TÉCNICO LISBOA

Cooperative Motion Control Using Hybrid Acoustic-Optical Communication Networks

Rodrigo Tavares Rego

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor: Prof. António Manuel dos Santos Pascoal

Examination Committee

Chairperson: Prof. João Fernando Cardoso Silva Sequeira
Supervisor: Prof. António Manuel dos Santos Pascoal
Member of the Committee: Prof. Fernando Manuel Ferreira Lobo Pereira

February 2021

Declaration

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

Acknowledgments

The realization of this thesis was possible due to the opportunity provided by Prof. António Pascoal. I am very grateful to have been considered to study and research under his notorious experience and knowledge, and also note that his guidance was always indispensable.

It is also important to mention the invaluable input given by Nguyen Hung, who was always very patient on helping me overcome some implementation obstacles. I'm grateful for his accessibility and for pushing me to reach the answer on my own.

I wouldn't have come this far without the support of my family. I thank my parents, who helped me keep my sanity in check, my very stubborn older brother Ricardo, who must be finally at ease now that this thesis is completed, and my chill younger brother Miguel, who indulged me in any escape from reality during quarantine times. I also thank Laura for her good nature towards any issue I had.

To all my good friends, especially Francisco, Beatriz, and Ana, who made sure I didn't stray off too much and who helped me recall that having a social life is also important.

Finally, I acknowledge the support from the Institute for Systems and Robotics (ISR), under which this research was made possible.

Resumo

Existe atualmente interesse em desenvolver grupos de veículos marinhos autónomos para exploração dos oceanos. Isto requer o uso de robots que cooperam entre si através da transmissão de dados com base numa rede de telecomunicação. Para este propósito, as comunicações acústicas têm sido a escolha por excelência. Contudo, para operações que envolvem veículos que operam a curtas distâncias entre si, existe atualmente um grande interesse no uso de comunicações óticas, capazes de taxas de transmissão maiores. Posto isto, o objetivo desta dissertação é viabilizar o uso de comunicações óticas e utilizar ambas tecnologias durante missões cooperativas realizadas debaixo de água.

Tendo em vista este objetivo, a primeira parte da dissertação propõe algoritmos para controlo de um veículo marinho autónomo. Nomeadamente, um controlador de malha interna que aceita referências de um controlador seguidor de caminho de malha externa, baseado na cinemática. Simulações que usam o modelo de veículos do tipo MEDUSA ilustram a implementação dos sistemas de controlo propostos, para diferentes técnicas de seguimento de caminho.

A segunda parte da dissertação é dedicada aos assuntos de controlo cooperativo. Um controlador de coordenação é projetado e comunicações discretas entre veículos são consideradas, tendo em conta as comunicações acústicas e mecanismos de desencadeamento que refletem as suas limitações.

Por fim, a terceira parte da dissertação propõe um algoritmo que permite viabilizar as comunicações óticas, que se resume a alcançar o alinhamento entre os feixes óticos de um par de veículos. O algoritmo proposto é ilustrado por resultados que mostram o alinhamento dos feixes óticos dos modems de comunicação durante uma missão cooperativa.

Palavras-chave: Controlo de Veículo, Seguidor de Caminho, Controlo Cooperativo, Comunicações Óticas, Veículos Marinhos Autónomos

Abstract

There is currently widespread interest in the development of groups of autonomous underwater vehicles for ocean exploration. This calls for the deployment of robots that can act in cooperation by exchanging data over communication networks. For this purpose, acoustic networks have been so far the choice par excellence. However, for operations involving vehicles operating at close range, there is currently a flurry of activity on the use of optical-based communication systems, capable of higher transmission rates. As a result, the goal of this thesis is to make optical communications viable and to use both communication technologies during underwater cooperative missions.

Motivated by this goal, the first part of this thesis proposes single vehicle motion control algorithms. Namely, an inner-loop heading controller that accepts references from an outer-loop path following controller designed at the kinematic level. Simulations with the MEDUSA-class vehicles illustrate the implementation of the proposed control systems, for different path following approaches.

The second part of this thesis is dedicated to cooperative motion control issues. A coordination controller is designed and discrete communications among the vehicles are considered, taking into account the acoustic communications and triggering mechanisms that reflect their limitations.

Lastly, the third part of the thesis proposes an algorithm to make optical communications viable, which ultimately boils down to achieving optical beam alignment between a pair of cooperative agents. The proposed algorithm is illustrated by results that show optical beam alignment being reached for a pair of vehicles in a cooperative formation.

Keywords: Motion Control, Path Following, Cooperative Control, Optical Communications, Autonomous Underwater Vehicles

