Universidade Federal da Bahia
Instituto de Matemática / Escola Politécnica

Programa de Pós-Graduação em Mecatrônica

OPTIMAL GUIDANCE AND CONTROL OF AUTONOMOUS UNDERWATER VEHICLE FOR DOCKING TASK

Yuri de Matos Alves de Oliveira

DISSERTAÇÃO DE MESTRADO

Salvador
April 14th of 2023

YURI DE MATOS ALVES DE OLIVEIRA

OPTIMAL GUIDANCE AND CONTROL OF AUTONOMOUS UNDERWATER VEHICLE FOR DOCKING TASK

Esta Dissertação de Mestrado foi apresentada ao Programa de Pós-Graduação em Mecatrônica da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Mecatrônica.

Orientador: Dr. Leizer Schnitman
Co-orientador: Jeremy Nicola

Salvador
April 14th of 2023

I dedicate this work to my family. Especially, my wife Yasmin, for her emotional support, encouragement and comprehension. My mom Pollyana, for all her effort put into my education. Aunt Natasha (Titati) and uncle João (Titino), for always being there, showing me the way and giving me support. My godson and little cousin Luca, for being a source of joy, love and happiness. My grandpa Charles and my grandma Ivone, for being amazing hosts during the period of this master program and for all the incentive. My love and gratitude to you all and to many other family members which I would not be able to write about in this single page. I'd like also to thank my friend Diogo for diving into various challenges with me during the master program.

ACKNOWLEDGEMENTS

I am extremely grateful for my supervisors, without them this work would not have been possible. Thank you Leizer for joining in another endeavor with me, for your academic advice and for handling very well this supervision during pandemic times and from the other side of the Atlantic. Thank you Jeremy for trusting me with the resolution of this problem, for all the technical advice and the opportunity you have given me. I'd like to thank Roberto Kawakami, Fabrice Le Bars and Marco Carraro from the jury who have put time into the evaluation of this work.

Last but not least, I'd like to deeply thank the Technology Innovation Institute (TII) for sponsoring and supporting this work, Universidade Federal da Bahia (UFBA) for the classes, support of professors and for making this work possible, and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) for supporting the university and mechatronics master program.

RESUMO

A inserção e retirada de veículos autônomos submarinos (AUV) são operações de alto custo, duração e risco para os operadores. Uma possível solução para a redução desses três fatores é a substituição destes operadores por veículos de superfície não tripulados (USV). Uma etapa fundamental para retirada do AUV de forma autônoma é a atracação, que é o momento em que o AUV entra na estação de atracação (DS) móvel, que por sua vez estaria sendo rebocada pelo USV. O algoritmo de controle do AUV deve garantir que a DS está no campo de visão (FOV) dos sensores do AUV, assim como que o AUV está no FOV dos sensores da DS. Ademais, o controlador deve controlar o AUV de forma precisa para evitar a colisão indesejada entre o AUV e a DS.

Um controlador muito utilizado na literatura para resolver esses problemas é o controlador preditivo baseado em modelo (MPC). Contudo, a maioria dos MPCs de atracação não consideram a predição do movimento da DS em seu problema de otimização. Além disso, diversos MPCs de atracação consideram que o objetivo do MPC é levar o AUV para a mesma pose da DS. Porém, ao fim da manobra de atracação, o AUV deve estar em uma pose e com uma velocidade relativa a DS, e não nas mesmas. Dessa forma, para a atracação de um AUV, essa pose e velocidade relativas devem ser consideradas no cálculo de referência do controlador. Portanto, a proposta deste trabalho é a de levar em consideração essa pose relativa de atracação e a predição de sua movimentação no MPC. Por fim, esse trabalho também se propõe a manter contribuições já realizadas por outros trabalhos. Tais contribuições são as implementações no MPC dos limites dos propulsores, garantia que o AUV está no FOV da DS e garantia que a DS está no FOV do AUV.

Para a validação da proposta deste trabalho, a solução foi testada em um ambiente de simulação no Gazebo, no qual a dinâmica submarina foi computada pelo *UUV Simulator* [1]. A *framework robot operating system* (ROS) [2] e a biblioteca de otimização Casadi [3] foram usados para o desenvolvimento do controlador e para a sua conexão com o simulador. Para os casos de teste, os parâmetros e o modelo 3D do veículo submarino comercial *BlueROV2 Heavy* foram utilizados para representar a DS e o AUV na simulação.

Dois casos de testes foram elaborados para demonstrar a eficiência da predição do movimento da pose de atracação no MPC, um com uma oscilação vertical da DS e outro com um movimento rotacional e horizontal. As restrições de FOV e a predição da pose de atracação foram testadas simultaneamente com sucesso para caso de teste com oscilação vertical, mas por ter sido computacionalmente muito custoso, a restrição do FOV do AUV foi retirada para o outro teste. Para ambos os casos de teste concluiu-se que o tempo de atracação é reduzido e que o erro entre a pose do AUV e sua pose de atracação é menor quando é realizada a predição do movimento da DS.

Palavras-chave: AUV, Atracação, Controle, Otimização, MPC.

ABSTRACT

The deployment and retrieval of Autonomous Underwater Vehicles (AUV) are operations with high costs, duration and risks for the operators. A possible solution to reduce these three factors is the autonomous deployment and retrieval of the AUV from Unmanned Surface Vehicles (USV) instead of operators in a surface vessel. A fundamental step in the full autonomous retrieval of the AUV is the docking stage, in which the AUV enters inside a mobile docking station (DS) attached to the USV. The control algorithm for docking has to ensure that the DS is in the AUV's sensors field of view (FOV) and that the AUV is in the DS's sensors FOV. Moreover, the controller has to control the vehicle precisely to avoid crashing the AUV with the DS or USV.

A controller that is vastly used in the literature to solve these problem is the model predictive controller (MPC). Nonetheless, most of docking MPCs do not take into account the prediction of DS movement in the optimization problem. Additionally, various docking MPC proposed in the literature considers, that the goal pose of docking MPC is the pose of the DS. Although, in reality, by the end of docking maneuver, the AUV has to be in a pose and with a velocity relative to the DS, but not in the same. Therefore, for an AUV docking, the computation of pose and velocity setpoint for the docking controller has to consider this relative pose. The proposal of this work is to take into account this docked pose and its movement prediction in the MPC. Additionally, this work will maintain some contributions that already have been developed for AUV MPC docking, which are the implementation of thrusters limits and FOV constraints of DS and AUV in the docking MPC controller.

For validation of the proposal, the solution was tested in a Gazebo simulated environment, in which the underwater dynamics were computed by underwater unmanned vehicles (UUV) simulator [1]. The robot operating system (ROS) [2] framework and Casadi [3] optimization library were used to develop the controller and connect it with the simulation. For the test cases, the parameters and 3D model of the commercial BlueROV2 Heavy were used to represent the DS and the AUV in the simulation.

Two test cases were designed to demonstrate the efficiency of predicting docked pose movement with the MPC, one with a vertical oscillation motion of the DS and the other with a rotational and horizontal motion. The FOV constraints and the prediction of the docked pose movement were successfully tested together for the vertical oscillation, but it was computationally expensive, thus only the DS FOV constraint was used for the other tests. For both test cases the conclusion was that the docking maneuver is faster and that the error between the AUV and the goal pose is smaller when the goal of the docking MPC uses the prediction of DS movement, instead of its current states.

Keywords: AUV, Docking, Control, Optimization, MPC.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

SYMBOLS

u	control action
x	states of a system or position in x axis
y	position in y axis
z	position in z axis
sp	setpoint
e	error
ph	prediction horizon
ch	control horizon
v_{lb}	variable v lower bound
v_{ub}	variable v upper bound
η	concatenation of position and rotation in tridimentional space
p	position or point in three dimentional space
Θ	euler angles vector
ν	concatenation of tridimentional linear and angular velocity
v	tridimentional linear velocity or vertex
ω	tridimentional angular velocity
ϕ	roll
θ	pitch
ψ	yaw
R	rotation matrix
M_A	added mass matrix
M_{RB}	rigid body intertia matrix
C	coriolis matrix
D	damping matrix
D_L	linear damping matrix
D_Q	quadratic damping matrix
g	hydrostatics vector
B	buoyancy force
T	thruster configuration matrix
f_n	thruters force vector
τ	thruters force in the 6 degrees of freedom
W	weight force
J	euler angle rate matrix
n	norm vector

cf	cost function
H	half space
S	smoothness weight diagonal matrix
Q	error weight diagonal matrix
Q_t	terminal error weight diagonal matrix

ACRONYMS

AUV Autonomous Underwater Vehicle

COG Center of Gravity

COB Center of Buoyancy

DOF Degrees Of Freedom

DS Docking Station

DP Docked Pose

FOV Field Of View

LOS Line Of Sight

GPS Global Positioning System

MPC Model Predictive Controller

NED North East Down

NWU North West Up

ODE Ordinary Differential Equation

PID Proportional Integral Derivative

ROV Remote Operated Vehicle

ROS Robot Operating System

SIL Software In the Loop

TCM Thruster Control Matrix

USBL Ultra Short BaseLine

USV Unmanned Surface Vehicle

UUV Underwater Unmanned Vehicle

Chapter 1

INTRODUCTION

1.1 MOTIVATION

Unmanned underwater vehicles (UUV) are robots used for performing underwater tasks. These robots can be used for a variety of tasks, such as military underwater mine countermeasures, observing marine life and inspection of underwater infrastructures, which sometimes are submerged at several kilometers under the surface of the ocean [4]. Generally, these vehicles are divided in two categories: remotely operated vehicles (ROV) and autonomous underwater vehicles (AUV). Their main difference is that ROVs are actively controlled by human operators via a communication tether, whereas AUVs are able to execute tasks with very low human interaction, once the mission has started and the AUV dives, the communication has very limited bandwidth.

POODLE, the first ROV, was developed in 1953 [5]. Still, it took almost thirty years for ROVs to become widely present on offshore applications [5]. Their spreading happened in 1980s due to oil and gas offshore extraction expansion, which created the need to perform underwater tasks at depths that are too dangerous for human divers to operate [5]. Most ROV's tasks could be done by divers, but there are many disadvantages on using humans divers to perform some of them. Diving in deep waters is a dangerous activity: any equipment fault can be fatal and sea creatures could harm the diver [6]. Divers cannot work in environments with high water currents, require advanced training, insurance and certifications due to the danger of their missions, and are slower than ROVs in the execution of some tasks, such as ship hull inspection [7]. Therefore, switching from divers to ROVs can reduce the risks, duration and costs of offshore underwater operation.

Even if the usage of ROVs instead of divers can improve the execution of underwater tasks, the human factor is still present and can cause errors due to fatigue or trouble interpreting data from robot sensors [8]. These errors can be mitigated by replacing ROVs with AUVs, removing entirely the operator from the task pipeline beyond the definition of the mission. Moreover, the tasks realized by ROVs can be more expensive than the ones realized by AUVs [4] [9] due to the expenses of hiring trained operators, and buying, renting or setting up a ship to enable the control of the robot. Therefore,

the replacement of ROVs by AUVs can improve the quality and reduce the costs of some operations.

Even though the automation of underwater tasks with AUV has advantages, AUVs still have a common problem with ROVs, which is its deployment and retrieval. Moreover, AUVs usually do not have tether cables that connect them with power sources like ROVs. Therefore, it may require a recharging mechanism for long missions. Some recharging mechanisms that were already used in the literature were solar energy for the SAUV [10] and wave motion for the self powered AUV [11]. However, these options depend on the environment conditions to function properly, differently from the docking station (DS), which is another recharging solution. It is a device that act as a garage for the vehicle. An advantage of the DS usage in opposition to the others is because it can solve three problems at once. DSs normally have three purposes: download of data collected by the AUV during missions, AUV recharging and AUV retrieval [12].

In [12], a DS can be classified in three categories regarding where and how it is deployed. The categories are fixed, floating and mobile. Fixed and floating are anchored, but the first stays underwater and the second stays floating on the surface. Mobile DSs are attached to surface vessels and stay hanging underwater. Independently of which DS is used, their launch and retrieval procedures still rely on human manual tasks, which again elevates risks and costs of AUV operations [12, 13]. In [13–16], the autonomous launch and recovery problem was further investigated and solved by the usage of an unmanned surface vehicle (USV) with a mobile DS to deploy and retrieve autonomously the AUV from an USV, via acoustic or optical communication.

An USV is another type of a marine autonomous vehicle, but instead of being underwater, it stays on the surface. USVs have a shape similar to ships and they can be used for some tasks, such as seabed mapping and seabed mineral research [14]. Using a USV alongside an AUV for launch and recovery have other advantages besides solving the problems listed with AUV autonomy. One of the advantages of combining AUV and USVs is the ability to perform longer tasks, since the USV can charge the AUV multiple times. Another advantage is regarding AUV localization. Differently from an AUV, the USV can use a global positioning system (GPS). By using GPS with an ultrashort baseline (USBL) mounted on the USV, it is possible to measure the AUV position and communicate it to the AUV.

Overall, it is possible to conclude that whether it is a fixed or mobile DS, the usage of a DS extends the autonomy of an AUV and reduces costs and risks of operation. Moreover, the mobile DS used with an USV and its cooperation with an AUV can enhance the performance of a task. However, to have these advantages that the DS can provide in a fully autonomous task, the AUV needs to be capable of autonomously docking inside the DS. The autonomous docking operation has several challenges. To overcome these challenges, the main software components are the DS localization, and AUV guidance and control [12].

Summarizing, increasing the automation of underwater tasks is key for their cost and risk reduction. From divers to ROV, AUV, and AUV with mobile and fixed DS to extend its autonomy, multiple researches have been conducted for the achievement of a fully autonomous underwater task with minimal human intervention. One important step for

this to be achieved is the autonomous docking task. The control of the vehicle for the docking task is a key component for its success and it has been researched since 1992 [17]. Control for docking task of AUVs has evolved from fuzzy and PIDs to optimal control [12], but it still has room for improvement.

1.2 PROBLEM STATEMENT

The AUV docking task in a mobile DS is a type of rendezvous problem, in which the robot has to meet a moving DS at a specific place. The control of an AUV for the docking task is the software component that decides in real time how to activate its thrusters to make the robot reach the docked pose without colliding with the docking station. The docked pose (DP) is a pose attached to the DS local frame, it is where a reference pose in the AUV local frame shall be at to be considered as docked. For a good performance of the controller, it is important that it acquires the correct pose of the DS, which can change frequently for the mobile DS case, since the DS is subjected to water currents, sea oscillation and to being pulled by the USV. This localization problem can be solved with acoustic, visual and electromagnetic sensors. Each type of sensor has its advantages and disadvantages.

Visual localization uses underwater cameras to localize features or markers in the DS. Since these features and markers are in fixed poses in relation to the DS, it is possible to calculate the DS pose from their pose estimation. The markers can either be passive, such as Apriltags [18], or active lights to be visualized even in the dark. The issue with visual localization is that it needs to have the DS feature or marker in the camera field of view (FOV) and it can be affected by dark environments and turbid water. Other than visual localization, it is possible to use acoustic and electromagnetic localization. These types of localization entails the trilateration of acoustic or electromagnetic signals broadcasted by beacons in the DS and obtained by a receiver in the AUV. It can also be the other way around, with the beacons in the AUV and the receiver in the DS. For this kind of method to work, the AUV must be close enough to the DS for the receivers to pick up signals that the beacons are sending out. All of the aforementioned methods involve estimating the DS pose, but there is also the option of just transmitting the AUV and DS poses to one another. Both acoustic and optical modems are capable of achieving it. For some optical modems, it's also necessary for one to be in the other's field of view in order for them to be able to communicate.

All of these problems regarding localization have a direct impact on the control problem, consequently, the controller has to be able to overcome the possible delays or measurements errors. All of the aforementioned localization and communications require a certain geometrical limitation in order to function. Some specifications require for the AUV to be nearby the DS, while others require for one to be in the FOV of the other. There is no range constraint for acoustic or optical modems because the AUV and DS will already be around 5 meters apart for the docking problem that we are contemplating solving. Even so, they can still move outside of the other FOV for optical communication and pose estimation. In order to prevent moving the AUV in a way that disrupts localization, the controller handling the docking problem must be aware of these circumstances

and how to avoid them.

Moreover, the controller also has to deal with the movement of the AUV and mobile DS. Due to the increased number of potential sources of disturbance, docking in a mobile DS in a USV is significantly more challenging. The DS movement motion will be impacted by both surface disturbances like wind and waves, as well as underwater disturbances like water currents and oscillation. Optimal control theory can handle these problems by using the knowledge of the vehicle dynamics model to predict its future movement and to optimize the error between the vehicle current pose and the future docked pose. However, this approach leads to the problem of getting an accurate dynamics model of the vehicle. For docking into a mobile DS, the problem is even harder, because not only the vehicle is moving but also the DS. To have a better prediction of the relative position of the vehicle in respect to the DS docked pose, the dynamics of the docked pose needs also to be taken into account.

