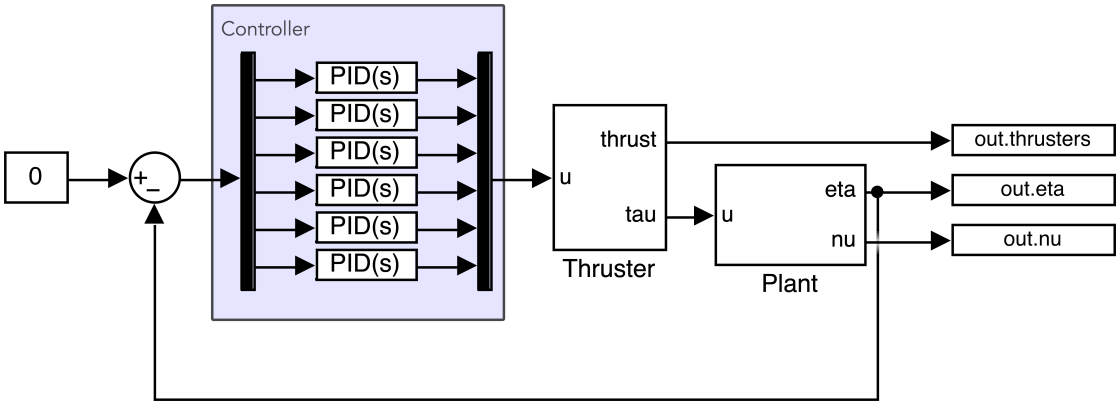



**Figure 3.12:** Control system design using 6 PIDs, one for each degree of freedom.

(3.25).

The same linearization method described in Sec. 3.4.2 has been performed. Weight matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, respectively for the states and inputs, are needed to solve the Algebraic Riccati Equation (ARE, 3.26), and were choosen diagonal and positive-definite. The parameters have been chosen considering trying to obtain the maximum deliverable thrust without saturating the actuators. This ensures the best achievable times without introducing non-linear effects. They were tuned considering as initial states 0,5 m for linear positions and $\frac{\pi}{6}$ rad for angular positions.

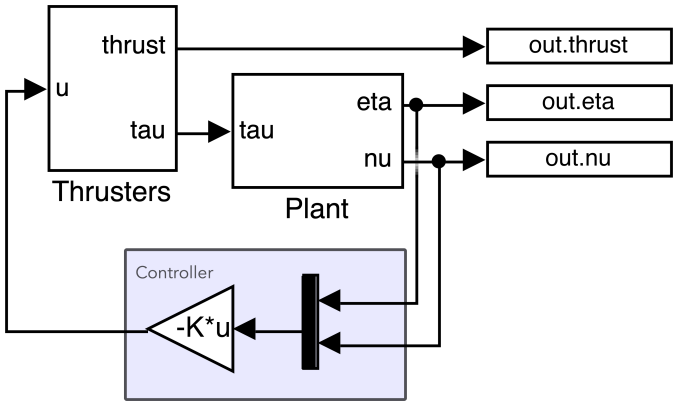



**Figure 3.13:** Control system design using an LQR

# Chapter 4

# Simulation results

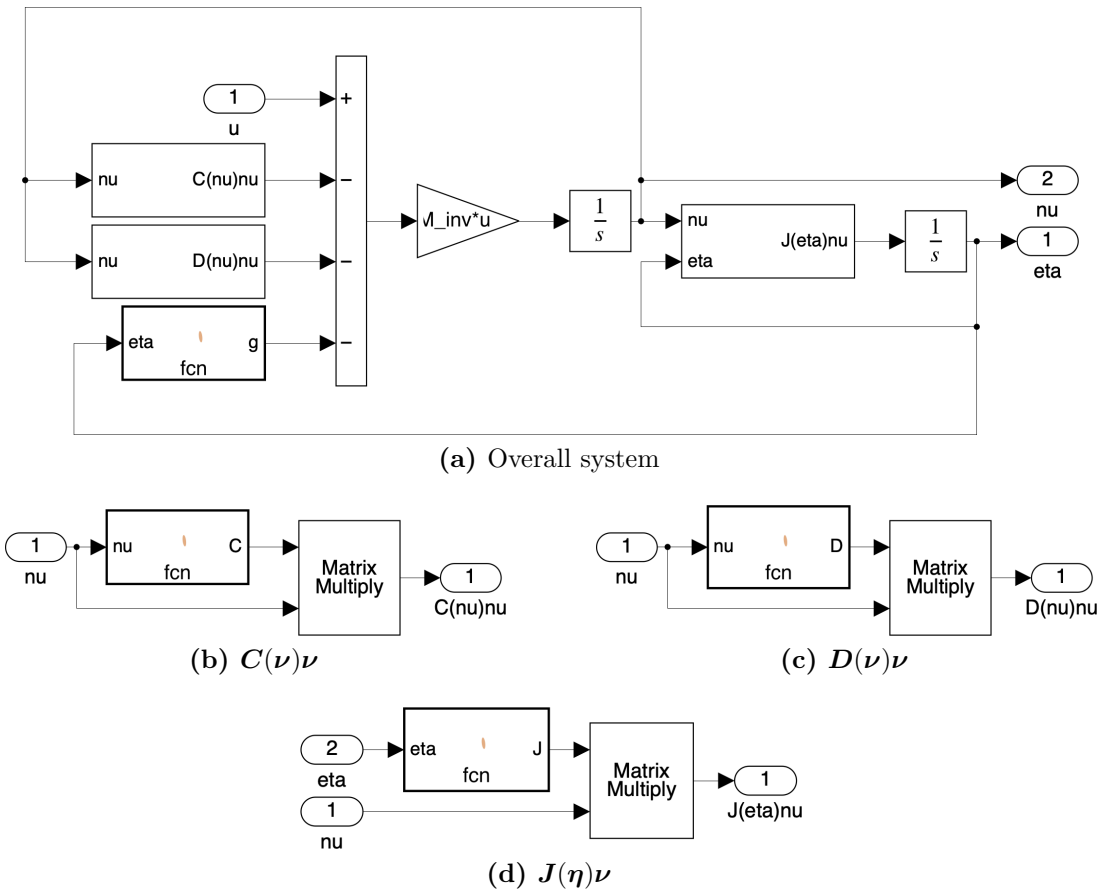
In this chapter, the results obtained from PID controllers and LQR/LQI controllers are analyzed and compared. Both the techniques are valid control strategies, the difference is minimal and mainly depends on the accuracy of the model and on the presence of unmodeled non-linearities, such as the thruster's saturation.