Contents

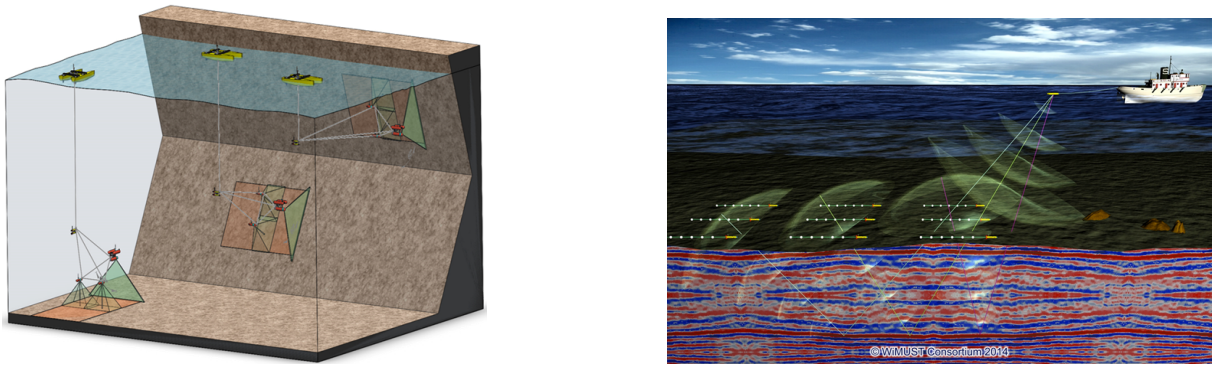
List of Tables

List of Figures

Chapter 1

Introduction

There is currently widespread interest in the development of groups of autonomous underwater vehicles for ocean exploration. This calls for the deployment of robots that can act in cooperation by exchanging data over communication networks. For this purpose, acoustic networks have been so far the choice par excellence. However, for operations involving vehicles operating at close range, there is currently a flurry of activity on the use of optical-based communication systems, capable of higher transmission rates. As a result, the objective of this work is to use a combination of acoustic and optical communication systems to perform cooperative navigation and control of multiple autonomous vehicles of the MEDUSA class, property of IST/ISR. The optical modems to be installed on the vehicles have been designed and tested at IST/ISR.



(a) MORPH Project (2012-2016). (b) WiMUST Project (2015-2018).

Figure 1.1: Example of ocean exploration projects that relied on acoustic communications.

1.1 Problem Statement

The problem to be addressed is one of cooperative navigation and control of autonomous underwater vehicles using hybrid acoustic-optical communication systems. Acoustic systems are the pervasive solution to communications, but their price is quite high and transmissions rates low. For this reason, there is currently considerable interest in the possible use of optical modems, as affordable units with the capability to transmit data at higher rates. Optical communication systems have, however, narrow

directivity patterns, posing a challenge to achieve optical beam alignment while steering the vehicles. As a result, the emphasis of this work lies in the development of algorithms for cooperative maneuvers, using data exchanged among the vehicles, by resorting to acoustic modems, when the vehicles are distant from each other and to broadcast their coordination states, and optical modems, when the vehicles operate at close range. In the latter case, the optical modems can also serve the dual purpose of transferring large volumes of data from one vehicle to another.

Ultimately, it should be shown in this work that optical communications can actually be viable, assuming that proper optical beam alignment can be solved with the proposed mechanism. With this in mind, this thesis elaborates a first approach on a complex problem applied to cooperative underwater missions. Simulations are used to illustrate all the proposed algorithms, based on a nonlinear model of the MEDUSA-class vehicles. Provided that the proposed solution shows promising results in a simulation environment, testing this first approach on a real scenario, with real MEDUSA vehicles, should come next, as future work on the subject.

1.2 State of Art

In order to accomplish the cooperative mechanism that is capable of aligning the optical beams properly to establish optical communications, while the acoustic configuration is being used for longer ranges and to broadcast the agents' states, one must start by revisiting existing control strategies for single vehicle motion.

The strategy to achieve single vehicle motion control, in this thesis, starts with designing an inner-loop heading controller by state-feedback. In succession, an outer-loop for path following is designed, which will feed a reference to the inner-loop. This inner-outer loop decoupling strategy is commonly adopted as it offers flexibility. All of this is done considering a linearised system of the MEDUSA vehicle at a particular speed of operation, and designing a controller that is then applied, with proper modifications, to the original nonlinear system.

The underlying issue of actively steering the vehicle and assuring optical beam alignment lies, first and foremost, in designing path following strategies, starting from a path parametrized in space. In this thesis, three path following algorithms are discussed, accompanied by some results that support the revised theory.

The first strategy, Line of Sight algorithm, is tightly related to the geometry of the problem at hand, needing a different formulation for straight lines and circumferences, for instance, even though the basic underlying principles are the same. For this reason, this algorithm is promoted as the most straightforward path following approach, once the geometric problems are solved. The basic reasoning is that of considering a lookahead distance, measured along a tangent to the path, with the starting point defined as the orthogonal projection of the vehicle's position on the path, and control the vehicle's heading towards the final point on that tangent.

Secondly, the method by P. Maurya proposed in [1] is discussed. This guidance algorithm is designed as a controller that drives the cross-track error to zero, commanding the yaw of the vehicle. This method

also has the advantage of accounting for ocean currents, by means of adding an integral term in the virtual input – to tackle the fixed bias introduced by the current. The convergence of this algorithm boils down to the assumption of inner-loop instantaneous heading command, which can be ensured by manipulating the gains of the obtained second order system, outer-loop controller, to our need.

Lastly, the approach proposed by A. Micaelli and Samson [2] reveals to be the most complex as it can be applied to an arbitrary path. However, it is formulated considering that the inner-loop controls the angular velocity of the vehicle, instead of its angular position. In order to test it, one has to come up with a system controlled in yaw rate, or try and adapt it to function with a heading controller. This approach makes use of a Frenet frame and a Lyapunov function that is used not only for convergence analysis purposes, but also at the control design stage. The basic reasoning is to use the Lyapunov's direct method and come up with the virtual control laws that make the Lyapunov function decrease, also proving stability. These outer-loop control laws and the Lyapunov function can consider additional user objectives to shape the convergence of the vehicle's trajectory to the desired path. Although this method seems to generalize well for different paths, only local convergence is proven, having a stringent initial condition constraint, which motivates the research of an improvement to Micaelli and Samson's work.