Other things that the controller has to consider are the docking problem constraints. For an AUV to be able to dock, its sensors have to be capable of measuring the position of the DS, so the controller can avoid collision of the AUV with the DS. Some sensors, such as cameras and sonars, can only measure or capture data from objects that are in its FOV, which has a certain area and shape. Not only sensors, but some communications devices, such as optical modems, also need to have some alignment to transmit and receive data. Moreover, most DSs also have sensors to measure the AUV relative pose and then send it to the AUV. Therefore, another constraint of the docking controller is to ensure that the movement of the AUV will not remove at anytime over the prediction horizon the DS from the AUV's sensors FOV. At the same time, the controller has to assure that the AUV will not move out of the FOV of DS's sensors. These two constraints to keep the AUV and DS in each other sensors FOV are going to be referred in this work as co-visibility constraints. Finally, the controller shall also know the force limits of AUV's thrusters, so it does not try to send an impossible control action to the actuators.

Even considering that all these problems are solved, that the model is estimated, and that the localization and control software are developed, it is still a problem to test the system. It is very expensive to get sensors for accurately track AUV and DS pose and have a test site capable of generating wave and current disturbances in a controlled manner to check if the vehicle is able to respond well to these disturbances. To test the code directly on the vehicle is also a risk, because if some error occurs, it is possible to have a crash between the vehicle and the garage, damaging both of them. Therefore, another problem to be solved is the realistic test of the control software in simulated environment.

1.3 OBJECTIVES

From all these problems regarding AUV docking, the main goal of this work is to develop a controller capable of docking an AUV in a mobile DS. In order to accomplish that, a new model predictive controller (MPC) for underwater docking is developed. This new controller is inspired by the MPC approach developed for a spacecraft rendezvous with a non cooperative target in [19], taking into account the pose and velocity error between the

vehicle and the docked pose for every prediction. Moreover, the proposed MPC merges some positive aspects of the current AUV docking MPCs in the literature [20–24]. The details of the proposed MPC and its importance for docking are further explained in the Chap. 3.

The problem regarding localization and model parameter estimation is not in the scope of this work. The model parameters, localization and velocity of the AUV and DS in the experiments are considered as known. There will be no simulation of localization, the AUV will simply receive a very precise pose and velocity of the DS. Therefore, localization filters, measurement problems and disturbance observers are not a scope of our work.

To overcome the problem of testing the vehicle in a controlled environment, a computer simulation is developed. The simulation is done using UUV simulator [1], which simulates underwater rigid body dynamics, and displays the AUV and DS movement. Some sub-goals are necessary to complete all these steps, which are listed below:

- Model the dynamics of the AUV, DS and the docked pose.
- Include thrusters control in the model dynamics of the AUV.
- Develop a regular docking MPC with thrusters constraints.
- Include co-visibility constraints in the MPC.
- Include docked pose movement prediction in the MPC.
- Test the solution with software in the loop (SIL) simulation for different DS movement cases.
- Simulate and compare MPC with and without docked pose movement prediction.

1.4 THESIS STRUCTURE

In this chapter we explained the importance of docking an AUV in a mobile DS, its associated challenges and which of these problems we aim to address in this thesis. Chapter 2 explains the mathematical background necessary for understanding the thesis. Chapter 3 describes the state of the art for AUV docking with MPC, and elaborates our scientific contributions. Chapter 4 details the implementation of the MPC. Chapter 5 presents the system in which the solution is going to be tested and how it is implemented in the simulation environment. Chapter 6 describes the tests that were executed to prove the efficiency of the proposed solution and the comparison with current approaches. Chapter 7 concludes the thesis and suggests further future research.

Chapter

2

THEORETICAL FOUNDATION

2.1 OPTIMIZATION AND MODEL PREDICTIVE CONTROL

Model predictive control consists in converting a control problem to an optimization problem, thus before explaining MPC, explaining optimization is necessary. Optimization problem is the mathematical problem of finding values for variables of an objective function that minimizes it, while respecting some equality and inequality constraints [25]. Eq. (2.1) shows a standard way to specify an optimization problem [25]. To minimize the objective function $f(x)$ is the optimization goal. The inequality constraints are $g_i(x)$ and the equality constraints are $h_i(x)$. Then, an algorithm using an optimization method can be used to solve the problem.

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{s.t.} \quad & h_i(x) = 0,\ i \in 1, ..., p \\ & g_i(x) \leq 0,\ j \in 1, ..., m \end{aligned} \tag{2.1}$$

The control problem consists in choosing the control action u that leads system states x to a certain setpoint sp. MPC uses optimization to solve the control problem and find the optimal control action based on a heuristic, which is defined by the optimization cost function. Generally, this cost function involves the weighted sum of squares of system states error $e = sp - x$, but can also include other term, such as the square sum of control action variation $\Delta u = u_k - u_{k-1}$, thus the MPC chose a smooth control action that moves the system states to the setpoint [26]. It is also possible to add constraints such as the upper and lower bounds of the control action (u_{ub}, u_{lb}) and system states (x_{ub}, x_{lb}). Furthermore, it doesn't just optimize the states error in the next control iteration, but all states error in the prediction horizon ph, by selecting the optimal control actions in

the control horizon ch [26]. This example can be visualized in the Eq. (2.2).

$$\begin{aligned} \text{minimize} \quad & \sum_{i=0}^{ph-1} ||e_i|| + \sum_{j=0}^{ch-1} ||\Delta u_j|| \\ \text{s.t.} \quad & u_{lb} \leq u_j \leq u_{ub},\ j \in \{0, ..., ch-1\} \\ & x_{lb} \leq x_i \leq x_{ub},\ i \in \{0, ..., ph-1\} \\ & \Delta u_k = 0,\ k \in \{ch, ch+1, ..., ph-1\} \end{aligned} \tag{2.2}$$

2.2 UNDERWATER VEHICLE DYNAMICS

One important step of model prediction control is to obtain a dynamic model, which is a set of differential equations, that represents the system to be controlled. For docking AUVs with MPC, it is necessary to have a dynamic model of how the vehicle move and how the actuators influence its movement. This section goal is to explain the 6DOF AUV dynamics and mathematical notation of it. The entire section is based on the theory presented in [27, 28].

2.2.1 Notation and conventions

The SNAME notation [29] was defined in 1950 and included most of the nomenclature for marine craft dynamics modeling, which has also been vastly adopted for underwater robotics. SNAME notation will be utilized by this thesis as well, which will be introduced in this section, alongside other common robotics notation. First, for describing motion of the vehicle, it is necessary to define names and symbols for the 6DOF pose and velocities. The 6DOF pose is represented by the vertical $6x1$ vector η, which is a concatenation of the $3x1$ position vector p and the $3x1$ euler angles Θ. The 6DOF velocity is represented by the vertical $6x1$ vector ν, which is a concatenation of the $3x1$ linear velocity vector v and the $3x1$ angular velocity vector ω.

2.2.1.1 Frames Conventions The vehicle motion has to be defined in relation to an inertial frame of reference. It will be used a North West Up (NWU) frame convention on the water surface instead of a North East Down (NED) frame, which is used more often for underwater robotics. That decision was taken to approximate more the description of underwater robotics problem with aerial, spacecraft and ground vehicles, since most of them use NWU as well. Not only the inertial frame will be NWU, but all robots and objects used in this research are considered NWU. This means that for any robot the x direction it is from its center to its front, the y from its center to its left and the z from its center to its top side.

2.2.1.2 Relative Pose and Motion Notation All the velocities, forces and accelerations are describing the motion of a frame over time, expressed in its perspective. Independently of what the physical quantity being described by a symbol, its subscript represents the frame that has the physical quantity. For example, the linear velocity

of frame a would be represented as v_a, the angular acceleration of frame b would be represented as $\dot{\omega}_b$.

Even though this notation is enough for most cases, position and rotation cannot be represented only by it. Since rotation and position describes one frame in relation to another, a superscript will be added to represent the base frame. Therefore, the position and rotation will be represented from the superscript to the subscript from the perspective of the superscript. For example, the position of frame a with respect to the frame b is represented by p_a^b. In case the base frame is the inertial frame of the environment, the superscript can be omitted.

2.2.1.3 Rotation Conventions Rotation can be represented by using quaternions, axis angle or euler angles. From these three, the easiest one for a human to visualize are euler angles. Moreover, euler angles are the convention in underwater vehicle dynamics. Therefore, euler angles will be our choice for representing orientation. From all the 24 euler angles possibilities, the XYZ extrinsic rotation will be utilized, in which the rotation are in first executed around x, then y and then z fixed axis. The name of the rotation around XYZ axis are respectively roll ϕ, pitch θ and yaw ψ. Therefore, the Eq. (2.3) represents rotation matrix from frame a to frame b, represented by $R(\Theta_b^a)$, generated by the euler angles Θ_b^a. In the Eq. (2.3), $cos(x)$ and $sin(x)$ were abbreviated to $c(x)$ and $s(x)$ respectively. Moreover, the operation defined by $\Theta(R_b^a)$ represents the conversion of the rotation matrix in roll, pitch and yaw angles Θ_b^a.

$$R(\Theta_b^a) = \begin{bmatrix} c(\psi)c(\theta) & s(\psi)c(\theta) & -s(\theta) \\ -s(\psi)c(\phi) + c(\psi)s(\theta)s(\phi) & c(\psi)c(\phi) + s(\psi)s(\theta)s(\phi) & s(\phi)c(\theta) \\ s(\psi)s(\phi) + c(\psi)s(\theta)c(\phi) & -c(\psi)s(\phi) + s(\psi)s(\theta)c(\phi) & c(\phi)c(\theta) \end{bmatrix} \tag{2.3}$$

With the position and rotation convention established, it is still necessary to define how to use them for pose transformations. Thus, the operator $\otimes$ will be used in our work to represent a position and rotation transformation, which is defined in Eq. (2.4).

$$\eta_b^a \otimes \eta_c^b = \eta_c^a = \begin{bmatrix} p_c^a \\ \Theta_c^a \end{bmatrix} = \begin{bmatrix} p_b^a + R(\Theta_c^b) \times p_c^b \\ \Theta(R(\Theta_b^a) \times R(\Theta_c^b)) \end{bmatrix} \tag{2.4}$$

2.2.2 12 DOF model

To understand the AUV motion in 6DOF simply from its current state and actuators commands, two steps are necessary. First, we must understand the dynamic model, which is how all the forces contribute for generating the 6DOF velocity of the vehicle. Afterwards, it is required to understand how the 6DOF velocity calculated in the dynamic model will impact on the change of pose of the vehicle, which is the kinematics model. Combining these two models is essential to fully comprehend vehicle motion, which is a fundamental part of the AUV docking MPC.

2.2.3 Dynamics

Newton's second law of motion stated that the sum of forces had to be equal to the mass times acceleration. Eq. 2.5 is also Newton's second law, but it is also detailing forces that act on a immersed rigid body. The mass of 6DOF submersed rigid body is the 6x6 regular mass of the rigid body M_{RB} plus added mass M_A. The added mass is a 6x6 matrix that represents the inertia of a moving fluid around the immersed rigid body. In ground and aerial robotics, this added mass is normally discarded because of the air density being much lower than water density. The added mass matrix is mainly dependent on water density and geometry of the immersed rigid body.

$$(M_A + M_{RB})\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\Theta) = \tau \tag{2.5}$$

The coriolis and centripetal forces are caused by earth rotation and are represented by 6x6 $C(\nu)$ matrix multiplied by velocity. These forces actually can be neglected since they are very small for small velocities, which is the case of the docking maneuver.

The damping forces are a dissipative drag force in the vehicle due to the viscosity of the fluid, which is represented as $D(\nu)$. It can be divided into linear and quadratic terms as it is in Eq. (2.6), in which the linear damping D_L is a 6x6 constant matrix and the quadratic damping D_Q is a 6x6 matrix that is proportional to the modulus of the velocity. The constants of the damping matrix are related to the geometry of the vehicle. For vehicles with XY, ZY and ZX planes symmetry, the damping matrix is a diagonal.

$$D(\nu) = D_L + D_Q(\nu) \tag{2.6}$$

Thus far, all the forces explained depend on vehicle movement. In addition, the weight and buoyancy forces $g(\Theta)$ always keep being applied in an immersed rigid body. They act similar to a pendulum, with the weight pulling the center of gravity (COG) down, and buoyancy pushing the center of buoyancy (COB), as it is illustrated in the Fig. 2.1. The COG is a point of the rigid body where the mass distribution is the same for all directions. The COB is similar, but instead of being the mass distribution of the rigid body is the mass distribution of the fluid that would occupy the space that is now occupied by that rigid body. The farther away these centers are in an AUV, the stronger is the force putting the vehicle back in its default orientation. Without any external forces, the default orientation of a submersed body is when the COB on top of the COG, vertically aligned with it. Therefore, for a more stable AUV, the farther away the centers are the better, but for an AUV to be more maneuverable, the closer they are, the better.

In a fully submersed rigid body in a constant density fluid, the COB can be considered a point where the distribution of volume is the same in all directions. How weight and buoyancy force act on the 6DOF of the vehicle is expressed in Eq. (2.7). In this equation, W represent the magnitude of the weight force, B the magnitude of buoyancy force, xyz_G

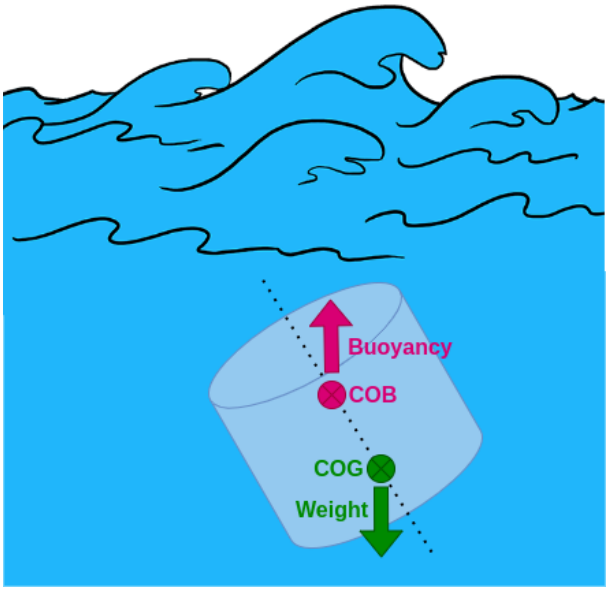

Figure 2.1 Illustration of buoancy and weight forces in a submersed body.

COG relative coordinates and xyz_B COB relative coordinates.

$$g(\Theta) = \begin{bmatrix} (W-B)sin(\theta) \\ -(W-B)cos(\theta)sin(\phi) \\ -(W-B)cos(\theta)cos(\phi) \\ -(y_GW - y_BB)cos(\theta)cos(\phi) + (z_GW - z_BB)cos(\theta)sin(\phi) \\ (z_GW - z_BB)sin(\theta) + (x_GW - x_BB)cos(\theta)cos(\phi) \\ -(x_GW - x_BB)cos(\theta)sin(\phi) - (y_GW - y_BB)sin(\theta) \end{bmatrix} \tag{2.7}$$

Finally, the last force τ in the Eq. (2.5) is the one produced by AUV thrusters. Each thruster of the AUV has an specific position and is pointed at one specific direction, this need to be mapped in the 6DOF, as it is in Eq. (2.8). This mapping is done by the Thruster Control Matrix (TCM) T, which is a $6xn$ matrix that convert the n thrusters forces in vertical vector f_n into a 6DOF vector force aligned with the robot frame.

$$\tau = Tf_n \tag{2.8}$$

2.2.4 Kinematics

AUV dynamic model Eq. (2.5) enables the acceleration estimation with the AUV orientation, velocity and control output. By integrating the acceleration is possible to estimate and predict the AUV velocity, which is going to be used for the docking MPC. Although, for docking an AUV it is also important to predict its movement, how its pose will change

in time. In order to do that, the velocity obtained using the dynamic model is converted from the frame of the robot to the inertial frame on the water surface by Eq. (2.5). In this equation, O_{3x3} represents a $3x3$ matrix containing only zeroes.

$$\dot{\eta} = \begin{bmatrix} R(\Theta) & O_{3x3} \\ O_{3x3} & J(\Theta) \end{bmatrix} \nu \tag{2.9}$$

Two operations are needed for converting the AUV velocity from one frame to another: the conversion of linear and angular velocities. For the linear velocity conversion, the AUV linear velocity has to be rotated by the AUV orientation in relation to the inertial frame, as it is in Eq. (2.10). For the angular velocity conversion, the euler angle rate matrix J in Eq. (2.11) will be used.

$$\dot{p} = R(\Theta)v \tag{2.10}$$

$$\dot{\Theta} = J(\Theta)\omega = \begin{bmatrix} cos(\theta) & sin(\phi)sin(\theta) & cos(\phi)sin(\theta) \\ 0 & cos(\phi)cos(\theta) & -sin(\phi)cos(\theta) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix} \frac{1}{cos(\theta)}\omega \tag{2.11}$$

2.3 HYPERPLANES AND HALFSPACES INTERSECTION

A last important topic for understanding the MPC controller designed in this thesis is halfspace. It is important because it can be used as an MPC constraint to restrict the movement of the vehicle, so that it does not go into a place that is not supposed to. Particularly, it can be used to guarantee the co-visibility constraints.