## 4.1 Modeling

The 6-DoF mathematical model of an ROV described in Chapter 2 and formulated by (2.26) and (2.27) is complex as it takes into account hydrodynamics forces, in particular radiation-induced forces, i.e, added inertia, hydrodynamic damping and restoring forces. These forces are difficult to estimate and depend on characteristic parameters that need to be identified. In this work, approximations have been made to overcome the problem. In particular,

- the added inertia was approximated to that of a prolate ellipsoid
- the drag force has been calculated approximating the shape of the ROV to a cuboid, with a drag coefficient of 0.5
- the buoyant force has been considered $1.01\boldsymbol{f_G}$, because the buoyant has been designed to give the ROV a slightly positive bouyancy

The complete model is represented in Fig. 4.1 by using Simulink blocks. This representation is necessary to simulate the physical behavior. In particular, Fig. 4.1a shows the overall system, the Simulink block translation of (2.26) and (2.27), and it is passed to `linmod` to perform the linearization. Fig. 4.1b, 4.1c, 4.1d show in detail the subsystems used in the overall plant, respectively for $C(\nu)\nu$, $D(\nu)\nu$ and $J(\eta)\nu$. In Appendix B for the used functions are reported the MATLAB listings.




**(a)** Overall system

**(b)** $\boldsymbol{C(\nu)\nu}$

**(c)** $\boldsymbol{D(\nu)\nu}$

**(d)** $\boldsymbol{J(\eta)\nu}$

**Figure 4.1:** Simulink schema of a 6DoF ROV described by (2.26) and (2.27)

Fig. 4.2 shows the slightly positive behavior of the ROV. Indeed, it tends to go up in water if no thrust is applied. Notice that the $z$-axis is taken positive downwards. From the same figure, the effect of the drag force can also be seen. Drag is a force that opposes motion due to an object's shape, material, and speed [20]. Drag force is proportional to the velocity for low-speed flow and the squared velocity for high speed flow, where low and high speed can be defined by the Reynolds number. Its effect is to reduce the vehicle speed and saturate it to a lower value. This effect can be seen from Fig. 4.2.

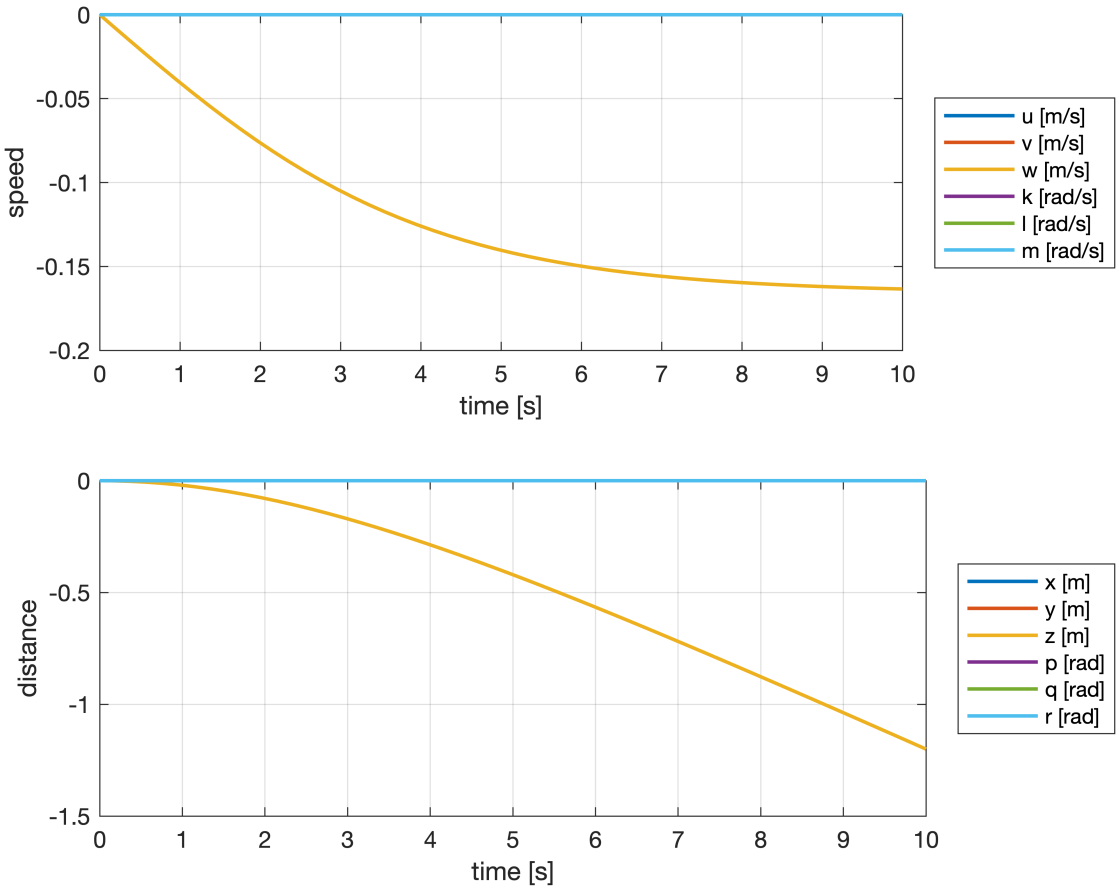



**Figure 4.2:** Speed reference tracking using an LQI control strategy.

## 4.2 Speed control

The speed controller was designed to reach the reference signal (maximum linear velocity of 1 m/s and maximum angular velocity of $\frac{\pi}{6}$ rad/s) within 1 s, avoiding

the thruster saturation. It is important to not introduce unmodeled non-linearities, which affect the performance of the developed control techniques.