The approach proposed by L. Lapierre, D. Soetanto, and A. Pascoal in [8] aims to improve upon Micaelli and Samson's work [2], overcoming the stringent initial condition constraint. The issue is solved by controlling explicitly the rate of progression of a "virtual target" to be tracked along the path. This method follows the same reasoning as the Micaelli and Samson's, in terms of using a Lyapunov function both at the control design stage and to prove convergence. This path following approach is the one that is ultimately chosen to accomplish path following.

Following the path following problem, it is important to consider an adaptation to the L. Lapierre's method that allows the outer-loop to feed yaw angle references, instead of yaw rate ones, to the inner-loop. This is important to facilitate a path following approach that also considers the influence of the ocean currents. For this reason, a solution based on an ocean current observer that allows the system to compensate the perpendicular velocity component of the current to the path is researched. Ultimately the goal of the compensation mechanism is to consider the vehicle's total velocity vector instead of its main body axis, while steering the vehicle using the heading controller.

Having studied all the aforementioned work to achieve single vehicle motion control, cooperative motion control comes next. To assure a certain vehicle formation, during a cooperative mission, a coordination controller must be designed, which relies on a communication network scheme represented by a graph. Thus, one has to consider [12] to address the consensus problem and look into [13] to establish and model discrete communications, which introduce the concept of triggering functions.

Finally, after all of this fundamental research, it is possible to start tackling the problem of optical beam alignment and to prove that using a hybrid communication network is feasible while cooperative motion control is achieved.

1.3 Main Contributions

This work ultimately presents a mechanism to establish optical beam alignment between a pair of vehicles.

The proposed mechanism considers two phases: one where a rough alignment of the beams is reached, followed by another where a refinement is introduced. The reasoning behind the refinement phase is to apply a correction term that sweeps a neighborhood of the roughly determined alignment angles.

A situation with more than two vehicles is also considered. Naturally, optical communications are limited to a pair, as a result, one can apply the proposed solution to each pair interchangeably.

Lastly, simulations were created in order to illustrate this first approach on the problem the thesis addresses, using MATLAB & Simulink.

1.4 Thesis Outline

This thesis is organized as follows:

Chapter 2 introduces the MEDUSA-class of autonomous underwater vehicles and describes the dynamic model used for control systems design and simulation purposes.

Chapter 3 describes the inner-loop heading controller design.

Chapter 4 describes each path following approach and the adopted outer-loop controller.

Chapter 5 addresses multiple vehicle motion control with discrete communications.

Chapter 6 describes the proposed optical beam alignment mechanism with results.

Chapter 7 summarizes the obtained results and suggests topics for further research.

Chapter 2

Autonomous Underwater Robot Model

This chapter summarizes the mathematical model that describes the motion of a generic underwater vehicle in a three-dimensional space, with 6 **DOF**. This model is subsequently simplified to planar motion, yielding a model with 3 **DOF**. Further simplifications will be done taking into consideration the specifics of the MEDUSA class of autonomous underwater vehicles. This is an essential step towards designing and testing a controller for the system that we wish to control. The concepts and notation used in this chapter are based on previous work, namely [3].

2.1 Notation and Reference Frames

In order to arrive at a mathematical model that describes the motion of an **AUV**, it is important to first establish the appropriate reference frames and introduce the notation.

The pose (position and orientation) of an **AUV** in 3D space can be completely described using six variables. These variables, and the two relevant reference frames, are shown in figure 2.1.

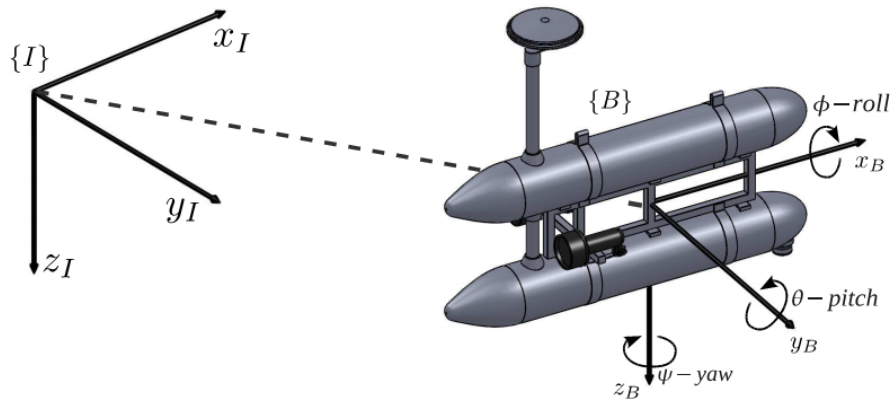


Figure 2.1: AUV reference frames and notation. (source: Ribeiro [4])

The body-fixed reference frame is represented by $\{B\}$ and the inertial reference frame is represented by $\{I\}$. The body frame is fixed to the vehicle's centre of mass, with the principal axes of inertia coin-

cident with the body axes x_B, y_B, z_B. With the inertial frame $\{I\}$ fixed somewhere in the world, it is possible to describe the pose of the vehicle relatively to this frame.