A halfspace is one of two parts of the affine space divided by a hyperplane [30]. A hyperplane is an affine subspace with one less dimension than the affine space being used [30]. Since this work is being done in three dimensional space, the hyperplane is simply a plane and the halfspace is part of the space divided by a plane, which can be represented by the Eq. (2.12). For example, if we consider the XY plane passing through the origin, we would have two halfspaces, which would be above and below the z axis.

$$ax + by + cz \leq d \tag{2.12}$$

One way of finding the parameters a, b, c and d of the Eq. (2.12) is by selecting a point that belongs to a certain plane and a vector normal to the plane. If a point with coordinates (p_x, p_y, p_z) is selected and a vector normal to the plane (n_x, n_y, n_z) is selected, the halfspace, which is represented in Fig. 2.3, equation parameters can be found by Eq. (2.13). In that equation, any point xyz that makes the equation valid is a point that belongs to the halfspace.

$$n_x x + n_y y + n_z z \leq n_x p_x + n_y p_y + n_z p_z \tag{2.13}$$

Chapter 3

RESEARCH IMPACT

The objective subsection listed six features of AUV docking MPC. The goal of this chapter is to show how these features are a scientific contribution to docking MPC research, how they can be implemented and how these features will contribute to the AUV docking task. First, in the literature review section, some research papers will be discussed to demonstrate what is the state of the art of AUV MPC docking and to highlight how researches have and have not implemented some of the objectives of this thesis. Afterwards, each proposed contribution of this thesis has a specific section presenting further details.

3.1 LITERATURE REVIEW OF AUTONOMOUS UNDERWATER VEHICLE DOCKING

More than 60 years have passed since the invention of the first AUV, the SPURV [31]. Nonetheless, it took until 1992 for the first research concerning docking an AUV autonomously to be published [17]. This first attempt of docking consisted in using a fuzzy controller to dock one AUV into a submarine with a mobile DS on simulation. Following that, the REMUS 100 was the first AUV designed with autonomous docking goal. Its docking strategy, docking station and localization features are detailed in [32]. The first successful autonomous docking was also made with the REMUS 100 [33], but with a success rate of approximately 58.6%, which is not a very reliable rate.

Afterwards, multiple methods regarding docking guidance and control have been developed [12]. These methods were mostly non optimal control solutions [12], attempting to solve the problem using different guidance and control strategies to compensate for water currents [34], but not considering the dynamical model of the vehicle. Even though is possible to tune PIDs and to use guidance methods to dock an AUV, the control action is not optimal and it may not respect FOV, LOS and thrusters limits constraints that a docking problem has. Comparing underwater docking control to aerial and space vehicles, marine research is not taking much advantage of the vehicle dynamical model to control the vehicle in the docking/rendezvous problem [12].

Nevertheless, that is not the case for every AUV docking research. In the last decade, some papers implemented MPC for AUV docking [20–24]. In [20], a stochastic MPC was used to dock a AUV with 5 degrees of freedom (DOF). A model with AUV pose and velocity, excluding roll motion, was used for the AUV dynamics. The control actions were the force and torque in each DOF. For the experiment in the paper, it seems that the AUV was able to follow the DS along a circular path successfully, but the lack of a graph showing the position and velocity over time makes it hard to evaluate how was the pose and velocity error. Additionally, the docked pose was exactly the pose of the DS, the dynamics of the DS was not modeled nor its movement predicted. The actuators of the vehicle were not present in the model and there were no FOV and LOS constraints. It achieved the proposed stochastic MPC, but these missing features could have represented better the docking problem, and achieved better results.

In [23], it was used the same AUV that our research intends to use. A 6DOF MPC was developed as high level controller, to define the trajectory of the vehicle, which is the velocity of the vehicle at each moment for docking. Its cost function goals were to move the vehicle to DS location, reduce energy consumption and generate a smooth control action. The MPC was tested for the nominal case where the pose of the DS and the AUV were known during the docking. Also, even though the model is 6DOF, the control action and error were just evaluated in two degrees of freedom. A contribution of this paper was to use the DS movement prediction as the goal of the MPC. On the other hand, it considers that the docked pose is the same as the DS center, it does not considers the FOV nor the actuators constraints, and the MPC was controlling only the x and z DOF.

In [24], an NMPC for xyz and yaw docking control was developed, it acts as a high and low level, taking DS pose as input and effort as output. The cost function is designed to: reduce the error between the pose of the AUV and docked pose in the DS entrance, reduce the distance between them and have a smooth control action. The constraints of the MPC problem are set to avoid crashing into the seabed or the DS. The MPC also considers the FOV constraint. This research brought some important aspects of the docking problem with the docked pose being different than the DS pose and the FOV constraints. Although, a static DS was used, so even if the paper considers the docked pose, it does not make sense to predict its nonexistent motion.

In [22], a robust 6DOF MPC was developed to dock the AUV in a moving mothership. The goal of the MPC was to make the vehicle achieve the same velocity of the DS and a specific pose in relation the the DS. Besides the constraint used for the robust MPC problem, it was utilized the co-visibility and control effort limit. Although, it does not consider the docked pose in the model and does not predict the movement of the DS. Moreover, the controller outputs are the effort and torque in each DOF of the AUV instead of thrusters effort.

Even though all these papers mentioned in this chapter brought some contribution to AUV docking/rendezvous MPC, the work done by [19], for spacecraft rendezvous is the most important one for the scientific contribution of our thesis. This work describes the MPC of a spacecraft with a gripper that has to catch a rock on a moving asteroid. The predicted error of the spacecraft MPC is calculated by the iterating system of ODEs that describe the relative pose and velocity of the rock in respect to the gripper. That is the

reason why [19] is so important, because it solves the problem that no other AUV docking in mobile DS MPC has ever solved. It considers the relative motion between a point in the target and a point in the robot. Doing an analogy of the spacecraft rendezvous with AUV docking, the spacecraft and its gripper could be the AUV's center and the rock on the asteroid could be the docked pose in the DS. In that manner, using the same logic in the MPC of the spacecraft rendezvous, it will be possible to model and predict the error between the AUV and docked pose states. Furthermore, the paper also includes thruster allocation in the model, thrusters limits constraint and LOS constraint. The only thing missing regarding the problems that this work intends to solve is the co-visibility constraints and the underwater dynamics.

As demonstrated by the analysis of these researches regarding AUV MPC docking, it is possible to conclude that some strategies have been proposed and successfully implemented on simulation. The application of robust MPC, stochastic MPC, the inclusion of the integral of the error in the model have been able to overcome current disturbances, noisy measurement from sensors and modeling errors. Some papers even consider and detail the thrusters limits, LOS and FOV constraints, but none of them considers the three constraints together.

At the time that this work has been released, there is no docking in a mobile DS MPC that considers the docked pose being different from the DS pose. As for the prediction of DS movement, only [21, 23] are using it to generate the goal of the MPC, but only for 3DOF [21] and 2DOF [23] case. These two important aspects of the MPC docking problem have been dealt by [19], but targeting the rendezvous problem of a spacecraft robot instead of an underwater robot, which has different dynamics. Therefore, there is a research gap that can be closed for AUV docking in mobile a DS. Firstly, to consider the prediction of the docked pose movement introduced by [19]. Secondly, to implement of all the three docking constraints in the MPC. The implementation of these constraints and docked pose dynamics can improve AUV docking MPC already in the nominal case, which considers perfect model and perfect sensors measurements.

3.2 DOCKED POSE CONSIDERATION

Almost every AUV docking in a mobile DS paper does not consider the relative docked pose with respect to the DS that the AUV shall be in the end of the docking maneuver [20–23, 34]. Normally, it is assumed that the AUV has to be in the same pose of the DS. Considering that the docked pose is different than the pose of the DS has consequences, not only in the pose goal, but also in the velocity goal. The conversion from the DS dynamics to docked pose dynamics are represented in equation 3.1.

$$\begin{aligned} R_{dp} &= R_{ds} \times R_{dp}^{ds} \\ p_{dp} &= p_{ds} + R_{dp}^{ds} \times p_{dp}^{ds} \\ \omega_{dp} &= R_{dp}^{ds} \times \omega_{ds} \\ v_{dp} &= R_{dp}^{ds} \times (v_{ds} + p_{dp}^{ds} \times \omega_{ds}) \end{aligned} \tag{3.1}$$

Figure 3.1 illustrates in 2D the difference between using a docked pose and a DS pose. For this case there is a blue triangle representing the AUV and a purple one

representing the DS. The rectangles represent the electrical connector of the AUV and DS, that have to be in the same position at the end of the docking. The docked pose is fixed relative to the DS and it is defined in a way that both connectors are in the same place. For the position, rotation and angular velocity, the change from DS to the docked pose represented in equation 3.1 means a geometrical conversion of reference frames. Although, it is different for the linear velocity, since the velocity in the docked pose is the linear velocity of the DS plus the tangential velocity of its position due to the angular velocity of the DS.

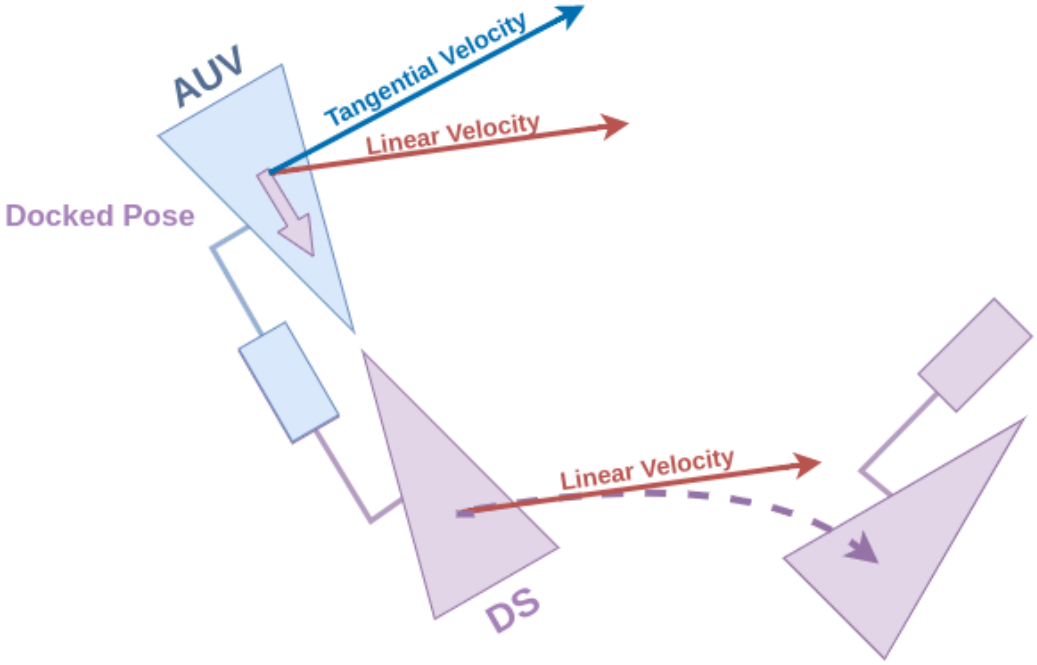

Figure 3.1 Illustration of AUV in docked pose and its linear velocity.

3.3 PREDICTION OF GARAGE MOVEMENT

As stated in Chapter two, the goal of an MPC is to minimize a cost function. For the case of AUV docking in a mobile garage, one important aspect of the cost function is the sum of the error between the AUV states, which are pose and velocity, and the docked pose states. The regular approach is to measure docked pose states, and use it as a setpoint for the whole prediction horizon, but this is not valid, since the garage can be moving and the docked pose states can be changing. It is possible to fix this issue by using the garage model to predict the docked pose future states for the whole prediction horizon, then using the predicted states as the setpoint in the cost function cf. To use the prediction of the DS movement in the cost function, the part representing states errors has to change from 3.2 to 3.3. The only difference is the usage of docked pose prediction

$x_{dp}(n)$ instead of its current measurements $x_{dp}(0)$.

$$cf(x_{auv}, x_{dp}, ph) = \sum_{n=0}^{ph} \|x_{auv}(n) - x_{dp}(0)\| \tag{3.2}$$

$$cf(x_{auv}, x_{dp}, ph) = \sum_{n=0}^{ph} \|x_{auv}(n) - x_{dp}(n)\| \tag{3.3}$$

After changes, this can have a great impact. This impact is illustrated in the Figure 3.2, in which the DS is moving laterally to its right. In case the MPC does not considers the movement of the DS, the optimal trajectory found by the MPC shall be similar to the red one. Otherwise, it would predict a trajectory leading the AUV to the predicted position of the DS, represented by the green curved line.

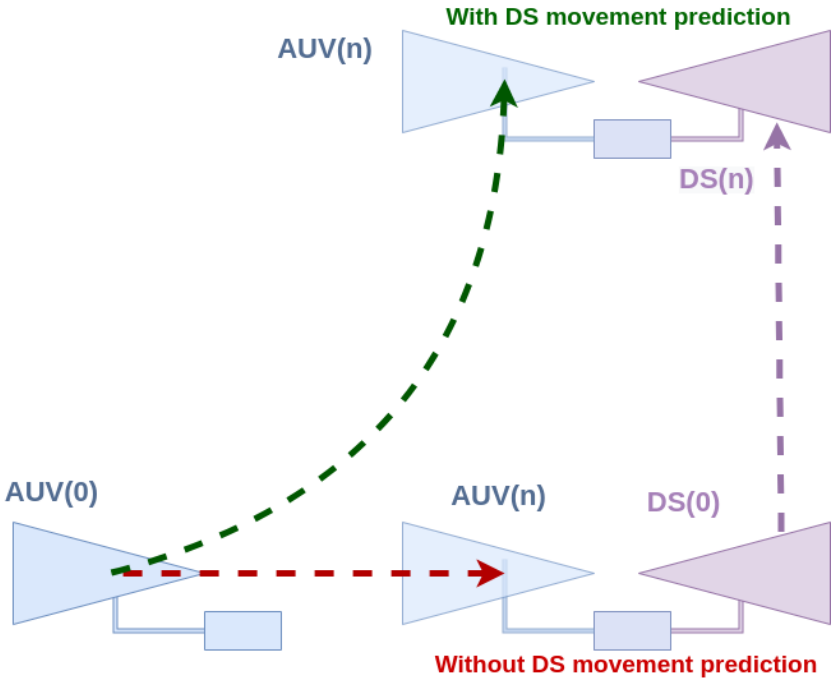

Figure 3.2 Illustration of optimization difference with and without DS movement prediction.

3.4 SOFTWARE IN THE LOOP

SIL test consists in running the code with a simulation to evaluate its performance on a software level. Particularly for MPC in robotics, SIL involves running a simulation of the robot that outputs sensor measurements, receives control input and moves the robot according to its dynamics and environment disturbances. That differs from running a single piece of code with the robot simulation and controller, because in that case the controller can take as much time it needs to output the control action. SIL of an AUV docking MPC forces the controller to solve the optimization problem with a time

constraint. Therefore, with SIL, it is possible to reduce the gap between simulation and implementation in a real robot, by assuring that the MPC solves the problem on time, without compromising its performance.

For SIL implementation of AUV docking MPC, it would require an underwater robot simulator to be connected with the MPC software. Gazebo [35] would be a viable option for simulating the robot, but it lacks underwater actuators, dynamics and sensors simulation. Therefore, UUV simulator [36] will be used in conjunction in Gazebo for the simulation of the robot. Finally, to communicate the MPC software with the simulation, ROS will be used [2]. By combining these three tools, ROS, Gazebo and UUV simulation, it is possible to implement the MPC with SIL to validate the solution.

3.5 CO-VISIBILITY CONSTRAINTS

These constraints are basically geometry boundaries for the docking problem. As it was stated in Chap. 1, the co-visibility constraints are the restrictions for the MPC to maintain the DS in the FOV of AUV's sensors and the AUV in DS's sensors FOV. Both of them basically limits the pose where the AUV have to be in relation to the DS. One way of defining both restrictions is by selecting a polyhedral cone to represent AUV's sensor FOV and DS's sensor FOV.