The two control techniques proposed below to realize the speed controller will be compared on the basis of the following criteria:

- settling time, $t_{s,5\%}$
- rise time $(10\% - 90\%)$, $t_r$

The results obtained by using 6 PIDs, one for each degree of freedom, are reported in Fig. 4.3. In particular, in Fig. 4.3a a step reference signal of 1 m/s has been assigned for velocity $u$, along the $x$-axis. The reference is reached within a maximum command input of 32 N to the thrusters, after a rise time of 0,45 s and a settling time of 0,63 s. In Fig. 4.3b the same step reference signal of 1 m/s has been assigned for the velocity $w$, along the $z$-axis. The reference speed is reached within a maximum command input of 36,4 N to the thruster, with a rise time of 0,3 s and a settling time of 0,42 s. Fig. 4.3c shows the PID results relative to the YAW velocity $r$, tracking a step reference signal of $\frac{\pi}{6}$ rad. The maximum command input delivered to the thrusters is of 21,6 N for both the forward thrust and reverse thrust, obtaining a rise time of 0,23 s and a settling time of 0,34 s. Table 4.1 summarizes the described results.

| Fig. | $t_{s,5\%}$ (s) | $t_r$ (s) | max forward thrust (N) | max backward thrust (N) |
|---|---|---|---|---|
| 4.3a | 0.63 | 0.45 | 32 | 3 |
| 4.3b | 0.42 | 0.3 | 36.4 | 0 |
| 4.3c | 0.34 | 0.23 | 21.6 | 21.6 |

**Table 4.1:** Speed control results using PIDs

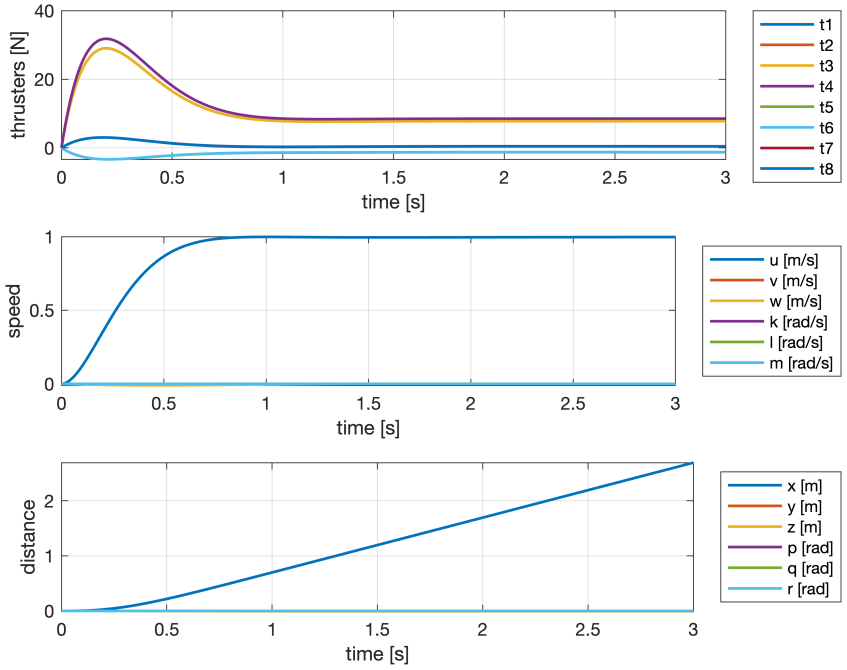
LQI has been designed and the results obtained are reported in Fig. 4.4. In particular, Fig. 4.4a shows a better performance than that obtained by the PID-based controller, with a rise time of 0,38 s and a settling time of 0,58 s. Fig. 4.4b shows that the step reference signal is reached with a rise time of 0,3 s and a settling time of 0,5 s, delivering to the thrusters a maximum command input of 36,5 N. The results achieved considering a step reference signal of $\frac{\pi}{6}$ rad for the YAW speed, $r$, are reported in Fig. 4.4c. It turns out that the system is able to reach the reference signal delivering to the thrusters a maximum command input of 22 N for both forward and reverse thrust, within a rise time of 0,15 s and a settling time of 0,25 s. Table 4.2 summarizes the results obtained from the LQI.

The analysis and the comparison of the obtained results for both the LQI and PID-based controller, lead to the conclusion that the LQI-based controller produces overall better results than the PID-based controller, in terms of rise time

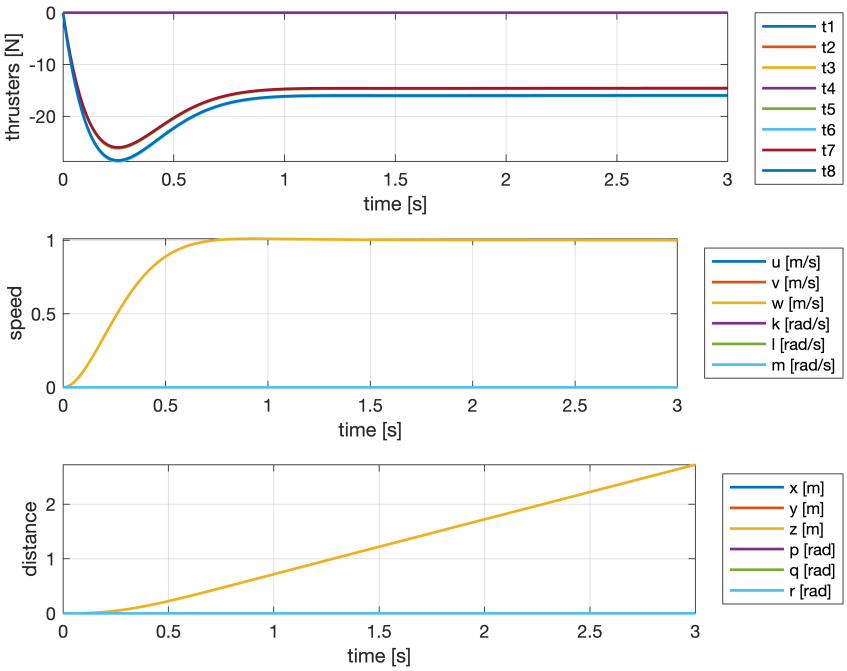
| Fig. | $t_{s,5\%}$ (s) | $t_r$ (s) | max forward thrust (N) | max backward thrust (N) |
|---|---|---|---|---|
| 4.3a | 0.58 | 0.38 | 32 | 3 |
| 4.3b | 0.5 | 0.3 | 36.4 | 0 |
| 4.3c | 0.25 | 0.15 | 21.6 | 21.6 |