Now that the frames have been established, the definition of the notation used to express the pose of the vehicle, as well as its velocities, forces, and torques is as follows:

- position of $\{B\}$ expressed in $\{I\}$: $\boldsymbol{\eta}_1 = [x, y, z]^T$;
- orientation of $\{B\}$, in Euler-angles, with respect to $\{I\}$: $\boldsymbol{\eta}_2 = [\phi, \theta, \psi]^T$;
- linear velocity of $\{B\}$ with respect to $\{I\}$, expressed in $\{B\}$: $\boldsymbol{\nu}_1 = [u, v, w]^T$;
- angular velocity of $\{B\}$ with respect to $\{I\}$, expressed in $\{B\}$: $\boldsymbol{\nu}_2 = [p, q, r]^T$;
- forces acting on the vehicle, expressed in $\{B\}$: $\boldsymbol{\tau}_1 = [X, Y, Z]^T$;
- moments acting on the vehicle, expressed in $\{B\}$: $\boldsymbol{\tau}_2 = [K, M, N]^T$.

In compact form,

$$\boldsymbol{\eta} = [\boldsymbol{\eta}_1, \boldsymbol{\eta}_2] \qquad \boldsymbol{\nu} = [\boldsymbol{\nu}_1, \boldsymbol{\nu}_2] \qquad \boldsymbol{\tau} = [\boldsymbol{\tau}_1, \boldsymbol{\tau}_2]\,. \tag{2.1}$$

Degree of Freedom	Forces and Moments	Angular and Linear Velocities	Position and Euler Angles
Motion in the x-direction (surge)	X	u	x
Motion in the y-direction (sway)	Y	v	y
Motion in the z-direction (heave)	Z	w	z
Rotation about the x-axis (roll)	K	p	ϕ
Rotation about the y-axis (pitch)	N	q	θ
Rotation about the z-axis (yaw)	M	r	ψ

Table 2.1: SNAME notation for marine vehicles motion.

2.2 Kinematics

This section deals with the kinematics of the vehicle, which corresponds to the geometrical aspects of its motion. In particular, the main goal is to relate the velocity vectors expressed in the body-fixed reference frame with the corresponding ones in the inertial frame.

Linear Velocity Transformation

In the process of translating velocities from the body-fixed frame $\{B\}$ to the inertial frame $\{I\}$, the concept of rotation matrices plays a key role. Let us consider the rotation matrix $R_C^D \in \mathbb{R}^{3\times3}$, which represents the transformation of coordinates from $\{C\}$ to $\{D\}$. This being said, the desired rotation matrix for the kinematics problem should be defined as R_B^I, with ${R_B^I}^T$ representing its inverse rotation.

In order to accept that R^T represents the inverse of a general rotation matrix R, one should consider the following property:

Property - *A coordinate transformation matrix $R \in \mathbb{R}^{3\times3}$ satisfies*

$$RR^T = R^T R = I, \quad \det(R) = 1.$$

This implies that R is orthogonal. As a result, the inverse of the rotation matrix can be computed as $R^{-1} = R^T$.

Using the rotation matrix R_B^I, one can relate the body linear velocity measured in $\{B\}$ with the body position derivative, with respect to time, in $\{I\}$ through the relation

$$\dot{\boldsymbol{\eta}}_1 = R_B^I(\boldsymbol{\eta}_2)\boldsymbol{\nu}_1. \tag{2.2}$$

The rotation matrix is obtained by multiplying a series of principal rotation matrices. In this case the $z-y-x$ convention is used – that is, first the frame is rotated about the z axis (given ψ), then about the y axis (given θ) and lastly about the x axis (given ϕ). After the multiplication of these three consecutive rotations, R_B^I is defined as

$$R_B^I\left(\boldsymbol{\eta}_2\right) = \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{2.3}$$

where $s\cdot = \sin(\cdot)$ and $c\cdot = \cos(\cdot)$.

Angular Velocity Transformation

Analogously, the relation between the angular velocities expressed in $\{B\}$ and the derivative of the Euler-angles - that is, the time derivative of the orientation with respect to $\{I\}$ – is also given by a transformation matrix, in this case called the attitude matrix. This yields

$$\dot{\boldsymbol{\eta}}_2 = T_B^I\left(\boldsymbol{\eta}_2\right)\boldsymbol{\nu}_2, \tag{2.4}$$

where $T_B^I\left(\boldsymbol{\eta}_2\right)$, the attitude matrix, is defined as

$$T_B^I\left(\boldsymbol{\eta}_2\right) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi c\theta & c\phi/c\theta \end{bmatrix}, \tag{2.5}$$

with $t\cdot = \tan(\cdot)$.

To sum up, considering the transformation matrices in (2.3) and (2.5), the velocity transformations can be aggregated as

$$\begin{bmatrix} \dot{\boldsymbol{\eta}}_1 \\ \dot{\boldsymbol{\eta}}_2 \end{bmatrix} = \begin{bmatrix} R_B^I(\boldsymbol{\eta}_2) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & T_B^I(\boldsymbol{\eta}_2) \end{bmatrix} \begin{bmatrix} \boldsymbol{\nu}_1 \\ \boldsymbol{\nu}_2 \end{bmatrix} \iff \dot{\boldsymbol{\eta}} = J(\boldsymbol{\eta})\boldsymbol{\nu}. \tag{2.6}$$

2.3 Dynamics

In this section, forces and moments actuating in the **AUV** will now be considered. For this purpose, the Newton-Euler Formulation in the body-fixed frame will be used to derive the equations that translate the vehicle's dynamics, so that the inertia tensor is constant and forces generated by the thrusters and hydrodynamic effects can be expressed more easily.