It is possible to guarantee that the AUV will be inside the DS FOV polyhedral cone or the DS inside AUV FOV polyhedral cone by the usage of halfspaces intersection. For this work, it will be used polyhedral cones with the shape of right rectangular pyramids without their base for these FOVs constraints. Therefore, considering the AUV as a point in space p_{auv} and H_1, H_2, H_3 and H_4 the four internal halfspaces formed by the DS's sensor FOV, if Eq. (3.4) is true, then the point is inside the pyramid [19, 22].

$$p_{auv} \subset H_1 \cap H_2 \cap H_3 \cap H_4 \tag{3.4}$$

The pyramid can be defined by choosing its five vertices, v_1, v_2, v_3 and v_4 being vertices from the base and v_0 the vertex of the top. Considering $\vec{v}_{ab}$ the vector pointing from vertex v_a to v_b, the pyramid's lateral planes norm vectors can be defined as $\vec{n}_1 = \vec{v}_{01} \times \vec{v}_{02}$. With the norm vectors and the vertices it is possible to define the internal halfspaces of the pyramid by Eq. (2.13) if the vertices are positioned in a manner that the norm points to the outside of the pyramid. Fig. 3.3 illustrates this definition of the pyramid vertices and norm vectors for better visualization of the problem.

After choosing the vertices of the DS FOV pyramid, if the Eq. (2.13) is true for $p = p_{auv}$ and for all the four pyramids halfspaces, then Eq. (3.4) is true and the DS FOV constraints are obeyed. For the AUV FOV is similar, but the pyramid is with respect to AUV's sensor FOV and $p = p_{ds}$. Therefore, to include the co-visibility constraints in the AUV docking MPC problem, it is just to include the 8 inequality halfspaces as constraints in the MPC.

3.6 THRUSTER CONTROL AND CONSTRAINT

AUV control can be done in multiple ways. A single controller can be used to send commands to the thrusters with the goal of reaching a specific position, or a chain of

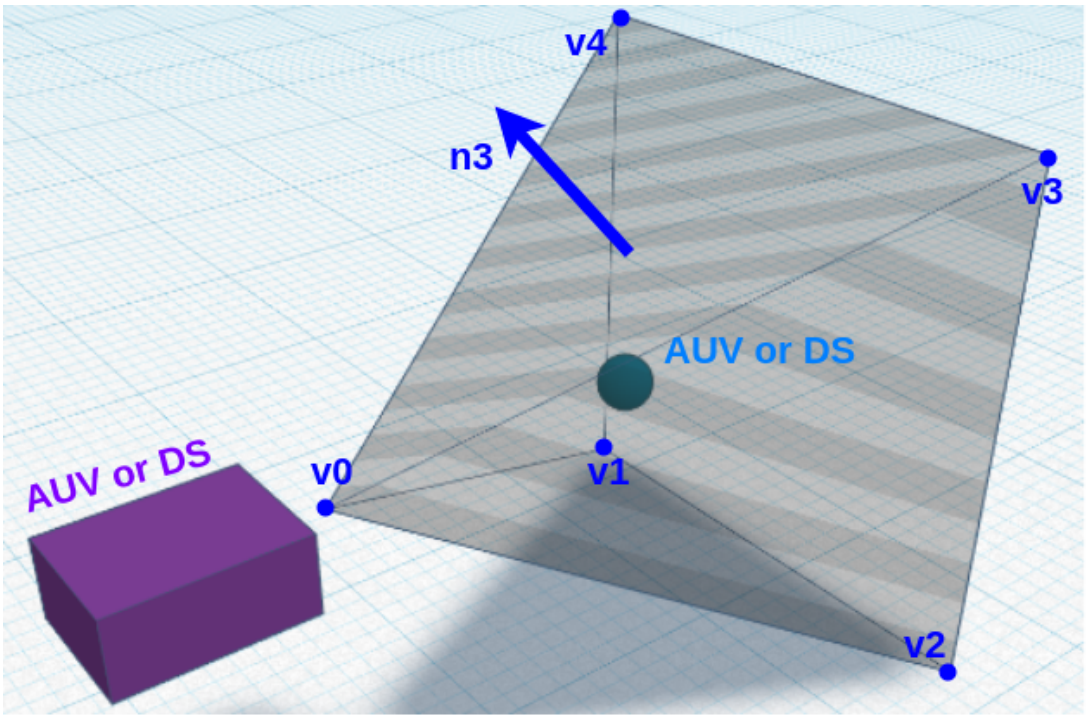

Figure 3.3 Illustration of FOV constraint elements.

different controllers can be used to achieve the same goal. For example, there could be a pose controller that outputs the velocity needed to reach a certain pose. This velocity can be passed to another controller that outputs the effort necessary to reach the desired velocity. Finally, the 6DOF effort would be passed to a controller that send commands to the thrusters to apply the desired 6DOF effort in the robot. At the end of this chain, the single component that is interacting with the robot and that has physical limitations is the thruster. Including the thruster control in the MPC has the benefit of calculating an optimal force for each thruster that it is feasible to apply. To add the thruster control in the MPC it is necessary to replace in (2.5) the 6DOF controlled effort τ for the equation in (2.8). Then it is possible to add the upper and lower boundaries constraints for each control output of the MPC.

Chapter

4

DOCKING MODEL PREDICTIVE CONTROLLER

Although the most important contributions of this thesis MPC were explained in Chap. 3, it was not explained how they all fit together in the MPC and the MPC itself, which is what this chapter intends to do.

4.1 UNDERWATER DYNAMICS IN ORDINARY DIFFERENTIAL EQUATIONS SYSTEM

Through Chap. 2 and 3 the math details for AUV dynamics modeling have been described, but the exact model that the MPC will use for prediction of vehicle movement has not been explained yet. The model will differ from the previous one described in Chap. 2 to facilitate the initial-value problem solution required to predict the behavior of the system in the MPC. Basically, a vector with the derivatives are going to be isolated on the left side, as in Eq. (4.1). Therefore, the acceleration for the dynamic model in Eq. (2.5) has to be left alone in the left side of the equation. Additionally, the replacement of 6DOF control action by the thruster control matrix and thrusters has to be done in dynamic model as well.

$$\dot{x} = f(x, u) \tag{4.1}$$

$$\begin{bmatrix} \dot{\eta} \\ \dot{\nu} \end{bmatrix} = \begin{bmatrix} \dot{p} \\ \dot{\Theta} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} O_{3x1} \\ O_{3x1} \\ (M_A + M_{RB})^{-1} \times (T \times f_n - g(\Theta)) \end{bmatrix} + \begin{bmatrix} R(\Theta) & O_{3x3} \\ O_{3x3} & J(\Theta) \\ (M_A + M_{RB})^{-1} \times (-D_L) \end{bmatrix} \times \begin{bmatrix} \nu \\ \nu \end{bmatrix} \tag{4.2}$$

Almost the same ODEs will be used for prediction of DS movement, the only change is that the control action part which will be eliminated, because the DS is considered as not actuated. The ODEs from Eq. (4.2) are represented as a non linear state space form, so the variables which have derivatives in the left side will still be referred to as states

and the thrusters effort vector as input vector. The state vector has 12 elements, $x_{0:5} = \eta$ and $x_{6:11} = \nu_{0:5}$. The details of how this ODEs model will be used are in Sec. 4.3 of this chapter.

4.2 COST FUNCTION

The cost function for the MPC is the Eq. (4.3). In this Eq., the operator $v^{\circ 2}$ represents a bitwise operation for raising the vector or matrix elements to the power of two. It is the weighted sum over the prediction horizon of the element-wise square sum of input variation u and the error between AUV states x and docked pose states r. The diagonal matrices Q_t, Q and S are respectively the weight for terminal error, for the regular error and for input variation.

$$\begin{aligned} A &= Q_t \times (x_{[ph]} - r_{[ph]})^{\circ 2} \\ B &= \sum_{i=1}^{ph} (Q \times (x_{[i]} - r_{[i]})^{\circ 2}) \\ C &= \sum_{i=1}^{ph-1} (S \times (u_{[i+1]} - u_{[i]})^{\circ 2}) \\ cf &= \sum_{i,j} A_{i,j} + \sum_{i,j} B_{i,j} + \sum_{i,j} C_{i,j} \end{aligned} \tag{4.3}$$

The goal of the AUV docking MPC is to match the pose and velocity of the AUV with the docked pose, therefore, both pose and velocity error are in the cost function. It is important to have separated the weight from terminal and regular error, because the velocity of the AUV only needs to match the velocity of the docked pose in the end of the predicted trajectory. The matching of velocities in the middle of the docking is not ideal since the AUV has to be able to go faster than the docked pose to reach it. That same strategy was adopted by [19].

The variation of control action is also in the cost function to avoid a noisy response. With a real thruster, it could not be able to respond in time to a noisy signal due to its own dynamics that are not considered in the problem. The energy spent is not in the formula thus the MPC do not have a trade off between performance and energy saving. Since docking is a relative short task that requires high precision, there is no need to sacrifice performance in favor of energy saving.

4.3 CONSTRAINTS

Before going into the constraints that are important contributions of this thesis explained in Chap. 3, there are some constraints necessary for the MPC to work that must be detailed. First, it is required to use the ODEs to predict the movement of the vehicle, which is going to be made using multiple shooting method [37]. Multiple shooting enables the computation of future states in parallel instead of sequentially to speed up the controller, but the number of variables in the optimization problem increases. During the experi-

ments in this work, it was observed that multiple shooting was able to reduce in half the time that it takes for the MPC to converge into a solution.

The multiple shooting is done by enforcing the discrete integration of the ODEs to calculate $x_{[i+i]}$. The euler method is going to be used for this initial-value problem solution in the discretization. Therefore, for i from one to ph, for every possible integer value of i, the constraint from Eq. (4.4) [38] will be applied. The prediction of DS movement is described in Sec. 4.4 and it is going to be estimated outside the MPC, because it is not influenced by the AUV movement.

$$x_{i+1} = x_i + \dot{x}_i dt \tag{4.4}$$

The second constraint for the MPC is for the implementation of a control horizon. Since the MPC optimization have the goal of finding only the best control action up to the control horizon step, from the control horizon to the prediction horizon, the MPC shall maintain the same control action. This will be done by enforcing the constraint in Eq. (4.5) for i being an integer starting in $ch + 1$ up to ph.

$$u_{i+1} = u_i \tag{4.5}$$

Other than these required constraints for the MPC problem to be solved, there are the constraints of the docking problem. For the FOV constraint, in each prediction step of the MPC there shall be four inequalities constraints to assure that the DS will be in the intersection of its pyramid half spaces, as described in Chap. 3 by Eq. (3.4). In the AUV FOV case, the center vertex of the pyramid is its camera c, and the vertices of the base can be defined by the camera horizontal α and vertical β half aperture angles, by the following Eq. (4.6).

$$\begin{aligned} v_0 &= p^c_{auv} \\ v_1 &= v_0 + [1, tan(-\alpha), tan(\beta)] \\ v_2 &= v_0 + [1, tan(\alpha), tan(\beta)] \\ v_3 &= v_0 + [1, tan(\alpha), tan(-\beta)] \\ v_4 &= v_0 + [1, tan(-\alpha), tan(-\beta)] \end{aligned} \tag{4.6}$$

After obtaining the vertices of the pyramid, each normal can be calculated simply by the cross product between each pair of basis vertex and central vertex in the pyramid, as it is shown in Eq. (4.7). Although, before generating the normals, the vertices have to be converted from the local frame of the robot to the world frame by the following Eq. (4.8). With the pyramid normals and central vertex in the global frame, it is possible to create the constraint for halfspace intersection with the Eq. (4.9). In other words, the MPC has to find the pose of the vehicle in which the normals and central vertex of the

FOV pyramid in the global frame obeys the constraint of halfspaces intersection.

$$\begin{aligned} n_1 &= (v_1 - v_0) \times (v_2 - v_0) \\ n_2 &= (v_2 - v_0) \times (v_3 - v_0) \\ n_3 &= (v_3 - v_0) \times (v_4 - v_0) \\ n_4 &= (v_4 - v_0) \times (v_1 - v_0) \end{aligned} \tag{4.7}$$

$$v_n = p_{auv} + R(\Theta_{auv}) \times v_n \tag{4.8}$$

$$\begin{aligned} n_1 \times p_{ds} &\leq n_1 \times v_0 \\ n_2 \times p_{ds} &\leq n_2 \times v_0 \\ n_3 \times p_{ds} &\leq n_3 \times v_0 \\ n_4 \times p_{ds} &\leq n_4 \times v_0 \end{aligned} \tag{4.9}$$

For the DS FOV constraint the same process has to be applied. The only difference are that in the Eqs. (4.6), (4.8) and (4.9) the p_{auv} and p_{ds} have to switch places. Although, even if this process is the same and that the halfspaces intersection constraints equations are applied in the same way, for the DS FOV constraint, the normals and pyramid center vertex are fixed known values for every prediction. Therefore, the MPC optimization has to find an AUV position that respect the Eq. (4.9). Since for this case the states variables are directly in the constraint and there is no need for computing normals and vertex transformation at every optimization iteration, this constraint is less computationally expensive than the AUV FOV constraint.

Finally, there are the thrusters constraints, which are simply lower and upper bounds for setting minimum and maximum effort of each thruster. Considering i as an integer from one to the number of thrusters, the Eq. (4.10) shall be added as a constraint to the MPC problem, in which lb_i and ub_i are respectively the minimum and maximum force that each thruster can apply.

$$lb_i \leq u_i \leq ub_i \tag{4.10}$$

4.4 FIXED AND DYNAMIC GOAL MPC PIPELINE

Two MPCs will be developed in this work, the fixed goal and dynamic goal MPC. The fixed goal is how the most papers on the literature have applied docking MPC, without considering that the garage move and thus the setpoint (reference) r is the same value through all prediction horizon. The dynamic goal is the MPC considering the model and the prediction of DS movement to define the value of r for each step of the prediction horizon. The main difference between both approaches does not happen inside the optimization problem, it happens in the definition of the docked pose reference that will be set for the optimization to solve. Both cases are represented in the diagram from Fig. 4.1, which shows that the difference between these MPCs is just prediction of the DS movement stage.

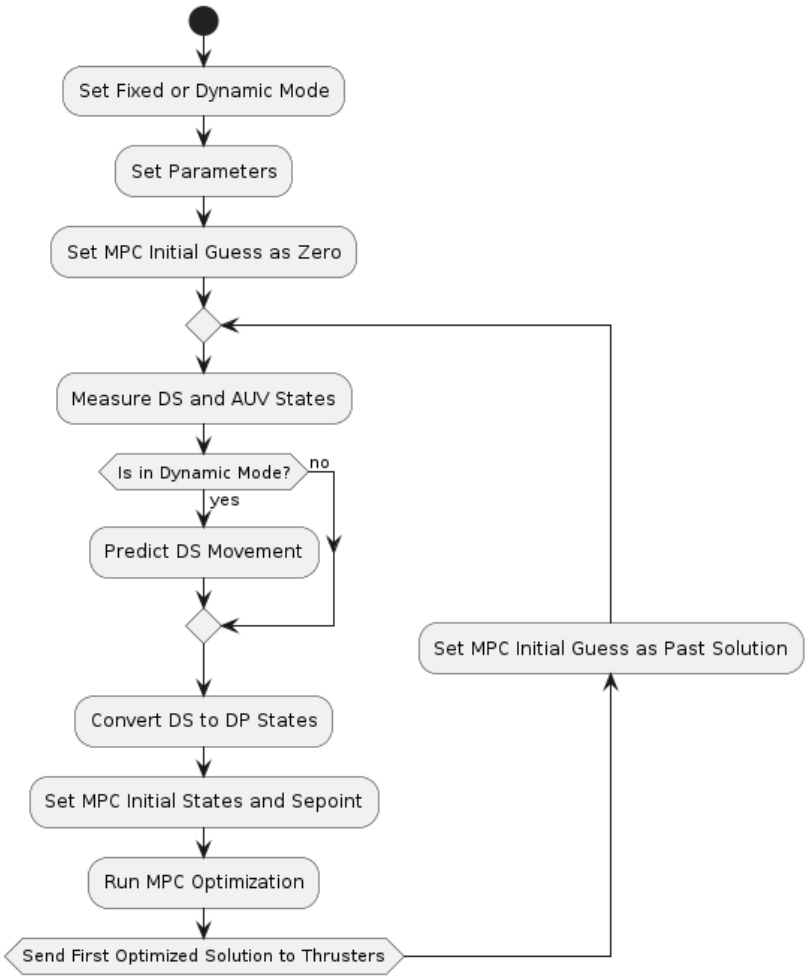

Figure 4.1 Sequence diagram of fixed and dynamic goal MPC.