**Table 4.2:** Speed control results using LQI

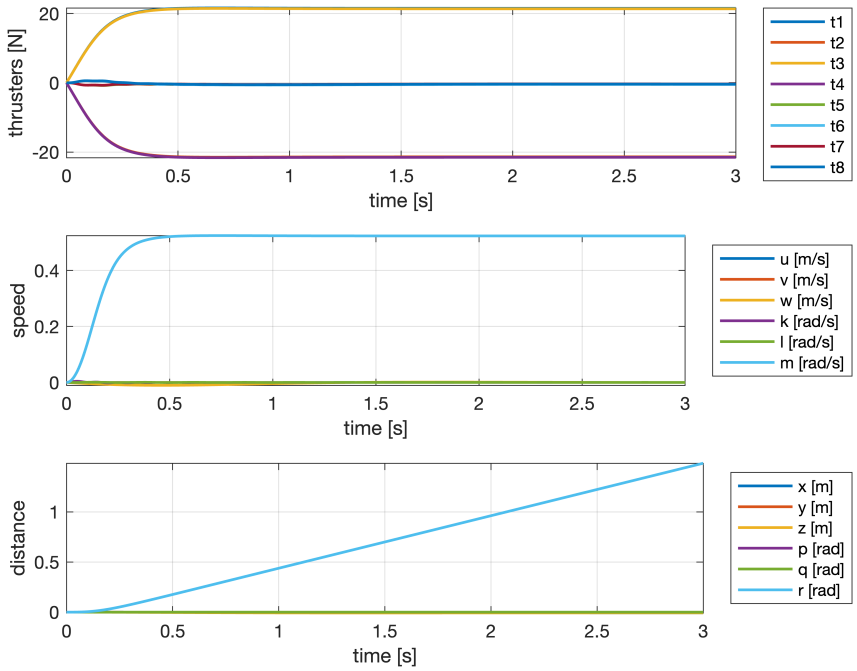
and settling time and considering the same delivered thrust. However, it can be noticed from Fig. 4.4c the presence of slight oscillations in both command input and angular velocities respectively from 0,2 N to 0,6 N and from −0,0045 rad/s to 0,004 rad/s after 1,5 s. In particular, the YAW velocity oscillates from 0,52375 rad/s to 0,52345 rad/s within a time period of 6 s, leading to a percentage steady-state error of 0,04%. Angular positions also oscillate −0,002 rad to 0,002 rad but within a larger time period (about 6 s). The oscillations are due to non-linear effects that were not taken into account by the linearization performed with `linmod`.