The rigid-body dynamics can be expressed, in a compact form, as

$$M_{RB}\dot{\boldsymbol{\nu}} + C_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_{RB}, \tag{2.7}$$

where

- M_{RB} is the inertia matrix. This matrix is positive definite, symmetric and constant, i.e. $\dot{M}_{RB} = 0$ and $M_{RB} = M_{RB}^T > 0$;
- C_{RB} is the Coriolis and centripetal matrix, which can always be parametrized such that it is a skew-symmetrical matrix, i.e. $C_{RB}(\boldsymbol{\nu}) = -C_{RB}^T(\boldsymbol{\nu}) \forall \boldsymbol{\nu} \in \mathbb{R}^6$;
- $\boldsymbol{\tau}_{RB} = \left[\boldsymbol{\tau}_1^T, \boldsymbol{\tau}_2^T\right]^T$ is the generalized vector of external forces and moments.

If the term $\boldsymbol{\tau}_{RB}$ is interpreted as the sum of multiple torques and forces, which can be classified accordingly to its origins, it is possible to perform the following decomposition

$$\boldsymbol{\tau}_{RB} = \boldsymbol{\tau} + \boldsymbol{\tau}_A + \boldsymbol{\tau}_D + \boldsymbol{\tau}_R + \boldsymbol{\tau}_E, \tag{2.8}$$

where the term $\boldsymbol{\tau}$ is the vector of forces and moments produced by the thrusters and it is also the control input of the vehicle.

The $\boldsymbol{\tau}_A$ term represents moments and forces due to hydrodynamic added mass. It may itself be decomposed in the added inertia matrix, M_A, and the hydrodynamic added Coriolis and centripetal contribution, $C_A(\boldsymbol{\nu})$, resulting in

$$\boldsymbol{\tau}_A = -M_A\dot{\boldsymbol{\nu}} - C_A(\boldsymbol{\nu})\boldsymbol{\nu}. \tag{2.9}$$

In marine robotics the $\boldsymbol{\tau}_A$ term is not negligible, contrary to robotics on dry land, where the density of the air is much smaller than the density of the moving mechanical system, this however does not hold for marine robotics. Moreover, the concept of added mass can be misinterpreted as a finite amount of fluid that is added to the mass of the actual vehicle, which, in fact, doesn't happen, since the motion of the

vehicle causes the particles of all the fluid to displace. However, this displacement can be neglected as the distance from the vehicle increases. Added mass should be understood as pressure-induced forces and moments due to a forced harmonic motion of the body, which are proportional to the acceleration of the body.

The $\boldsymbol{\tau}_D$ term corresponds to the forces and moments generated by hydrodynamic damping, which is the energy carried away due to lift, drag and skin friction (among others). It can be expressed as

$$\boldsymbol{\tau}_D = -D(\boldsymbol{\nu})\boldsymbol{\nu}. \tag{2.10}$$

The $\boldsymbol{\tau}_R$ term corresponds to the contribution due to restoring forces, that is the weight and buoyancy mutual effect. It can also be expressed as

$$\boldsymbol{\tau}_R = -g(\boldsymbol{\nu}). \tag{2.11}$$

Finally, the $\boldsymbol{\tau}_E$ term sums the contributions from environmental forces and moments, such as wind, waves and ocean currents.

If the term $\boldsymbol{\tau}_{RB}$ in (2.7) is replaced by its decomposition defined in (2.8) and consecutively replacing in the latter the different forces and moments defined in equations (2.9), (2.10) and (2.11), the following representation of the dynamics for 6 DOFs is obtained:

$$\begin{aligned}
M_{RB}\dot{\boldsymbol{\nu}} + C_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} &= \boldsymbol{\tau} + \boldsymbol{\tau}_A + \boldsymbol{\tau}_D + \boldsymbol{\tau}_R + \boldsymbol{\tau}_E \\
M_{RB}\dot{\boldsymbol{\nu}} + C_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} &= \boldsymbol{\tau} - M_A\dot{\boldsymbol{\nu}} - C_A(\boldsymbol{\nu})\boldsymbol{\nu} - D(\boldsymbol{\nu})\boldsymbol{\nu} - g(\boldsymbol{\eta}) + \boldsymbol{\tau}_E \\
\left[M_{RB} + M_A\right]\dot{\boldsymbol{\nu}} + \left[C_{RB}(\boldsymbol{\nu}) + C_A(\boldsymbol{\nu})\right]\boldsymbol{\nu} + D(\boldsymbol{\nu})\boldsymbol{\nu} + g(\boldsymbol{\eta}) &= \boldsymbol{\tau} + \boldsymbol{\tau}_E \\
M\dot{\boldsymbol{\nu}} + C(\boldsymbol{\nu})\boldsymbol{\nu} + D(\boldsymbol{\nu})\boldsymbol{\nu} + g(\boldsymbol{\eta}) &= \boldsymbol{\tau} + \boldsymbol{\tau}_E,
\end{aligned} \tag{2.12}$$

where

- M is the inertia matrix (including added mass);
- $C(\boldsymbol{\nu})$ is the matrix of Coriolis and centripetal terms (including added mass);
- $D(\boldsymbol{\nu})$ is the damping matrix;
- $g(\boldsymbol{\eta})$ is the vector of gravitational forces and moments;
- $\boldsymbol{\tau}$ is the vector of control inputs;
- $\boldsymbol{\tau}_E$ is the environmental forces and moments.