At the first iteration of the MPC software, the parameters shall be parsed. These parameters are everything that the MPC needs, such as the parameters for DS and AUV models, docked pose coordinates, constraints parameters, gain matrices of the MPC and MPC mode. These parameters were partially specified in this section and will be fully presented in the results section. Afterwards, the MPC initial guess for the first iteration is set as zero. Then, the main loop of the MPC starts by measuring DS and AUV states. After that, if the dynamic mode is activated, the DS movement will be predicted, otherwise, the current DS measurement will be repeated for all ph steps. Then, the DS pose and velocity are converted to the actual goal of the AUV states, which is the docked pose. Finally, the MPC initial states are set as the measurements, and the setpoint as the DP current or predicted states, then the MPC optimization is executed. The optimized control action values of the first iteration is sent to the thrusters and their values for the whole prediction are set as a initial solution for the next loop. This loop keeps happening until the software stops by user termination. It does not stop when the vehicle reaches the DS because the DS can still be in movement and there is no locking mechanism in

the simulation.

Chapter 5

SYSTEM DESCRIPTION

All mathematics aspects and contributions of this thesis explained so far need to be tested and implemented in a robot simulation to prove its effectiveness. This will be done by using SIL which was introduced in Chapter 3. The implementation of SIL will be further detailed in this chapter.

5.1 SIMULATION AND SOFTWARE

As stated in Chapter 3, Gazebo simulation Software in conjunction with the UUV simulator plugins will be used for simulating the robot with its sensors and thrusters. To simulate the docking of the AUV into a DS, two robots are going to be used inside Gazebo, one representing the AUV and the other the DS. As it is in the software component diagram in Fig. 5.1, each robot publishes its odometry information, which has its 6DOF pose and velocity. Each robot can also receive the force command for every thruster it has.

The MPC software will be running in ROS exchanging information with the simulation to control the robot and measure its current states. In the Fig. 5.1, it is possible to visualize that there are three nodes running, each node is a program running in a loop, receiving and publishing information. The MPC node, is computing all the logic detailed in Chapter 4, using the odometry of the DS and the AUV to get their model states and to output the optimal force to be applied by each thruster of the AUV for docking. The disturbance simulation node is responsible for calculating the 6DOF forces to be applied in the DS to create some DS motion profiles for the MPC validation. Finally, the DS thruster manager node is a standard node already available inside UUV Simulator. This node just receives the 6DOF force command from the disturbance simulation node and does the matrix multiplication of Eq. (2.8) to output the 1D force in each thruster inside the simulation.

5.2 AUV AND DS DESCRIPTION

Both DS and AUV simulated robots in Gazebo will be the same for simplicity of the problem. The MPC developed by this work is designed for a 6DOF AUV model that

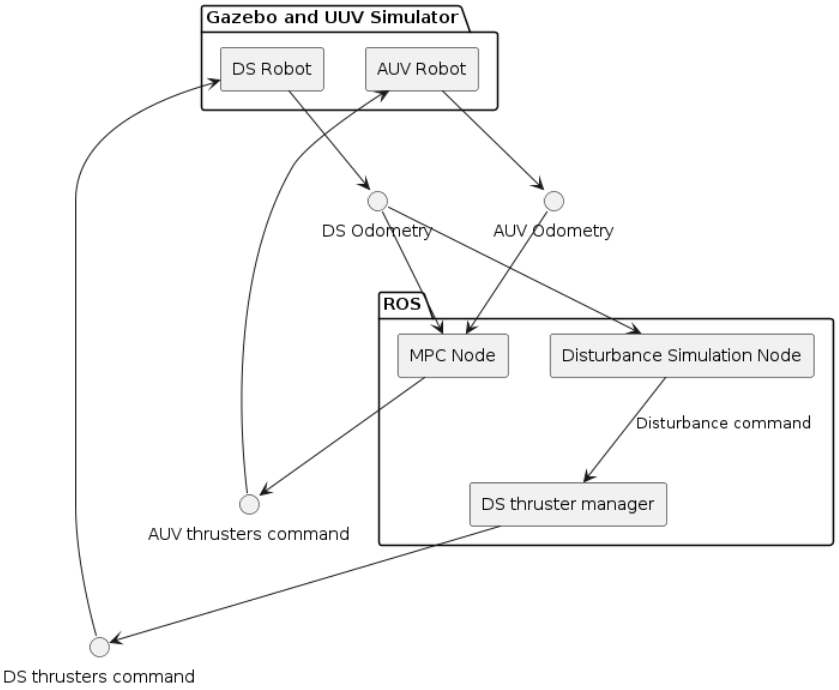

Figure 5.1 Software component diagram.

Table 5.1 BlueROV parameters table

Parameter	Description	Values	Unit
M_A	Added Mass Matrix Diagonal	5.5, 12.7, 14.6, 0.12, 0.12, 0.12	kg
M_{RB}	Rigid Body Mass Matrix	11.5, 11.5, 11.5, 0.16, 0.16, 0.16	kg
D	Linear Damping Matrix	-4.03, -6.22, -5.18, -0.07, -0.07, -0.07	kgm

only have thrusters as actuators. Since BlueROV2 Heavy robot is an AUV of this format, its model will be used for the validation of the proposed MPC. Additionally, the BlueROV is an open source project, and it had its 3D model, thrusters and sensors specs available online. By using the specs of a real robot in the simulation, the tests become more realistic, proving that if the MPC software succeeds in simulation, it would be able to succeed in a real robot.

Gazebo and UUV simulator require some parameters for the simulation of the robot. These parameters are the ones in the kinematics and dynamic model of an AUV, which was explained in Chap. 2. The description and values for the BlueROV parameters are listed in Tab. 5.1. Most of the parameters, like the diagonal damping and added mass matrices were obtained from [23], a paper that not only applied MPC for BlueROV docking, but also estimated the parameters of the real vehicle.

For simplification, the vehicle was considered with neutral buoyancy, therefore, it has the weight equal to the buoyancy. Moreover, the COB and the COG of the vehicle were consider to be in the same position, in that way, there are no hydro-statics forces present. After inserting all these parameters in Gazebo and UUV simulator, loading the

3D model of the BlueROV and specifying the pose of every thruster, it is possible to start the simulation. Fig. 5.2A shows a photo of the real BlueROV2 and, Fig. 5.2B and 5.2C shoes how it is its representation inside the Gazebo and a simulated environment.

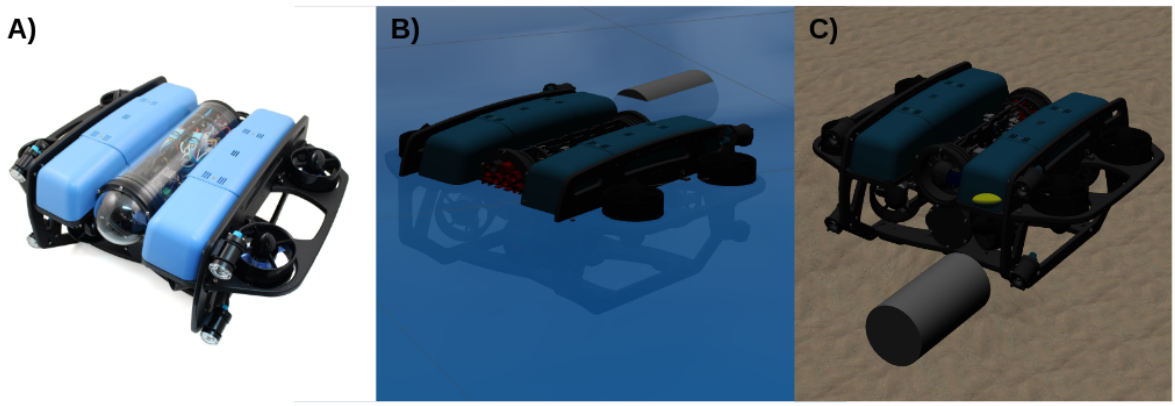

Figure 5.2 BlueROV2 Heavy real picture (A) and screenshot from simulation (B and C).

The thrusters poses were measured from its 3D model. The robot has eight thrusters, four in the horizontal plane tilted 30 degrees from the x axis and four on the vertical plane aligned with the z axis. The configuration of the thrusters is illustrated in Fig. 5.3. The matrix T in Eq. (2.8) was calculated using an UUV simulator plugin, that output this matrix using the thrusters pose in relation to the center of the vehicle. The TCM of BlueROV is a $6x8$ matrix, because it is converting each one the eight thrusters force to the 6DOF force in the frame of the vehicle. This matrix can be visualized in Eq. (5.1).

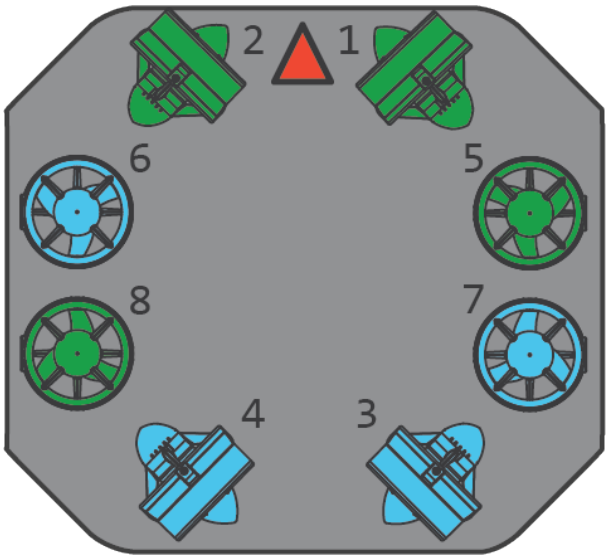

Figure 5.3 Illustration of thrusters placement inside BlueROV Heavy 2.

$$\tau = \begin{bmatrix} 0.866 & 0.866 & -0.866 & -0.866 & 0.000 & 0.000 & 0.000 & 0.000 \\ -0.500 & 0.500 & 0.499 & -0.499 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 1.000 & 1.000 & 1.000 & 1.000 \\ -0.0145 & 0.0145 & 0.0145 & -0.0145 & -0.213 & 0.223 & -0.213 & 0.223 \\ -0.025 & -0.025 & 0.025 & 0.025 & -0.124 & -0.124 & 0.116 & 0.116 \\ -0.162 & 0.153 & -0.157 & 0.166 & 0.000 & 0.000 & 0.000 & 0.000 \end{bmatrix} \times f_n \tag{5.1}$$

All these parameters and dynamic model described for Gazebo were also used in the DAE for the MPC. The parameters and the model from Gazebo and MPC might not be exactly the same because Gazebo has its own way of iterating the model, parsing parameters and even a different programming library to model DAEs.

The last parameter for the AUV and DS are their connectors pose. AUV and DS connectors are the gray cylinders in Fig. 5.2. They are simply a marker for the viewer to clearly see when the AUV has docked. The AUV is going to be considered as docked when the AUV and DS connectors are closer than a specified tolerance, which will be defined in Chap. 6. AUV connector ac has $\eta^{ac}_{auv} = [0.5, 0.2, 0, 0, 0, \pi]$ and the DS connector dc has the $\eta^{dc}_{ds} = [0.5, -0.2, 0, 0, 0, \pi]$.

Chapter 6

RESULTS

6.1 TESTS AND MPC DEFINITIONS

The MPC problem was defined in Chap. 4, and the model used for prediction of AUV and DS motion was detailed in Chap. 5. However, there are still some MPC parameters that need to be set. One of them is the docked pose of the AUV in relation to the DS. It has to be defined so that the DS connector (DC) and the AUV connector (AC) are in the same position, and rotated 180 degrees in yaw, when the AUV docks. This can be achieved by the Eq. (6.1), which gives a result of $\eta_{dp}^{ds} = (1, 0, 0, 0, 0, \pi)$ for the connectors defined in Chap. 5.

$$\begin{aligned} \eta_{ac}^{dc} &= (0, 0, 0, 0, 0, \pi) \\ \eta_{dp}^{ds} &= \eta_{dc}^{ds} \otimes \eta_{ac}^{dc} \otimes (\eta_{ac}^{auv})^{-1} \end{aligned} \tag{6.1}$$

Other MPC parameters that have to be defined are regarding the constraints. DS's and AUV's sensors FOV and AUV's thrusters were decided based on the real parameters of the BlueROV. All eight thrusters are the same T200 model with the maximum thrust being close to 40 N at 16 V [39], thus the MPC maximum and minimum value for thrust will be -40 N and 40 N respectively. The sensor in the AUV and DS will be a Sony IMX322 camera, which has a horizontal FOV angle of 80 degrees and vertical FOV angle of 64 degrees [40]. The cameras where placed in the front of the AUV and the DS at the same position $p_c^{auv} = (0.25, 0, 0)$ and $p_c^{ds} = (0.25, 0, 0)$.

The MPC weights, which are defined by the matrices Q, Q_t and S in Eq. (4.3), are not normalized. Their values have to be selected considering the magnitude of the error. The weights associated with pose error are set as zero in terminal states Q_t, because the MPC has the goal to get the AUV closer to the docked pose in every movement, not only on the last step. A important reminder is that the Q matrix is also a weight for the last step, therefore the AUV tries to match the final pose as well when we assign this weight for pose error, but it is does not focus on the terminal error.

The velocities error weights only received non-zero values for the terminal states Q_t. That was done because the AUV needs to be able to speed up or slow down to reach the

docked pose and only has to match the docked pose velocity in the end of the docking maneuver. Q highest value is the weight associated with the pitch angle, because it was observed during experiments, that the AUV has the tendency of inclining in pitch to reach the docked pose faster. This happens because the AUV has four thrusters in the top and it is faster to move vertically than horizontally. The roll and yaw angles error also have higher weights than position error because their error values are smaller. According to these definitions made through a series of tests, the weights for the MPC were defined as Eq. (6.2).

$$\begin{aligned} Q &= diag(0,0,0,0,0,0,1,1,1,3,8,4) \\ Q_t &= diag(1.4,1.4,1.4,1.4,1.4,1.4,0,0,0,0,0,0) \\ S &= diag(1,1,1,1,1,1,1,1) \end{aligned} \tag{6.2}$$

The prediction and control horizon as well as the time-step for MPC prediction was different for each case and it will be discussed in the next sections of this chapter. The simulation software Gazebo enables the user to select a real time factor to slow down the simulation, it is a value from zero to 1 that defines if the percentage of real time simulation. For example, if the simulation real time factor is 0.5, it takes 2 seconds to pass one second on the simulation. The simulation real time factor was set to 0.1 to avoid breaking the MPC software.

Regarding docking evaluation, for the vehicle to be considered docked, we defined that it needs to be less than 4 cm in XYZ from the DS. We chose this metric based on the results obtained by [23], which used the same AUV for docking as we are using. In contrast to a real AUV docking, our simulation lacks a locking mechanism. Therefore, the AUV will be considered as docked as soon as it meets all docking criteria at once, even if the vehicle later slightly drifts away from the DS. That is because in a real environment, there would be a lock to fix the AUV inside the DS as soon as it reached the threshold.

6.2 GRAPHICAL DEBUG TOOLS AND CONSTRAINTS VALIDATION

The co-visibility constraints are spatial constraints, which makes them easier to be evaluated with graphical tools. ROS framework has the Rviz software [41] that can be used to create visualization tools within the framework. For checking that the FOV pyramids vertices and normals where correctly calculated, a plugin for drawing the co-visibility constraints was developed in this work. The validation of these FOV pyramids can be observed in the Fig. 6.1. In this Figure, there are two BlueROVs. The pink pyramid BlueROV is representing the AUV and its FOV. The blue pyramid BlueROV is representing the DS and its FOV. It is possible to see that the pyramids and normals were correctly plotted in Fig. 6.1 A and B images, since they are in front of the vehicles in the correct position and orientation with all normals pointing to the outside of the pyramid. In Fig. 6.1 C and D images, it is possible to see that that the normals forms a ninety degrees angle with the pyramid's planes, which it is also correct.