**(a)** $u = 1, \quad v = w = p = q = r = 0$

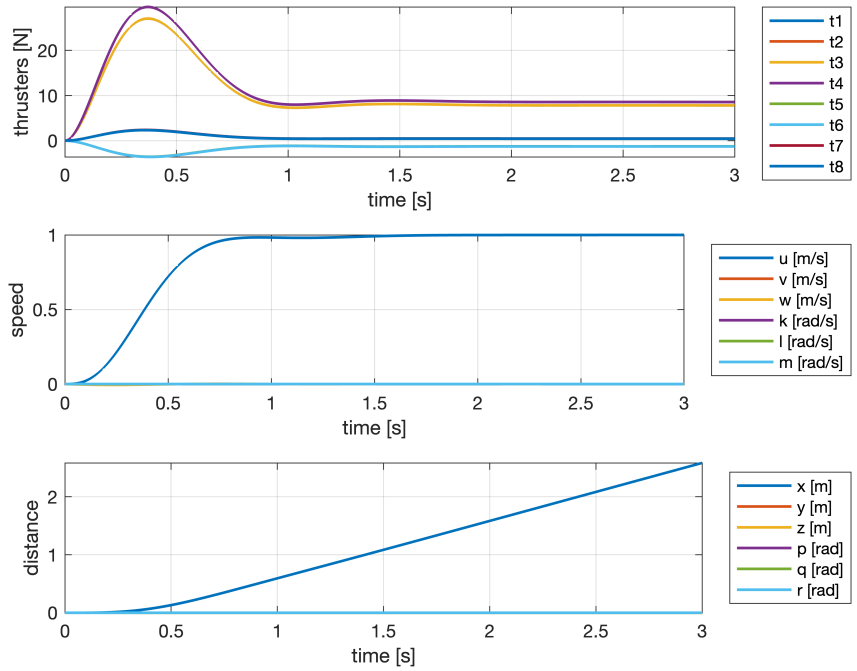



**(b)** $w = 1, \quad u = v = p = q = r = 0$

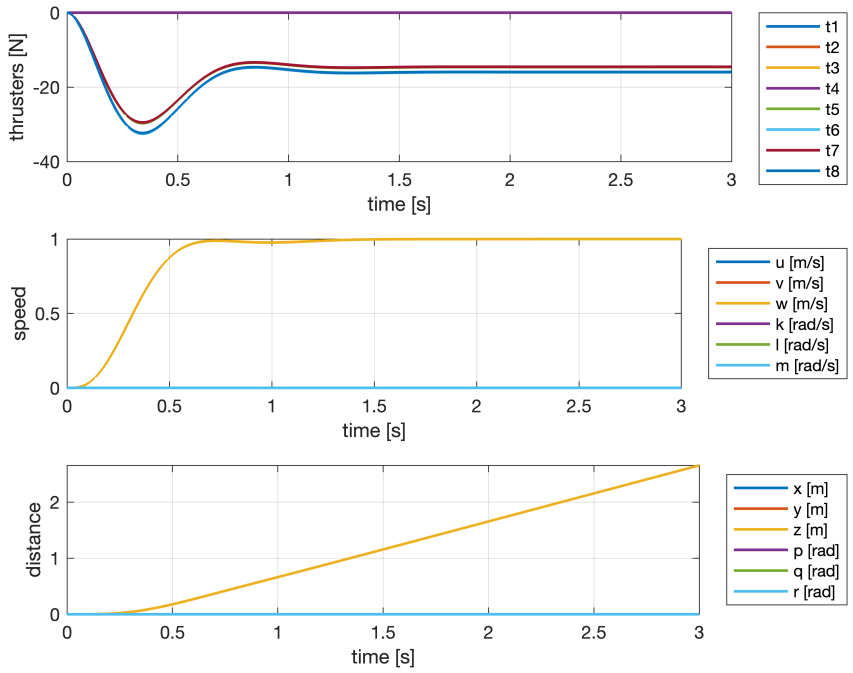



**(c)** $r = 1\,\mathrm{m/s}, \quad u = v = w = p = q = 0$

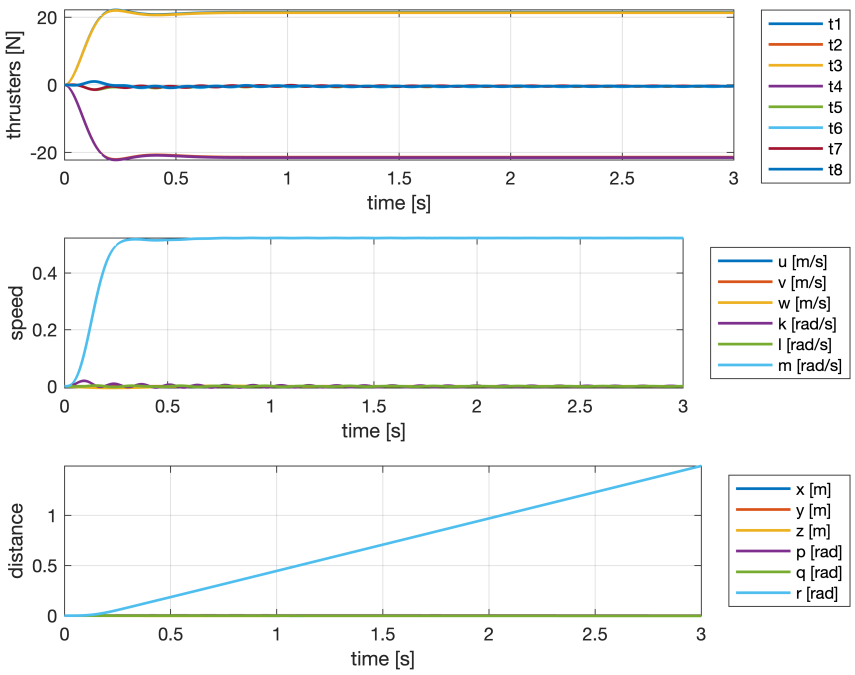
**Figure 4.3:** Speed reference tracking using a PID-based control strategy.




**(a)** $u = 1\,\mathrm{m/s}, \quad v = w = p = q = r = 0$




**(b)** $w = 1\,\mathrm{m/s}, \quad u = v = p = q = r = 0$




**(c)** $r = \dfrac{\pi}{6}\,\mathrm{rad/s}, \quad u = v = w = p = q = 0$

**Figure 4.4:** Speed reference tracking using an LQI control strategy.

## 4.3 Position control

The task of the position controller is of regulate the ROV position and attitude, making sure they do not change over time. In particular, it is fundamental to keep the ROV's depth and attitude unchanged while it moves on the horizontal plane. The next sections report the results which have been obtained by considering first an initial state of 0,5 m in depth and then $\frac{\pi}{6}$ rad in YAW, to simulate a change from the reference state and evaluate the controller's ability to correct it.

The comparison between the two proposed control techniques for the design of the position controller was made based on the following parameters:

- settling time, $t_{s,5\%}$
- rise time $(10\% - 90\%)$, $t_r$

To obtain appreciable results using PIDs, that is convergence time less than 3 s, it is necessary to induce the actuators' saturation, in particular $t_5$, $t_6$, $t_7$, $t_8$. Despite the saturation of the thrusters is an unmodeled non-linear effect, the PID-based control system is able to handle it avoiding command inputs, states and outputs oscillations.

Fig. 4.5a shows that, by setting an initial state of 0,5 m for the $z$-axis position (depth), a rise time equal to 1,1 s and a settling time equal to 1,55 s are obtained. For what concern the correction of $\frac{\pi}{6}$ rad on the YAW (Fig. 4.5b), a rise time equal to 1,66 s and a settling time equal to 2,11 s have been obtained without inducing the saturation of the actuator. The obtained results are summarized in Tab. 4.3.

| Fig. | $t_{s,5\%}$ (s) | $t_r$ (s) | max forward thrust (N) | max backward thrust (N) |
|---|---|---|---|---|
| 4.5a | 1.55 | 1.1 | 36.38 | 28.43 |
| 4.5b | 2.11 | 1.66 | 28.3 | 28.3 |

**Table 4.3:** Position control results using PIDs

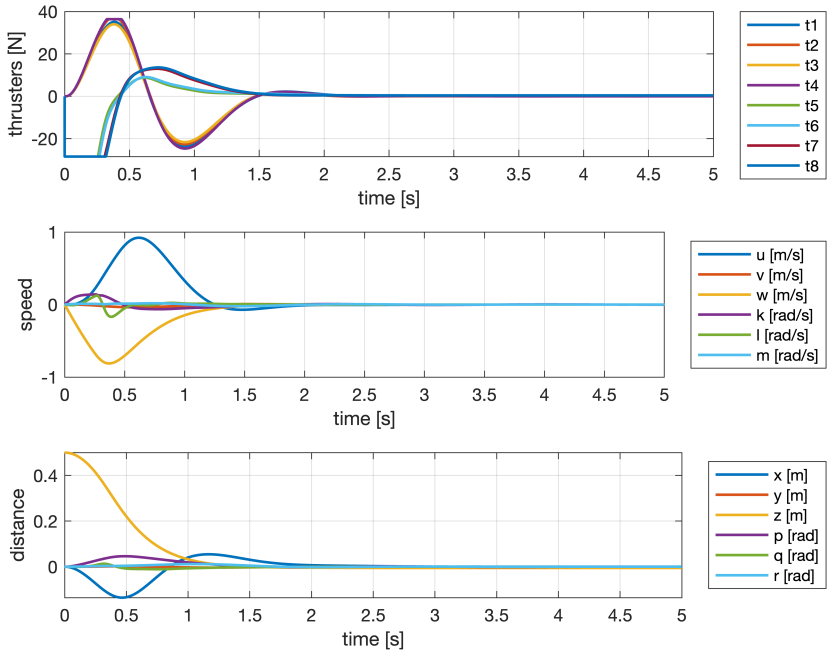
The LQR-based control systems achieve better performance than the corresponding controller based on PIDs, described above. Fig. 4.6a, shows that the designed controller is able to correct a change of 0,5 m of depth within a rise time of 0,85 s and a settling time of 1,24 s. From Fig. 4.6b, a rise time of 1,24 s and a settling time of 1,68 s are obtained by setting an initial state of $\frac{\pi}{6}$ rad for the YAW. The discussed results are reported in Tab. 4.4.