2.4 Simplified Equations

Thus far, a general 6 DOF description of the motion of an **AUV** has been assumed in the kinematic and dynamic equations. However, for this work, it is expected that the vehicle is controlled only in the xOy

plane. As a result, and also due to the MEDUSA vehicle's geometry, roll and pitch can be neglected. In order to drive the vehicle to a desired operation depth, it is assumed access to an outer-loop heave motion controller.

Having said that, it is now possible to perform some simplifications to the kinematics and dynamics models, ending up with a 3 DOF model, where $\boldsymbol{\eta}$ becomes $\boldsymbol{\eta} = [x, y, \psi]^T$ and $\boldsymbol{\nu}$ becomes $\boldsymbol{\nu} = [u, v, r]^T$.

Starting with the rotation matrix R_B^I defined in (2.3), it is possible to replace now θ and ϕ by zero, resulting in

$$R(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}. \tag{2.13}$$

The angle transformation matrix defined in (2.5), taking into account the same approximations as for the rotation matrix, is simplified to $\dot{\psi} = r$. Together with (2.13), the simplified kinematic model can be fully described as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ r \end{bmatrix}. \tag{2.14}$$

Considering the first two lines of (2.14), that is

$$\begin{cases} \dot{x} = u\cos(\psi) - v\sin(\psi) \\ \dot{y} = u\sin(\psi) + v\cos(\psi) \end{cases}, \tag{2.15}$$

it is possible do define a new inertial position vector $\boldsymbol{p} = [x, y]^T$ and a new linear velocity vector, in the body-fixed coordinates, $\boldsymbol{v} = [u, v]^T$. In addition to this, a vector with the two inertial components of the current's velocity can be expressed as $\boldsymbol{V}_c = [V_{cx}, V_{cy}]$ – assuming it doesn't vary with time. Joining these three vectors together in an equation, the result is

$$\dot{\boldsymbol{p}} = R_B^I(\psi)\boldsymbol{v} + \boldsymbol{V}_c. \tag{2.16}$$

To address the simplification of the dynamics model defined previously in (2.12), the same new vectors for $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$, with 3 DOF, are considered. The inertia matrix M, the Coriolis and centripetal terms matrix $C(\boldsymbol{\nu})$, the damping matrix $D(\boldsymbol{\nu})$ and the restoring forces vector $g(\boldsymbol{\eta})$ can be expressed considering these new $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ vectors, for a 3 DOF simplification. In addition, it is relevant to recall that the vehicle's centre of gravity is coincident with the origin of the body-fixed frame. As a consequence,

$$\boldsymbol{\tau} = [\tau_u, \tau_v, \tau_r] \tag{2.17}$$

$$M = \begin{bmatrix} m_u & 0 & 0 \\ 0 & m_v & 0 \\ 0 & 0 & m_r \end{bmatrix} \tag{2.18}$$

$$C(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -m_v v \\ 0 & 0 & m_u u \\ -m_v v & m_u u & 0 \end{bmatrix} \tag{2.19}$$

$$D(\boldsymbol{\nu}) = \begin{bmatrix} d_u & 0 & 0 \\ 0 & d_v & 0 \\ 0 & 0 & d_r \end{bmatrix} \tag{2.20}$$

$$g(\boldsymbol{\eta}) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \tag{2.21}$$

where

$$\begin{aligned} m_u &= m - X_{\dot{u}} & d_u &= -X_u - X_{u|u|}|u| \\ m_v &= m - Y_{\dot{v}} & d_v &= -Y_v - Y_{uv|}|v| \\ m_r &= I_z - N_{\dot{r}} & d_r &= -N_r - N_{r|r|}|r| \\ m_{uv} &= m_u - m_v \end{aligned} \tag{2.22}$$

and $g(\boldsymbol{\eta})$ is zero, because, in this case, the weight and the buoyancy forces are equal and the vehicle's centre of gravity is coincident with the centre of buoyancy. The simplified dynamics model can now be expressed as

$$\begin{aligned} &m_u \dot{u} - m_v v r + d_u u = \tau_u \\ &m_v \dot{v} + m_u u r + d_v v = \tau_v \\ &m_r \dot{r} - m_{uv} u v + d_r r = \tau_r \end{aligned} \quad . \tag{2.23}$$

2.5 MEDUSA-class Vehicles

The MEDUSA-class vehicles are autonomous submerged or semi-submerged robotic vehicles developed at the Laboratory of Robotics and Systems in Engineering and Science (LARSyS)/ISR of Instituto Superior Técnico. The available fleet at IST is comprised of three similar vehicles, two of them with diving capabilities. These vehicles can have different hardware mounted on them, depending on the mission. This allows for distinct roles for each vehicle, where they all complement each other's capabilities in a certain mission.