In the MPC software, it is possible to know when an optimal value was found or not. To ensure that the co-visibility constraints were working, the MPC was turned on and then pose disturbances were applied to the vehicles. It was observed that whenever the

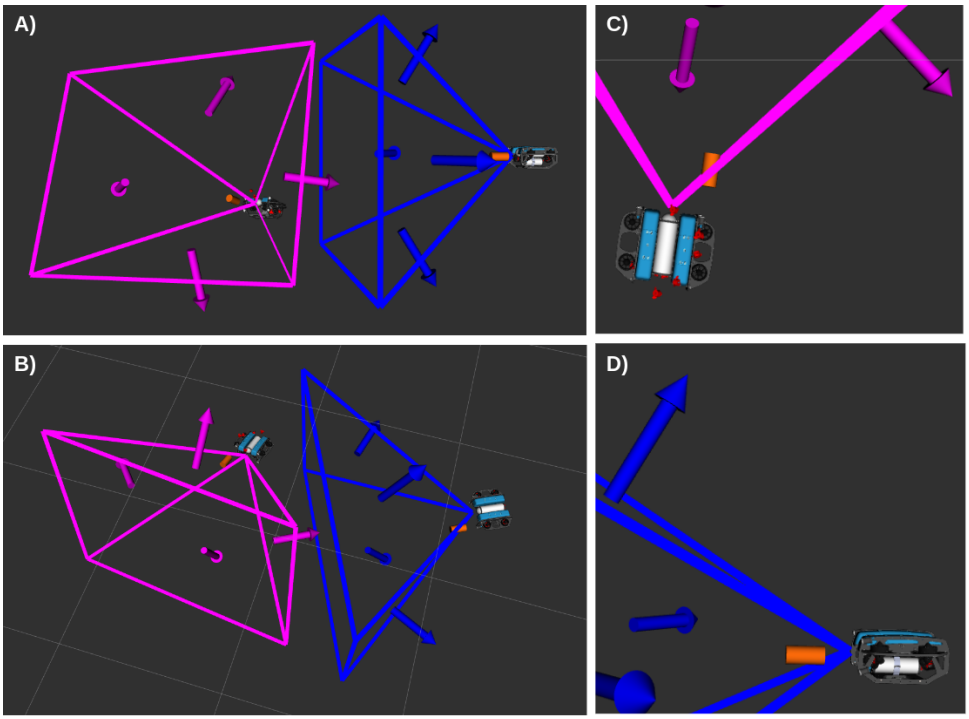

Figure 6.1 FOV pyramid and normals in global frame.

AUV was outside of the blue pyramid or when the DS was outside of the pink pyramid, the MPC could not find an optimal solution. Therefore, the co-visibility constraint was successfully implemented.

Unfortunately, the visualization of 3D models, as in Fig. 6.1, is computationally expensive and does not affect MPC performance. Therefore, for MPC simulation visualization, two rectangle prisms are going to be used instead of vehicles 3D models. A purple prism representing the AUV and a blue prism representing the DS, as in Fig. 6.2.A. DS and AUV connectors are going to be represented as cylinders of the same color as the its vehicle prism. The propellers 3D models do not reduce computer performance, thus they are displayed as they are in red for both vehicles. Moreover, the normals are not going to be displayed to avoid confusion, as it can be seen in Fig. 6.2.B. Still in the same image, it is shown two green curves with their origin in AUV and DS center. The curve starting in AUV center represents the optimal path calculated by the MPC, and the curve starting in DS center represents the prediction of its movement. For all the test cases the visual debug can be seen in the video in this link: `https://www.youtube.com/playlist?list=PL-mEqQgXoq_QGB126zWjCqyrweL2i9Soi`.

For the validation thrusters effort it was simply observed that for all the test cases there was no violation of maximum and minimum thrusters limit, which will be shown in the next section.

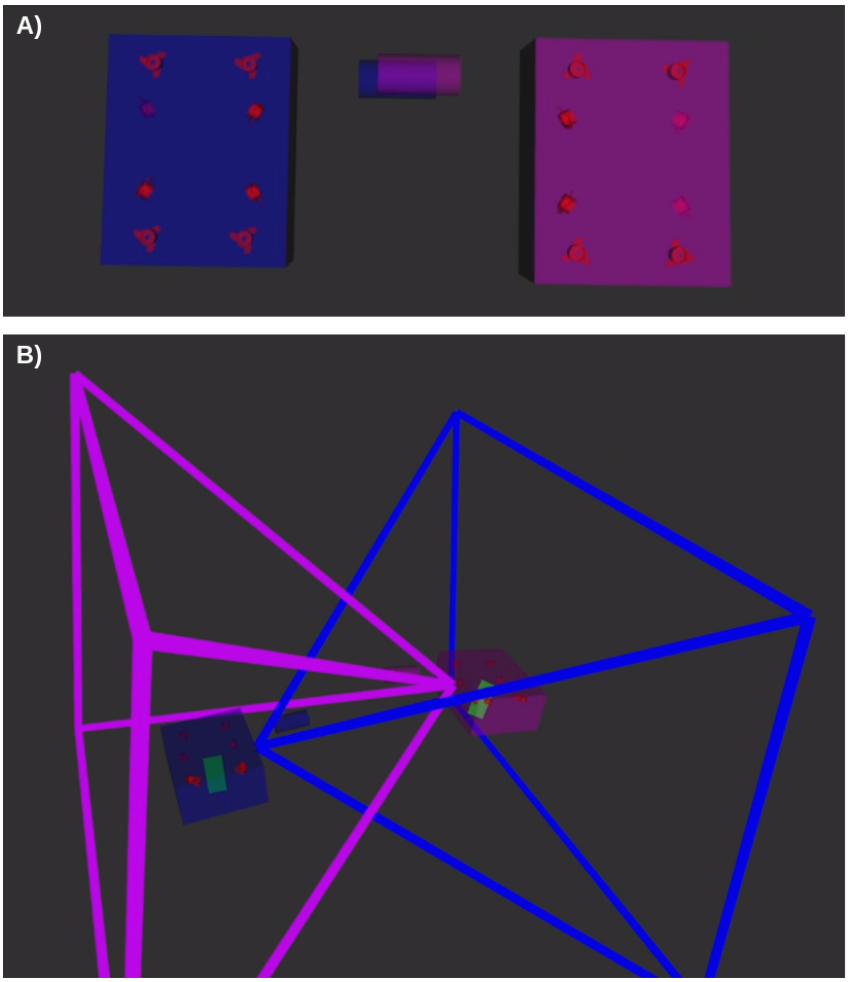

Figure 6.2 Visualization example in runtime.

6.3 STATIC DOCKING

This case is simply to check the functionality of the code and the implementation of constraints. For this case, the timestep was set to 0.1 s, and the prediction horizon was set as five. The short prediction horizon was used because the computer could not find the optimal control action on time if the prediction horizon was greater. Since the prediction horizon was already small, the control horizon was set to be equal to the prediction horizon. The results for this case can be seen in the graphs from Fig. 6.3. For this and all the other results graphs, the MPC goal was represented as dashed curves for each one of the twelve states, while the robot states were displayed as regular lines.

It is possible to conclude that even with a small prediction horizon, the MPC was able to successfully dock the AUV in 2.29 s, respecting the thrusters limits and the co-visibility constraints. Just for this static case the thrusters information is displayed in Fig. 6.4. This information is not very useful for understanding the control problem, because the thrusters are not aligned with the 6DOF of the robot. Although, one thing that can be observed in the plot is that the forty newtons limits defined in Sec. 6.1 were respected.

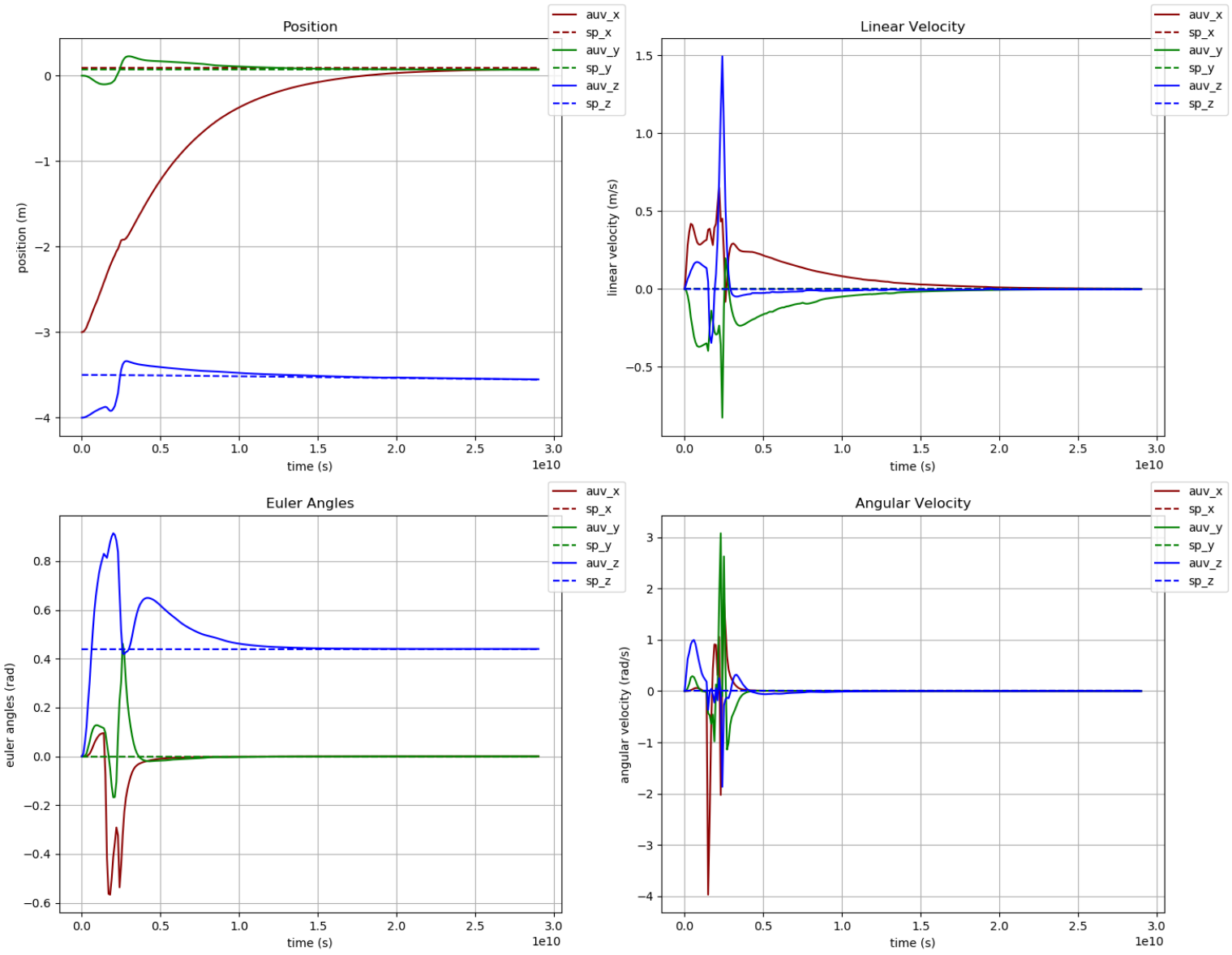

Figure 6.3 Static case docking results.

6.4 VERTICAL OSCILLATORY DS MOVEMENT DOCKING

For this case, the disturbance simulation node was used to move the DS up and down, simulating an oscillatory disturbance. The node was programmed to apply 5 N of force downwards, until the DS had descended two meters, then to apply 5 N upwards, until the DS ascended two meters. That behavior was kept in loop until the end of the simulation. The first test with this disturbance was also made using the same parameters as last tests, with prediction and control horizon as five and timestep set to 0.1 s, but this time the DS movement prediction was enabled. The results of this experiment are displayed in the Fig. 6.5.

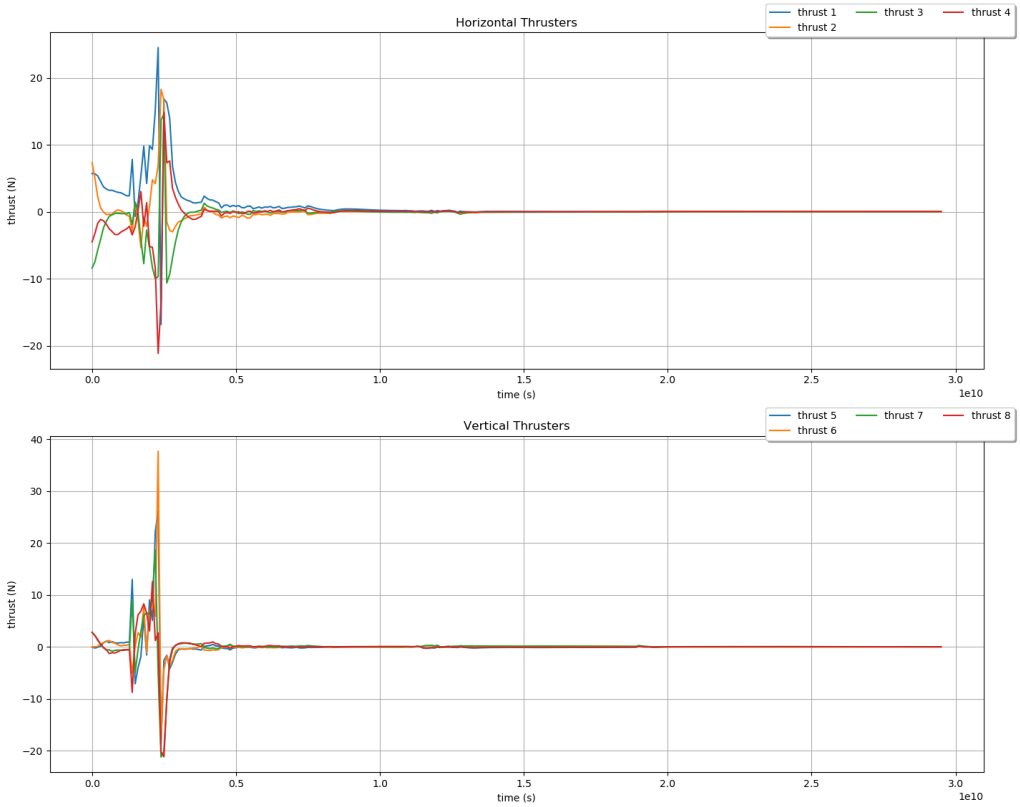

Figure 6.4 Static case docking thruster effort.

Even with a small prediction horizon, the controller was able to dock in 1.63 s. It was also able to follow the movement of the garage, but without respecting the 4cm threshold for docking. An interesting aspect of this test was the ability of the controller to keep the co-visibility constraints. In the Fig. 6.6, it is possible to notice that the MPC is tilting the AUV and moving it upwards in order to respect the AUV FOV constraint. Another important aspect of this test was to validate the work initial proposal of combining the docking constraints and DS movement prediction into a single MPC. Although, the most important contribution of this work is the prediction of the relative docked pose, which could not be evaluated properly without a bigger prediction horizon. Therefore, to increase the prediction horizon, the FOV constraint for the AUV will be disabled for the next tests. The FOV constraint for the DS is still going to be active, as well as the thrusters limits.

After disabling the AUV FOV constraint, it was possible to increase the prediction horizon to 25 steps. The timestep of the MPC prediction was reduced to 0.02 s, in order to reduce integration errors. The control horizon was set to 15, which was decided by trial and error. It was observed that with this control horizon the AUV movement was smoother and that the optimizer could always find a solution to the problem. Two tests for the same vertical oscillation were made, one with the DS motion prediction disabled and another with the prediction enabled. The first test has the results plotted in Fig. 6.7 and the second in Fig. 6.8.

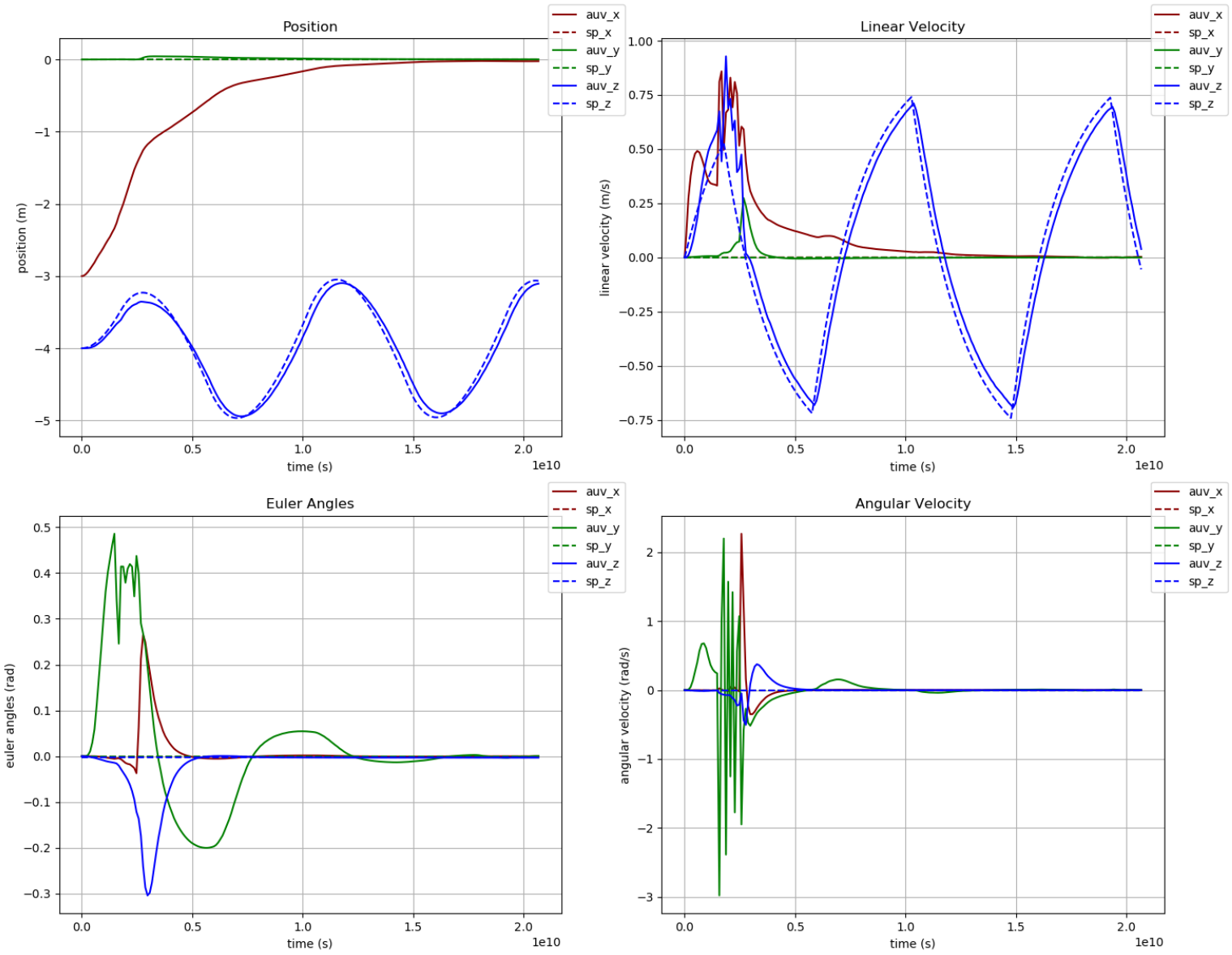

Figure 6.5 Vertical oscillation test with co-visibility constraint results.