By comparing Tab. 4.3 and 4.4, it can be noticed that the times obtained from the LQR are better than the PIDs, even if the difference is small. Nevertheless, the LQR shows oscillations in both command input and angular velocities when

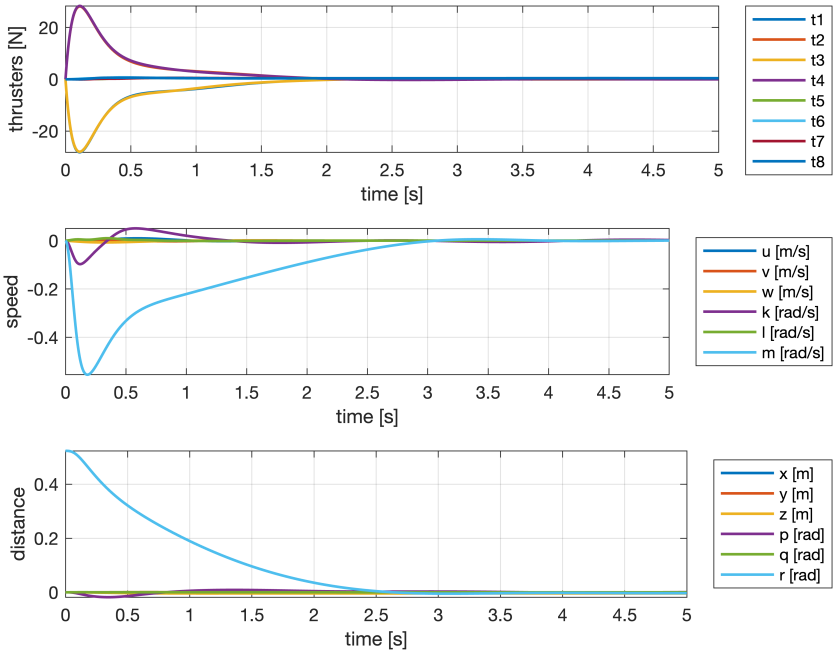
controlling the YAW, but unlike the LQI, oscillations decrease until they disappear at steady-state. It is noteworthy that oscillations tend to increase in the presence of saturating actuators. This is due to the fact that the design of LQR/LQI requires a linearization of non-linear models, and the actuators are not taken into account in the model. So the non-linearity due to the saturation of the actuators becomes an unmodeled component, affecting the performance of the control law.

| Fig. | $t_{s,5\%}$ (s) | $t_r$ (s) | max forward thrust (N) | max backward thrust (N) |
|---|---|---|---|---|
| 4.6a | 1.24 | 0.85 | 22.3 | 27.8 |
| 4.6b | 1.68 | 1.24 | 28.3 | 28.3 |

**Table 4.4:** Position control results using LQR

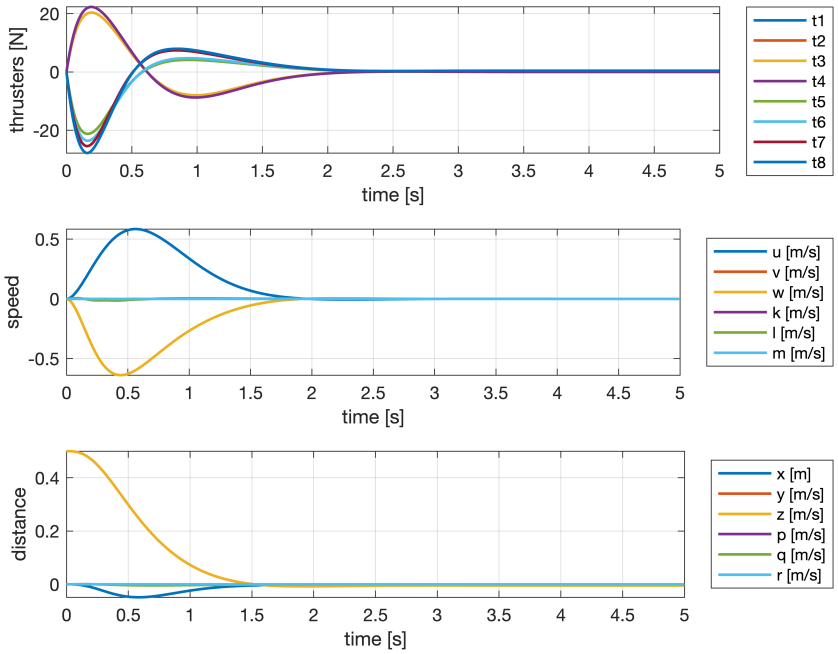



**(a)** $z = 0{,}5\,\mathrm{m}, \quad x = y = \phi = \theta = \psi = 0$

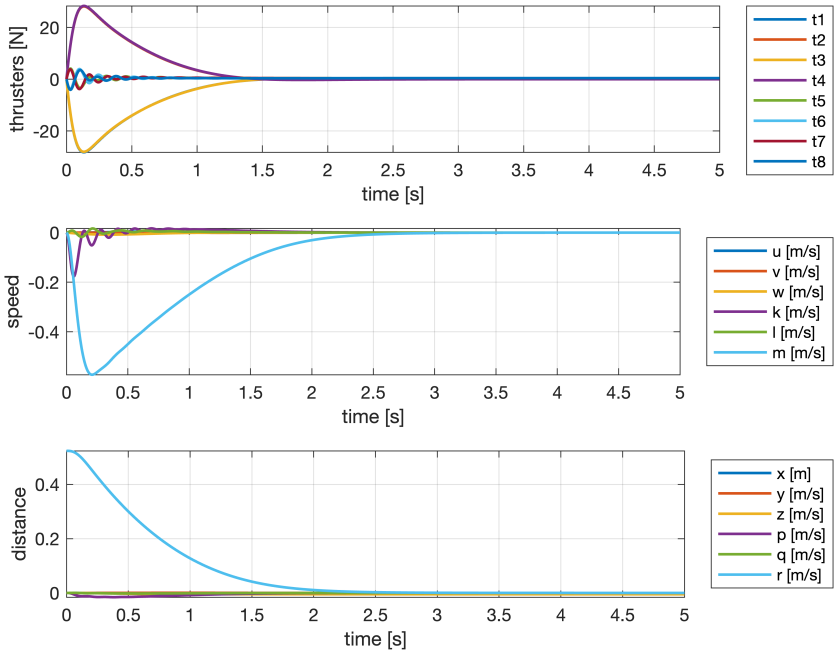



**(b)** $\psi = \dfrac{\pi}{6}\,\mathrm{rad}, \quad x = y = z = \phi = \theta = 0$

**Figure 4.5:** Compensation of a 0,5 m change in depth (4.5a) and of a $\dfrac{\pi}{6}$ rad change in yaw (4.5b) using a PID-based control strategy.




**(a)** $z = 0{,}5\,\mathrm{m}, \quad x = y = \phi = \theta = \psi = 0$




**(b)** $\psi = \dfrac{\pi}{6}\,\mathrm{rad}, \quad x = y = z = \phi = \theta = 0$

**Figure 4.6:** Compensation of a 0,5 m change in depth (4.6a) and of a $\dfrac{\pi}{6}$ rad change in yaw (4.6b) using an LQR control strategy.

# Chapter 5

# Conclusions

This thesis work addresses the design and simulation of two different control techniques, for both velocity and position of an ROV, starting from the definition of a 6-DoF mathematical model of the ROV.