The MEDUSA-class vehicles have two thrusters arranged symmetrically relatively to the xOz plane, with the direction of x. These thrusters can operate in common mode, moving the vehicle forward or backwards, or in differential mode, allowing the vehicle to turn around its z-axis. The diver version of the

vehicles has two additional thrusters aligned with the z-axis, allowing for heave control.

Having described the MEDUSA-class vehicles' thrusters, it is possible to define

$$\begin{aligned} \tau_u &= F_s + F_p \\ \tau_r &= l\,(F_s - F_p) \end{aligned}, \tag{2.24}$$

where F_s and F_p are the force produced by the starboard thruster and the force produced by the portside thruster, respectively. Moreover, $l = 0.15$ m, for a MEDUSA vehicle. The vehicles also have neutral buoyancy, meaning that $g(\boldsymbol{\eta}) = \mathbf{0}_{3\times 1}$. On top of this, there are no actuators to control the sway, this means that $\tau_v = 0$.

This being said, it is now important to know the parameters that adapt the dynamic equations to the MEDUSA-class vehicles. Some of them can be measured directly, however the rest of them have to be determined through experiments performed in water tanks. These model parameters were originally obtained by [4] and then tuned over the years. The most recent model parameters being used are defined in the following table:

$m = 17.0$ kg	$I_z = 1$ kg·m^2	
$X_{\dot{u}} = -20$ kg	$Y_{\dot{v}} = -30$ kg	$N_{\dot{r}} = -8.69$ kg·m^2
$X_u = -0.2$ kg/s	$Y_v = -50$ kg/s	$N_r = -4.14$ kg·m/s
$X_{\lvert u\rvert u} = -25$ kg/m	$Y_{\lvert v\rvert v} = -0.01$ kg/m	$N_{\lvert r\rvert r} = -6.23$ kg·m

Table 2.2: Model parameters for the MEDUSA-class vehicle.

All of this information can now be considered in the simplified dynamic model described in (2.23), after using the model parameters in (2.22). This sums up the adaptation of the previously derived equations to the MEDUSA-class vehicle.

Chapter 3

Heading Control

This chapter addresses the design of a heading controller for the vehicle. For this purpose, a state-space representation of the vehicle's model will be adopted, based on the simplified equations obtained in the last chapter.

The system to be controlled is non-linear, as shown in the previous chapter. For this reason, a linearisation about an equilibrium point can be performed, in order to apply the Linear Quadratic Regulator (LQR) method to control the vehicle's heading using state feedback.

3.1 State-Space Model

Assuming that a system has n states, p inputs and q outputs, a state-space representation is given by

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}.$$

In this case, the state vector is given by $x = [v, r, \psi]^T$ ($n = 3$) and the input is $u = \tau_r$ ($p = 1$). Since the yaw angle is to be controlled, the output is $y = \psi$ ($q = 1$), assuming the notation simplification: $x(t) = x$, $u(t) = u$ and $y(t) = y$. Moreover, $A \in \mathbb{R}^{n \times n}$ is the state matrix, $B \in \mathbb{R}^{n \times p}$ is the input matrix, $C \in \mathbb{R}^{q \times n}$ is the output matrix and finally, in this case, the system doesn't have a direct feedthrough, so $D \in \mathbb{R}^{q \times p}$ is a zero matrix.

In order to obtain the state-space model, for simplicity, it is assumed a constant speed, $u^* = 0.5$ m/s, in the surge direction. For this reason, observing the dynamic model (2.23), the first equation can be ignored, leaving only the last two to be considered in the system model.

Moreover, to complete the system model, considering the state vector $x = [v, r, \psi]^T$, one can resort to the kinematics in (2.14) and take the equation that relates the yaw with the yaw rate, that is, the last equation defined as $\dot{\psi} = r$. By isolating the state variables, it is possible to obtain the following model:

$$
\begin{aligned}
\dot{v} &= -\frac{m_u}{m_v} ur - \frac{d_v}{m_v} v \\
\dot{r} &= -\frac{d_r}{m_r} r + \frac{m_{uv}}{m_r} uv + \frac{\tau_r}{m_r}. \\
\dot{\psi} &= r
\end{aligned}
\tag{3.1}
$$

To obtain the linearised state-space model, the linearisation is to be performed about the equilibrium point $v = 0$, $r = 0$, $\psi = 0$ and $\tau_r = 0$. This is a reasonable approximation, because the side-slip and the rotation speeds are small with these vehicles. In view of this, the linearisation is defined as

$$
A = \frac{df}{dx}\Big|_{\substack{x=\bar{x} \\ u=\bar{u}}} \qquad B = \frac{df}{du}\Big|_{\substack{x=\bar{x} \\ u=\bar{u}}}
$$

$$
A = \begin{bmatrix} \frac{\partial \dot{v}}{\partial v} & \frac{\partial \dot{v}}{\partial r} & \frac{\partial \dot{v}}{\partial \psi} \\ \frac{\partial \dot{r}}{\partial v} & \frac{\partial \dot{r}}{\partial r} & \frac{\partial \dot{r}}{\partial \psi} \\ \frac{\partial \dot{\psi}}{\partial v} & \frac{\partial \dot{\psi}}{\partial r} & \frac{\partial \dot{\psi}}{\partial \psi} \end{bmatrix}_{\substack{x=\bar{x} \\ u=\bar{u}}} \qquad B = \begin{bmatrix} \frac{\partial \dot{v}}{\partial \tau_r} \\ \frac{\partial \dot{r}}{\partial \tau_r} \\ \frac{\partial \dot{\psi}}{\partial \tau_r} \end{bmatrix}_{\substack{x=\bar{x} \\ u=\bar{u}}}.
$$