With these two tests, enabling and disabling the DS motion prediction, it is possible to see the impact of it in the docking MPC. The MPC with the prediction enabled is able to better track the velocity and position of the goal, which is clear by comparing the position and velocity in axis z from both plots. The euler angles and angular velocity

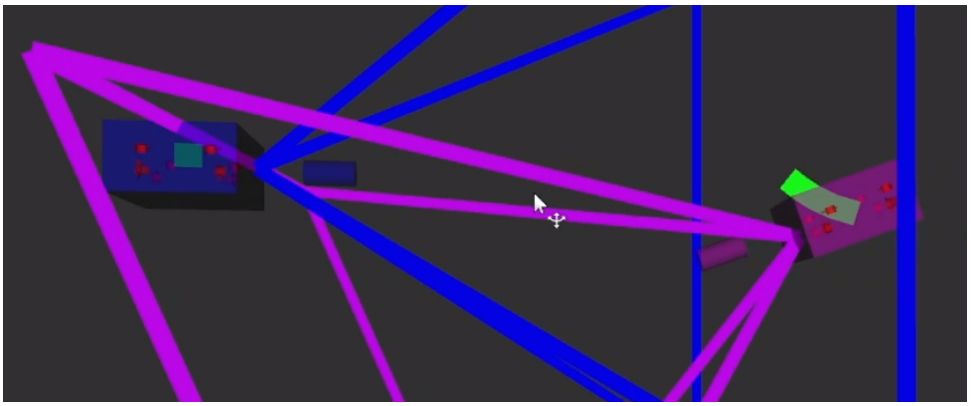

Figure 6.6 MPC forcing the vehicle to obey the co-visibility constraints.

errors were also smaller for the case with target prediction enabled. Furthermore, the motion of the vehicle was smoother for the case with prediction enabled, as can be observed by comparing in both plots the euler angles, angular velocities and the x and y position. Moreover, it took 7.50 s for reaching the docked state without the prediction and 4.25 s with the prediction enabled, thus the prediction of DS movement reduced the docking time by 43.3%. All these advantages with the DS motion prediction did not add computation time for the MPC. The average solving time of the optimization for the MPC with DS motion prediction was 0.04 s and without was 0.07 s. An important reminder is that the prediction of DS movements occurs before the MPC iteration, it just change the setpoints given to the MPC. For this case specifically the prediction of DS movement seems to have facilitated the optimization problem. Regarding the time that it takes to reach the goal, there is no apparent difference. It is possible to observe in the plots that all the states take approximately the same amount of time to reach their goals.

6.5 ROTATION AND TRANSLATION DS MOVEMENT DOCKING

After testing the static and vertical DS movement case, the case missing is the DS movement in the XY plane while also a rotating. To simulate this behavior, the disturbance simulation node was used again. For this case, during the first 1.5 s, a force of -1 N was applied in the x direction, a force of 4 N was applied in the y direction and a torque of 1 Nm was applied around the z axis. In a real DS towed to an USV, this disturbance in the DS can be caused by the USV pulling it. The same prediction horizon, control horizon and timestep of last vertical oscillation test cases were used. It was also tested the case with the DS motion prediction disabled and enabled. The result of the first case can be seen in Fig. 6.9 and the result of the second case in 6.10.

The difference between predicting and not predicting DS motion was very similar to the vertical oscillation case. There are still no difference related to how fast the AUV reaches its goals. Although, it is clear from the position and yaw errors in the plots that the error is again smaller in the case that the DS motion is being predicted. Regarding computational time, the results were different from the vertical oscillation case. For this case, it took an average of 0.1 s to solve the optimization for with the DS motion

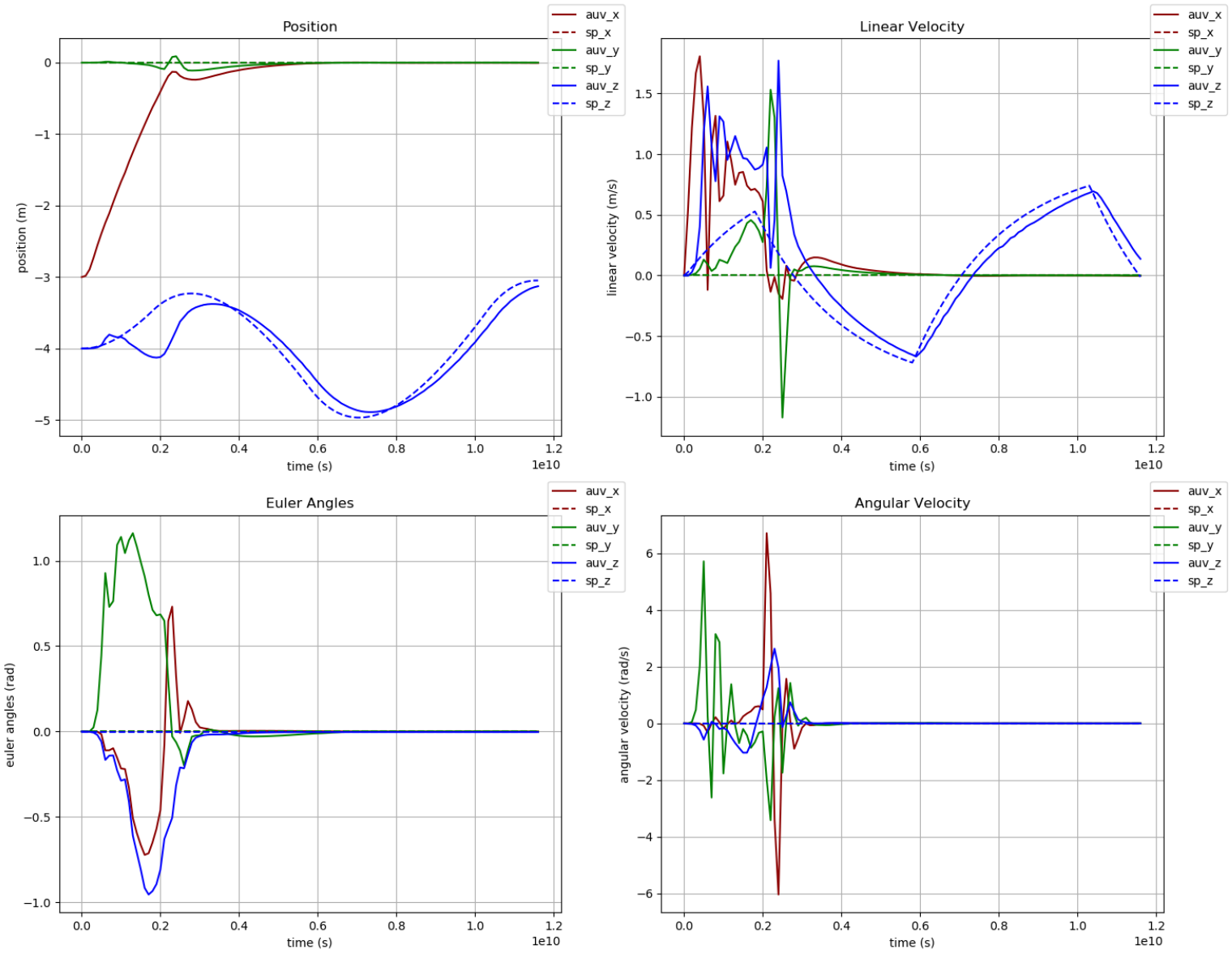

Figure 6.7 Vertical oscillation test without AUV FOV constraint and without DS motion prediction.

prediction and 0.07 s for the case without the prediction. For this case, it is also faster to dock with the prediction of the DS enable. It took 1.60 s to dock without DS motion prediction and 1.11 s to dock with the DS motion prediction, which corresponds to a

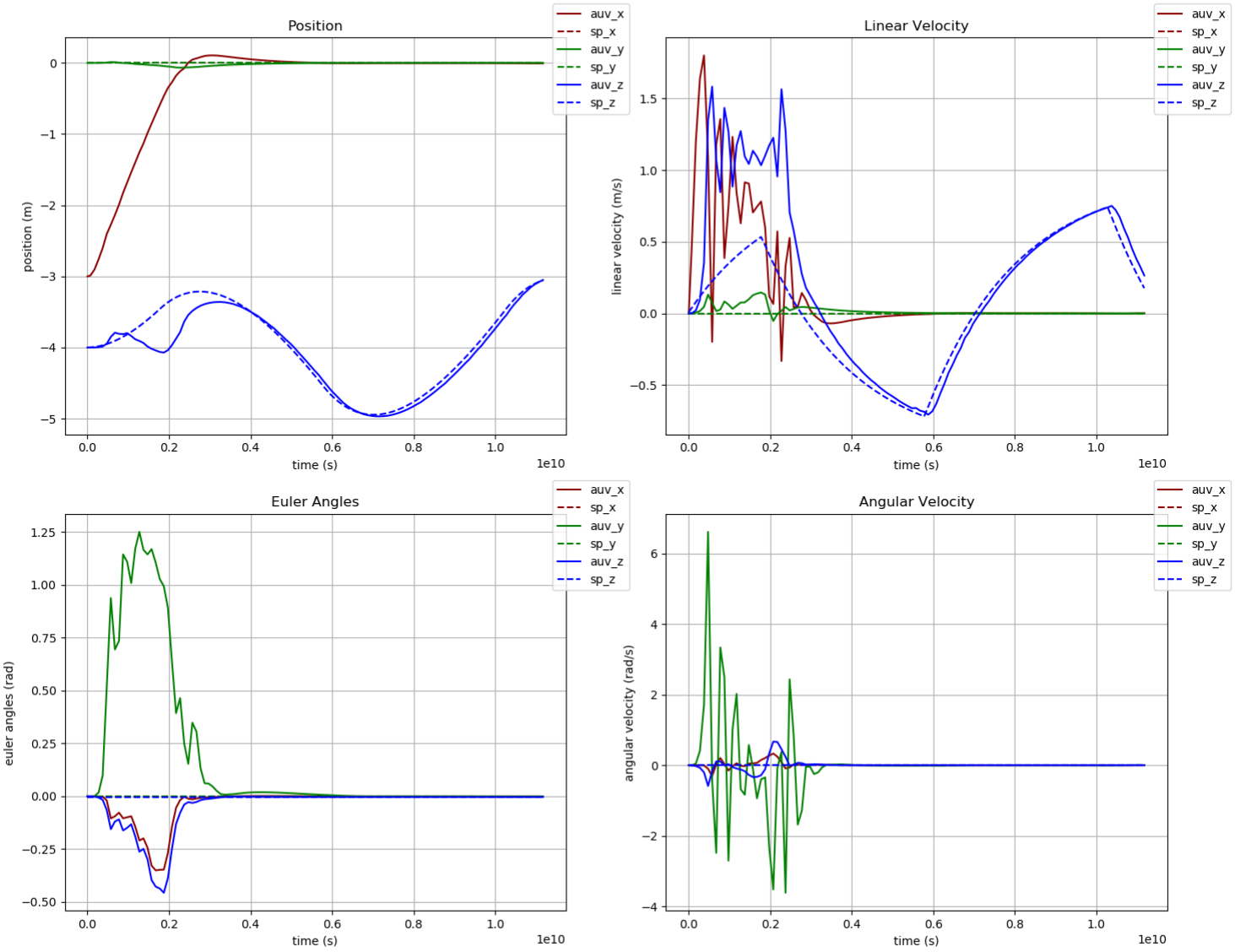

Figure 6.8 Vertical oscillation test without AUV FOV constraint and with DS motion prediction.

reduction of 30.6 % in the docking maneuver duration.

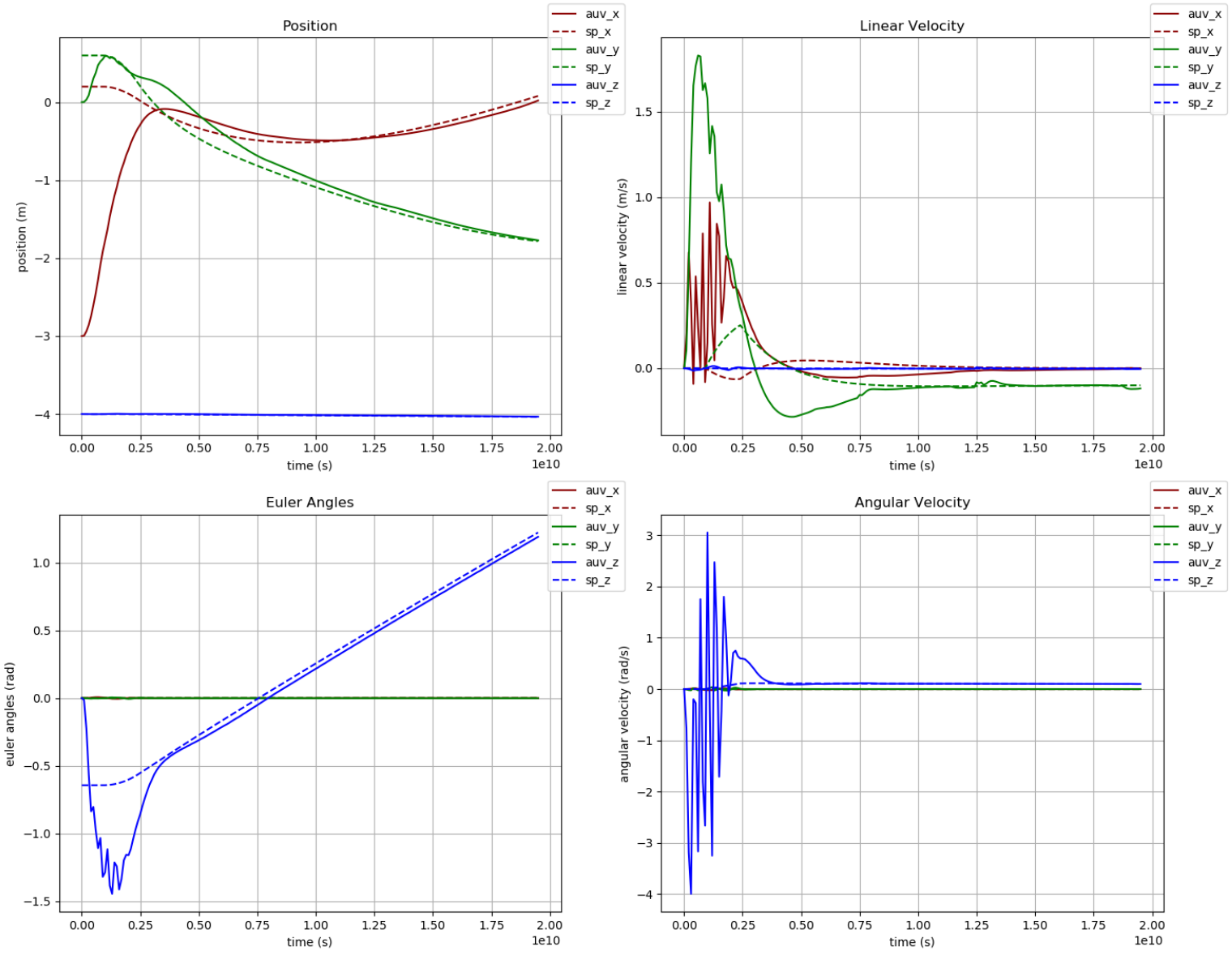

Figure 6.9 Lateral rotation test without AUV FOV constraint and without DS motion prediction.