The mathematical model of the ROV, described in depth in chapter 2 and formulated in (2.26) and (2.27) is highly non-linear, due to the hydrodynamic forces (drag and lift) and to the Corolis forces, which both depend from the ROV velocity. Moreover, the hydrodynamic forces, in particular the radiation-induced forces (added inertia and hydrodynamic damping), depend on characteristic parameters that are difficult to estimate and identify. A possible approach to estimate these parameters is CFD analysis, which allows to simulate fluids through numerical analysis. However, even the results obtained from CFD analysis must be validated by pratical and empirical trials. In this thesis work, the added mass has been approximated to that of a prolate ellipsoid, while the drag force has been approximated considering a cuboidal geometry and a drag coefficient of 0.5.

PID and LQR/LQI-based controllers are linear, therefore their design requires a linearization of the controlled plant. In chapter 4 a comparison between their performance has been carried out. From the simulation, it turned out the LQR/LQI seems to be more reactive in following the reference trajectory or in regulating the attitude, if the command input is not so high to saturate the thrusters. However, it shows some oscillations in controlling the YAW, vanishing at steady-state. When actuators are saturated, the PID-based controller performs better than LQR/LQI, mitigating the oscillations in command input, state and output. Indeed, the thrusters' saturation represents an unmodeled non-linear effect as it is not taken into account in the mathematical model, affecting especially the LQR/LQI performance, which is sensible to the model accuracy.

In conclusion, even if the LQR/LQI-based controller seams to be more reactive, the PID-based controller is slightly more robust to unmodeled behaviors.

## 5.1 Further works

An LQR/LQI-based controller cannot be implemented on the ROV EVA, due to sensors unavailability. Indeed, to implement an LQR/LQI-based controller, and to find the gain matrix $\boldsymbol{K}$, the system must be controllable, and all the states measurable. The *IMU* mounted on board of EVE measures the linear acceleration (*accelerometer*), the angular velocity (*gyroscope*) and the magnetic field (*magnetometer*). While the measure of the angular velocity is available, it is not possible to integrate the linear velocity, starting from the measures from the accelerometer, due to the sensor drift. A *DVL* (*Doppler Velocity Logger*) may be mounted to measure the velocity of the ROV, although, unfortunately, it is very expensive (about the same cost of the development of the whole ROV prototype).

For what concern the position controller, it is possible to measure the depth through the *Bar30* sensor, and the YAW, through the magnetometer from the *IMU*, allowing to control the ROV position along and around the $z$-axis. A sonar for each direction of motion should be mounted on board to measure the other positions.

Finally, it would be interesting to compare the results obtained in this work with non-linear control techniques, such as *NMPC* and *Sliding-mode controller*, the latter notably known for its robustness in the presence of parametric or unstructured uncertainties on the model.

# Appendix A

# Hydrodynamics

## A.1 Added mass

### A.1.1 Prolate spheroid

Consider an ellipsoid, totally submerged and with the origin at its center, described as:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1 \tag{A.1}$$

where $a$, $b$ and $c$ are the semi-axes. A prolate spheroid is obtained by setting the constraint $b = c$ and $a > b$. The following added mass coefficients can be derived:

$$\begin{aligned}
X_{\dot{u}} &= -\frac{\alpha_0}{2-\alpha_0} m \\
Y_{\dot{v}} &= Z_{\dot{w}} = -\frac{\beta_0}{2-\beta_0} m \\
K_{\dot{p}} &= 0 \\
N_{\dot{r}} &= M_{\dot{q}} = \frac{1}{5} \frac{(b^2-a^2)^2(\alpha_0-\beta_0)}{2(b^2-a^2)+(b^2+a^2)(\beta_0-\alpha_0)} m
\end{aligned}$$

where the mass of the prolate spheroid is:

$$m = \frac{4}{3}\pi\rho a b^2$$

and introducing the eccentricity $e$ defined as:

$$e = 1 - (\frac{b}{a})^2$$

the constants $\alpha_0$ and $\beta_0$ can be calculated as:

$$\alpha_0 = \frac{2(1-e^2}{e^3}\left(\frac{1}{2}\ln\frac{1+e}{1-e} - e\right)$$
$$\beta_0 = \frac{1}{e^2} - \frac{1-e^2}{2e^3}\ln\frac{1+e}{1-e}$$

### A.1.2 Cylinder

Consider a cylindrical body of mass $m$, length $L$ and with circular section of radius $r$. The following added mass coefficients can be derived:

$$X_{\dot{u}} = -0.1m$$
$$Y_{\dot{v}} = Y_{\dot{v}} = -\pi\rho r^2 L$$
$$K_{\dot{p}} = 0$$
$$N_{\dot{r}} = M_{\dot{q}} = -\frac{1}{12}\pi\rho r^2 L^3$$

# Appendix B

# MATLAB

**Listing B.1:** Function to evaluate $C(\nu)$ in Simulink block

```
function C = fcn(nu,Sys)

nu1 = nu(1:3);
nu2 = nu(4:6);

m = Sys.m;
I0 = Sys.I;
MA11 = Sys.M_A(1:3,1:3);
MA12 = Sys.M_A(1:3,4:6);
MA21 = Sys.M_A(4:6,1:3);
MA22 = Sys.M_A(4:6,4:6);

C_RB = [zeros(3)         -m*skew(nu1);
         -m*skew(nu1)    -skew(I0*nu2)];

C_A = [zeros(3)                    -skew(MA11*nu1+MA12*nu2);
         -skew(MA11*nu1+MA12*nu2) -skew(MA21*nu1+MA22*nu2)];

C = C_RB + C_A;
```

**Listing B.2:** Function to evaluate $D(\nu)$ in Simulink block

```
function D = fcn(nu, Drag)

A = Drag.A;
rho = Drag.rho;
Cd = Drag.Cd;

D = 0.5*rho*Cd*diag(diag(abs(nu).*A));
```

**Listing B.3:** Function to evaluate $g(\eta)$ in Simulink block

```
function g = fcn(eta, Sys)

ph = eta(4);
th = eta(5);

rho = Sys.rho;
V = Sys.V;

B = rho*9.81*V;
W = B*0.99;
g = [(W-B)*sin(th);
    -(W-B)*cos(th)*sin(ph);
    -(W-B)*cos(th)*cos(ph);
    0;
    0;
    0];
```

```
function J = fcn(eta)
ph = eta(4);
th = eta(5);
ps = eta(6);
J1 = [cos(ps)*cos(th) -sin(ps)*cos(ph)+cos(ps)*sin(th)*sin(ph) sin
   (ps)*sin(ph)+cos(ps)*cos(ph)*cos(th);
    sin(ps)*cos(th) cos(ps)*cos(ph)+sin(ph)*sin(th)*sin(ps) -cos(
   ps)*sin(ph)+sin(th)*sin(ps)*sin(ph);
    -sin(th) cos(th)*sin(ph) cos(th)*cos(ph)];

J2 = [1 sin(ph)*tan(th) cos(ph)*tan(th);
    0 cos(ph) -sin(ph);
    0 sin(ph)/cos(th) cos(ph)/cos(th)];

J = blkdiag(J1,J2);
```

**Listing B.4:** Using `polyfit` MATLAB command to find the coefficients of the polynomial interplating the data from datasheet [16]

```
load data/force.mat % Loads ForceKgF
load data/pwm.mat % Loads PWMs

p1 = polyfit(ForceKgF(1:92),PWMs(1:92), 3)
p2 = polyfit(ForceKgF(110:201),PWMs(110:201), 3)
```

**Listing B.5:** Inverse function to calculate ESC's PWM from thruster force using the polynomial coefficient found in B.4

```
function pwm = fcn(f)

if f < 0
    pwm = p1(4)*u^3-p1(3)*u^2+p1(2)*u+p1(1);
elseif f > 0
    pwm = p2(4)*u^3+p2(3)*u^2+p2(2)*u+p2(1);
else
    pwm = 1500;
end
```

# Bibliography


[1] Ying He, Dao Bo Wang, and Zain Anwar Ali. «A review of different designs and control models of remotely operated underwater vehicle». In: *Measurement and Control* 53.9-10 (2020), pp. 1561–1570. DOI: 10.1177/0020294020952483 (cit. on pp. 1, 5, 36).

[2] Kiam Beng Yeo, Teck Ho Wong, and Cheah Meng Ong. «Modelling and Manoeuvrability Design of Autonomous Underwater Vehicle». In: *Journal of Applied Sciences* 14.10 (2014), pp. 991–999. DOI: 10.3923/jas.2014.991.999 (cit. on p. 1).

[3] Zain Ali, Xinde Li, and Muhammad Tanveer. «Controlling and Stabilizing the Position of Remotely Operated Underwater Vehicle Equipped with a Gripper». In: *Wireless Personal Communications* 116 (2021). DOI: 10.1007/s11277-019-06938-2 (cit. on p. 3).

[4] Avilash Sahoo, Santosha K. Dwivedy, and P. S. Robi. «Design of a Compact ROV for River Exploration». In: AIR '17. New York, NY, USA: Association for Computing Machinery, 2017. ISBN: 9781450352949. DOI: 10.1145/3132446.3134894. URL: https://doi.org/10.1145/3132446.3134894 (cit. on p. 3).

[5] Christos Ioakeimidis, George Papatheodorou, Georgia Fermeli, N. Streftaris, and Enangelos Papathanassiou. «Use of ROV for assessing marine litter on the seafloor of Saronikos Gulf (Greece): a way to fill data gaps and deliver environmental education». In: *SpringerPlus* 4 (2015). DOI: 10.1186/s40064-015-1248-4 (cit. on p. 4).

[6] F.A. Azis, M.S.M. Aras, M.Z.A. Rashid, M.N. Othman, and S.S. Abdullah. «Problem Identification for Underwater Remotely Operated Vehicle (ROV): A Case Study». In: *Procedia Engineering* 41 (2012), pp. 554–560. ISSN: 1877-7058. DOI: 10.1016/j.proeng.2012.07.211 (cit. on p. 4).

[7] Francis Valentinis and Craig Woolsey. «Nonlinear control of a subscale submarine in emergency ascent». In: *Ocean Engineering* 171 (2019), pp. 646–662. DOI: 10.1016/j.oceaneng.2018.11.029 (cit. on p. 4).

[8] J.J.E. Slotine and W. Li. *Applied Nonlinear Control*. Prentice-Hall International Editions. Prentice-Hall, 1991. ISBN: 9780130400499 (cit. on p. 5).

[9] Bojarski M. et al. «End to End Learning for Self-Driving Cars». In: *CoRR* (2016) (cit. on p. 11).

[10] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. «You Only Look Once: Unified, Real-Time Object Detection». In: (2015). DOI: 10.48550/ARXIV.1506.02640. URL: https://arxiv.org/abs/1506.02640 (cit. on p. 12).

[11] Bellanova L. et al. «Detecting changes in a coral reef using CNN». In: *arXiv* (2020) (cit. on p. 14).

[12] T.I. Fossen. *Guidance and Control of Ocean Vehicles*. Wiley, 1994. ISBN: 9780471941132. URL: https://books.google.it/books?id=cwJUAAAAMAAJ (cit. on pp. 17, 28, 30, 37).

[13] Gianluca Antonelli. *Underwater Robots*. Springer Tracts in Advanced Robotics 123. Springer, 2018. ISBN: 9783319778983. DOI: 10.1007/978-3-319-77899-0 (cit. on p. 17).

[14] Jerzy Garus. «Optimization of Thrust Allocation in Propulsion System of Underwater Vehicle». In: *Int. J. Appl. Math. Comput. Sci* 14 (Jan. 2004), pp. 461–467 (cit. on pp. 27, 29).

[15] T.I. Fossen and M. Blanke. «Nonlinear output feedback control of underwater vehicle propellers using feedback form estimated axial flow velocity». In: *IEEE Journal of Oceanic Engineering* 25.2 (2000), pp. 241–255. DOI: 10.1109/48.838987 (cit. on p. 30).

[16] BlueRobotics. *T200 thruster*. URL: https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster-r2-rp/ (cit. on pp. 30, 37, 59).

[17] BlueRobotics. *Basic ESC*. URL: https://bluerobotics.com/store/thrusters/speed-controllers/besc30-r3/ (cit. on p. 32).

[18] Mathworks. *PID Tuner. Tune PID controllersexpand*. URL: https://it.mathworks.com/help/control/ref/pidtuner-app.html (cit. on p. 38).

[19] Mathworks. *linmod. Extract continuous-time linear state-space model around operating point*. URL: https://it.mathworks.com/help/simulink/slref/linmod.html (cit. on p. 38).

[20] Shane Maxemow. «That's a Drag: The Effects of Drag Forces». In: *Undergraduate Journal of Mathematical Modeling: One + Two* 2 (May 2013). DOI: 10.5038/2326-3652.2.1.4 (cit. on p. 44).