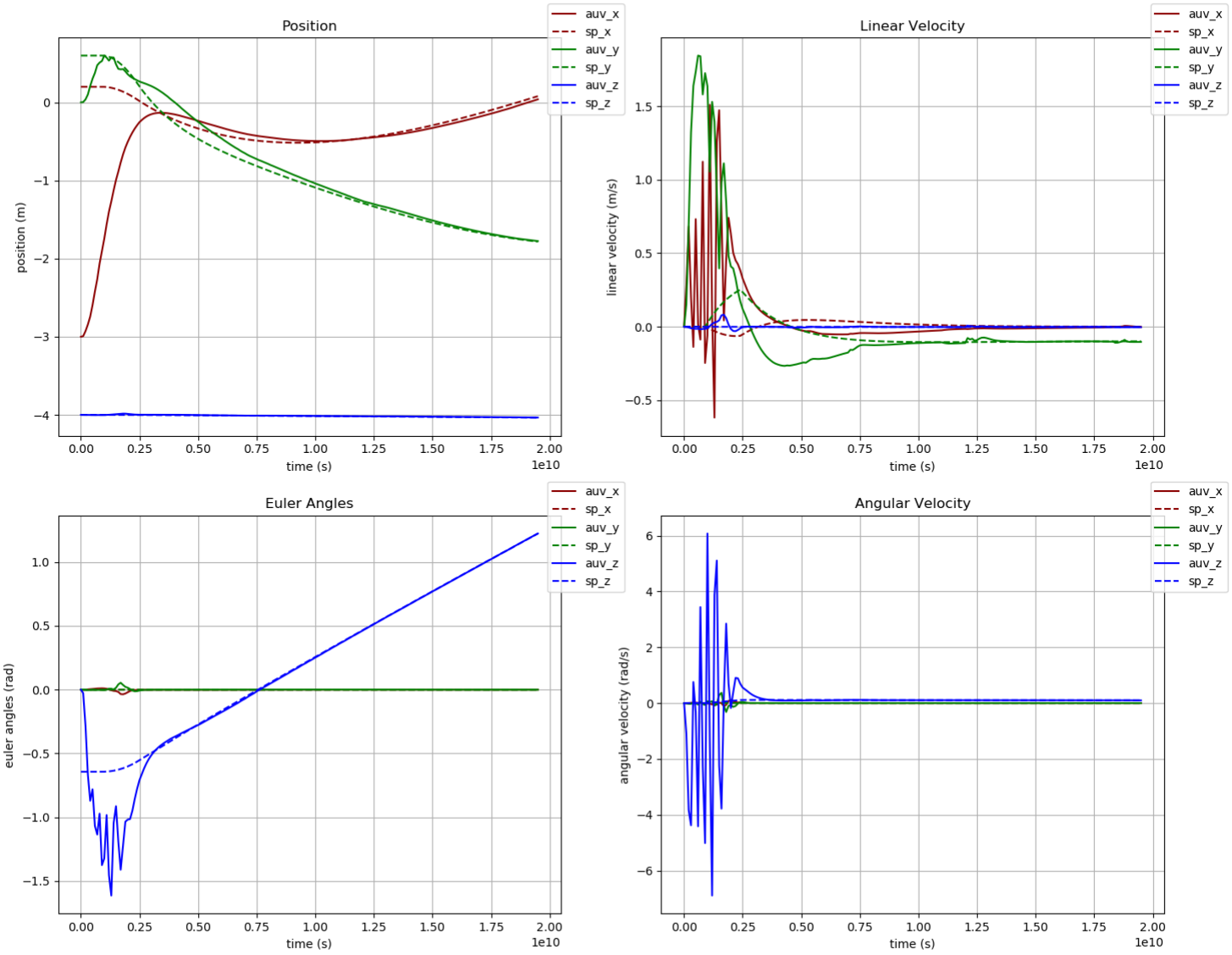

Figure 6.10 Lateral rotation test without AUV FOV constraint and with DS motion prediction.

Chapter 7

CONCLUSION

The goal of this research was to develop an MPC that would act directly on the thrusters of a 6DOF AUV to dock it in a mobile DS, while using the DS movement prediction for a more precise docking. The MPC developed in [19] for the spacecraft rendezvous problem was successfully adapted to the underwater docking problem. The proposed solution increases the accuracy of the MPC, since the prediction of DS motion actually reduces the pose and velocity error for AUV docking. Moreover, the time that it takes for the vehicle to dock it is also reduced when using DS motion prediction.

Not only this main goal was achieved, but also other problems related to the docking maneuver were also solved. The co-visibility problem was solved by adding the half-spaces intersection inequalities as MPC constraints functions. It was clear from the predicted path plotted in the video in this playlist `https://www.youtube.com/playlist?list=PL-mEqQgXoq_QGB126zWjCqyrweL2i9Soi`, that the MPC ensures the co-visibility constraint and manages to keep the AUV inside the DS FOV constraint in all test cases. The thrusters limits were also added as inequality constraints and they were respected by the controller. Therefore, the DS motion prediction MPC for AUV docking is still compatible with the previous already implemented solutions. However, all constraints implemented together were computationally expensive and could only be applied for a small prediction horizon, thus our work still needs to find solutions for speeding up the MPC solving for real time implementation.

In addition to the basic MPC docking results, it was noted that we could cut the MPC computation time in half by using multiple shooting rather than single shooting for MPC discretization. Another finding is that the MPC computation time does not appear to be impacted by the DS pose prediction, as it did for one test case and not for the other test case.

Even though all problems for an fully autonomous docking of a 6DOF AUV in a mobile DS are not solved, this work was another step towards this direction. It was a step towards full autonomous marine robotics operation and towards the reduction of the duration, costs as risks in marine robotics operations.

7.1 FUTURE WORK

The proposed MPC was successfully implemented, but there are many aspects that can be improved. One of them is the time for MPC optimization solution, it took more than 0.1 s to solve the problem with all the proposed constraints enable and with DS motion prediction. To speed up the solution of the MPC problem, some strategies can be adopted. The kinematics model can use quaternions instead of euler angles, that would not only avoid the gimbal lock problem, but also increase the speed of the dynamics model simulation, because with quaternions there is no need for computing sines and cosines at every iteration. Moreover, the optimization library utilized can be changed from casadi [3] to ACADOS [42], which uses C code generation and a different optimization solver for better performance. Apart from speeding up the MPC, there are some enhancements that can be done to have a better MPC software. One of them is the normalization of weights in the MPC, thus instead of choosing them considering the magnitude of the states and control variables, it would be possible to define them based on the states and control variables importance. Another enhancement would be the addition of constraints for co-visibility as slack variables to avoid breaking the MPC execution, when they cannot be respected.

Overall, this work was developed to show the importance of docked pose motion prediction and to create an MPC capable of ensuring the main constraints of the docking problem. Henceforth, it is possible to use this MPC with the speed up suggestions to enable docking of a 6DOF AUV in a mobile DS based on camera detection, since now it is possible to guarantee that the DS or the AUV will stay in each other FOV. To enable docking of an real AUV with noisy signals and delays, it is possible to use this work alongside with other MPC strategies, such as the addition of error integral in the model and the implementation of robust or stochastic MPC.

BIBLIOGRAPHY

[1] Musa Morena Marcusso Manhães, Sebastian A. Scherer, Martin Voss, Luiz Ricardo Douat, and Thomas Rauschenbach.
UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation.
In *OCEANS 2016 MTS/IEEE Monterey*. IEEE, sep 2016.

[2] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Ng.
Ros: an open-source robot operating system.
In *ICRA 2009*, volume 3, 01 2009.

[3] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl.
CasADi – A software framework for nonlinear optimization and optimal control.
Mathematical Programming Computation, 11(1):1–36, 2019.

[4] Robert D Christ and Robert L Wernli Sr.
The ROV manual: a user guide for remotely operated vehicles.
Butterworth-Heinemann, 2013.

[5] Marine Technology Society.
Rov brief history.
Accessed in 2022-06-03: `https://rov.org/history/`.

[6] Candace Reno.
Scuba diving dangers and risks.
Accessed in 2022-06-03: `https://www.wetsuitwearhouse.com/blog/scuba-diving-dangers-risks/`.

[7] SeaDrone.
Why it is faster and cheaper to use rovs than human divers.
Accessed in 2022-06-03: `https://seadronepro.com/blog/why-is-it-faster-and-cheaper-to-use-rovs-than-human-divers`, 2019.

[8] Jessie Y. C. Chen, Ellen C. Haas, and Michael J. Barnes.
Human Performance Issues and User Interface Design for Teleoperated Robots.
IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), 37(6):1231–1245, November 2007.

[9] Jonatan Scharff Willners, Ignacio Carlucho, Tomasz Łuczyński, Sean Katagiri, Chandler Lemoine, Joshua Roe, Dylan Stephens, Shida Xu, Yaniel Carreno, Èric Pairet, Corina Barbalata, Yvan Petillot, and Sen Wang.
From market-ready ROVs to low-cost AUVs, August 2021.

[10] J. Jalbert, J. Baker, J. Duchesney, P. Pietryka, W. Dalton, D.R. Blidberg, S. Chappell, R. Nitzel, and K. Holappa.
A solar-powered autonomous underwater vehicle.
In *Oceans 2003. Celebrating the Past ... Teaming Toward the Future (IEEE Cat. No.03CH37492)*, pages 1132–1140 Vol.2, San Diego, CA, USA, 2003. IEEE.

[11] Nicholas C Townsend.
Self-powered autonomous underwater vehicles: results from a gyroscopic energy scavenging prototype.
IET Renewable Power Generation, 10(8):1078–1086, September 2016.

[12] A.M. Yazdani, K. Sammut, O. Yakimenko, and A. Lammas.
A survey of underwater docking guidance systems.
Robotics and Autonomous Systems, 124:103382, February 2020.

[13] VItor H. Pinto, Nuno A. Cruz, Rui M. Almeida, and Carlos F. GoncCalves.
ALARS - Automated Launch And Recovery System for AUVs.
In *OCEANS 2018 MTS/IEEE Charleston*, pages 1–6, Charleston, SC, October 2018. IEEE.

[14] Tao Song and Rong Zheng.
Autonomous Launch and Recovery System for Underwater Towed Vehicle Based on USV.
In *2019 3rd International Conference on Robotics and Automation Sciences (ICRAS)*, pages 132–136, Wuhan, China, June 2019. IEEE.

[15] Brian R. Page and Nina Mahmoudian.
AUV Docking and Recovery with USV: An Experimental Study.
In *OCEANS 2019 - Marseille*, pages 1–5, Marseille, France, June 2019. IEEE.

[16] Edoardo I. Sarda and Manhar R. Dhanak.
A USV-Based Automated Launch and Recovery System for AUVs.
IEEE Journal of Oceanic Engineering, pages 1–19, 2016.

[17] G.J.S. Rae and S.M. Smith.
A Fuzzy Rule Based Docking Procedure For Autonomous Underwater Vehicles.
In *OCEANS 92 Proceedings@m_Mastering the Oceans Through Technology*, volume 2, pages 539–546, Newport, RI, 1992. IEEE.

[18] John Wang and Edwin Olson.
AprilTag 2: Efficient and robust fiducial detection.
In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2016.

[19] Di Malladi, S ; Cairano, A Weiss, Bharani P Malladi, Stefano Di Cairano, and Avishai Weiss.
Nonlinear model predictive control of coupled rotational-translational spacecraft relative motion, 2019.
URL `http://www.merl.com`.

[20] Zhepinz Yan, Peng Gong, Wei Zhang, Wenhua Wu, and Saibo Gao.
AUV docking control based on stochastic model predictive control.
In *Global Oceans 2020: Singapore – U.S. Gulf Coast*, pages 1–4, Biloxi, MS, USA, October 2020. IEEE.

[21] Mikkel Cornelius Nielsen, Tor Arne Johansen, and Mogens Blanke.
Cooperative Rendezvous and Docking for Underwater Robots using Model Predictive Control and Dual Decomposition.
2018.

[22] Wei Zhang, Yanbin Teng, Haitian Chen, and Chenxi Yu.
On the robust model predictive control method of dynamic positioning to line for uuv recovery.
In *OCEANS 2016 MTS/IEEE Monterey*, pages 1–6. IEEE, 2016.

[23] Jonathan Wallen, Nicholas Ulm, and Zhuoyuan Song.
Underwater Docking System for a Wave Energy Converter based Mobile Station.
In *OCEANS 2019 MTS/IEEE SEATTLE*, pages 1–8, Seattle, WA, USA, October 2019. IEEE.

[24] Kai Shi, Xiaohui Wang, and Huixi Xu.
On the Offset-free Nonlinear Model Predictive Control for AUV Docking *.
In *2021 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*, pages 117–122, Beijing, China, September 2021. IEEE.

[25] Gerhard Venter.
Review of optimization techniques.
Encyclopedia of Aerospace Engineering, 2010.

[26] Max Schwenzer, Muzaffer Ay, Thomas Bergs, and Dirk Abel.
Review on model predictive control: an engineering perspective.
The International Journal of Advanced Manufacturing Technology, 117:1–23, 11 2021.

[27] Leigh McCue.
Handbook of marine craft hydrodynamics and motion control [bookshelf].
IEEE Control Systems Magazine, 36(1):78–79, 2016.

[28] Gianluca Antonelli.
Underwater robots.
Springer, 2021.

[29] Society of Naval Architects and Marine Engineers.
Nomenclature for treating the motion of a submerged body through a fluid.
Technical and Research Bulletin, 1950.

[30] L. El Ghaoui.
Hyper-Textbook Optimization Models and Applications.
Springer, 2021.

[31] Salimzhan A. Gafurov and Evgeniy V. Klochkov.
Autonomous Unmanned Underwater Vehicles Development Tendencies.
Procedia Engineering, 106:141–148, 2015.

[32] R. Stokey, M. Purcell, N. Forrester, T. Austin, R. Goldsborough, B. Allen, and C. von Alt.
A docking system for remus, an autonomous underwater vehicle.
In *Oceans '97. MTS/IEEE Conference Proceedings*, volume 2, pages 1132–1136 vol.2, 1997.

[33] Ben Allen, Tom Austin, Ned Forrester, Rob Goldsborough, Amy Kukulya, Greg Packard, Mike Purcell, and Roger Stokey.
Autonomous Docking Demonstrations with Enhanced REMUS Technology.
In *OCEANS 2006*, pages 1–6, Boston, MA, USA, September 2006. IEEE.

[34] Joan Esteba, Patryk Cieslak, Narcis Palomeras, and Pere Ridao.
Docking of Non-Holonomic AUVs in Presence of Ocean Currents: A Comparative Survey.
IEEE Access, 9:86607–86631, 2021.

[35] N. Koenig and A. Howard.
Design and use paradigms for gazebo, an open-source multi-robot simulator.
In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2149–2154 vol.3, 2004.

[36] Musa Morena Marcusso Manhães, Sebastian A. Scherer, Martin Voss, Luiz Ricardo Douat, and Thomas Rauschenbach.
UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation.
In *OCEANS 2016 MTS/IEEE Monterey*. IEEE, sep 2016.

[37] Guido Sánchez, Marina Murillo, Lucas Genzelis, Nahuel Deniz, and Leonardo Giovanini.
Mpc for nonlinear systems: A comparative review of discretization methods.
In *2017 XVII Workshop on Information Processing and Control (RPIC)*, pages 1–6, 2017.

[38] Richard L Burden, J Douglas Faires, and Annette M Burden.
Numerical analysis.
Cengage learning, 2015.

[39] Bluerobotics.
T200 thruster specifications.
Accessed in 2022-09-12: `https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster-r2-rp/`, .

[40] Bluerobotics.
T200 thruster specifications.
Accessed in 2022-09-12: `https://bluerobotics.com/store/sensors-sonars-cameras/cameras/cam-usb-low-light-r1/`, .

[41] HyeongRyeol Kam, Sung-Ho Lee, Taejung Park, and Chang-Hun Kim.
Rviz: a toolkit for real domain data visualization.
Telecommunication Systems, 60:1–9, 10 2015.

[42] Robin Verschueren, Gianluca Frison, Dimitris Kouzoupis, Jonathan Frey, Niels van Duijkeren, Andrea Zanelli, Branimir Novoselnik, Thivaharan Albin, Rien Quirynen, and Moritz Diehl.
acados—a modular open-source framework for fast embedded optimal control.
Mathematical Programming Computation, 14:147–183, 2022.