Recalling that the output variable is ψ, the resultant state-space model is

$$
\begin{aligned}
\dot{x}(t) &= \begin{bmatrix} \frac{Y_v}{m_v} & -\frac{m_u}{m_v} u^* & 0 \\ \frac{m_{uv}}{m_r} u^* & \frac{N_r}{m_r} & 0 \\ 0 & 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{1}{m_r} \\ 0 \end{bmatrix} u(t) \\
y(t) &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} x(t)
\end{aligned}
\tag{3.2}
$$

3.2 Linear Quadratic Regulator

In order to control the system, a Linear Quadratic Regulator (LQR) approach will be followed. This is applied knowing beforehand the state-space model that was already obtained in the previous section, defined in (3.2). The control objective is to find, among the class of admissible control laws, the one which minimizes the cost criterion

$$
J = \int_{t=0}^{\infty} \Big[x(t)^T Q x(t) + u(t)^T R u(t) \Big] dt, \tag{3.3}
$$

where $Q \in \mathbb{R}^{n \times n}$ penalises the state error and $R \in \mathbb{R}^{p \times p}$ penalises the control effort. To ensure that the control problem is well-posed, one has to verify that the pair (A, B) is stabilizable and the pair $(A, \sqrt{Q})$ is detectable. If $Q = C^T C$, for some matrix C, it is shown in [5] that $(A, \sqrt{Q})$ is detectable if and only if (A, C) is detectable, under the assumption that Q is a positive semi-definite matrix, thus having only one positive semi-definite square root [6, Theorem 7.2.6]. Considering $Q = C^T C$, the cost criterion penalises the energy of the state for a virtual output, given by $y = Cx$, as follows:

$$
J = \int_{t=0}^{\infty} \Big[y(t)^T y(t) + u(t)^T R u(t) \Big] dt = \int_{t=0}^{\infty} \Big[x(t)^T C^T C x(t) + u(t)^T R u(t) \Big] dt. \tag{3.4}
$$

If the detectability and stabilizability assumptions are verified, there exists a unique stabilizable solution to the LQR problem, given by the state feedback control:

$$u(t) = -Kx(t), \tag{3.5}$$

where the gain vector $K \in \mathbb{R}^{n\times 1}$ satisfies

$$K = R^{-1}B^T P, \tag{3.6}$$

and $P \geq 0$ is a unique solution to the continuous time Algebraic Ricatti Equation:

$$A^T P + PA - PBR^{-1}B^T P + Q = 0, \tag{3.7}$$

which is used to solve the LQR optimal control problem. The solution to the Algebraic Ricatti Equation can be $P > 0$ if, in fact, the pairs (A, B) and $(A, \sqrt{Q})$ are proven controllable and observable, respectively (stronger concepts of detectable and stabilizable). Often it is too much to assume that one can have full observability and controllability, thus having to settle for detectability and stabilizability. Thus, if one has a stabilizable and detectable system, there could be dynamics one is not aware of and cannot influence, but they are at least stable.

The challenge of designing a LQR controller lies in finding appropriate weighting matrices Q and R. This implies an iterative process of weight tuning. A first good iteration for the Q matrix is given by Bryson's rule, where the Q weights are initially set with the inverse of the maximum accepted value of the square of each state variable, denoted x^2_{max}, as shown:

$$Q = \begin{bmatrix} \frac{1}{(v_{max})^2} & 0 & 0 \\ 0 & \frac{1}{(r_{max})^2} & 0 \\ 0 & 0 & \frac{1}{(\psi_{max})^2} \end{bmatrix}. \tag{3.8}$$

The same rule can be applied to the first iteration of the weighting term R, with $R = (\tau_{rmax})^{-2}$.

With the first iterations for the weighting matrices Q and R already defined, as mentioned before, it is possible to obtain the gain vector K by solving the Algebraic Ricatti Equation – in this case, this is done using MATLAB®.

In order to obtain the desired behaviour of driving the states to zero: as fast and smooth as possible, without oscillations and with minimal or non-existent overshooting, it is important to know what happens if one increases or decreases the penalisation weights. For instance, if R is a large value, this means that the system must be stabilised with less energy ("expensive control" strategy). Consecutively, if R is a small value, the control signal will be less penalised, allowing for higher peak control signal values – which may lead to aggressive thruster commands ("cheap control" strategy). Regarding the penalisation of the state error through Q, if these weight values are small, the evolution of the corresponding states is such that they may take longer to be driven to zero; however if the evolution of a certain state shows unwanted overshooting, its corresponding weight must be increased. A good response results from a

good trade-off between the penalisation weights that impact the output as just mentioned.

With this formulation it is assumed that the desired state is zero. As a result, the LQR controller will be driving the state to zero, including state variable to control, ψ. For this reason, one can replace the output state ψ by the error defined as the difference between the reference and the output: $e = \psi_{ref} - \psi$, which will, instead of ψ, be driven to zero as desired, while ψ converges to the reference value.

In figure 3.1 a block diagram is represented with the LQR state feedback controller for the 3 DOF system obtained previously and the appropriate adjustment to consider a reference signal.

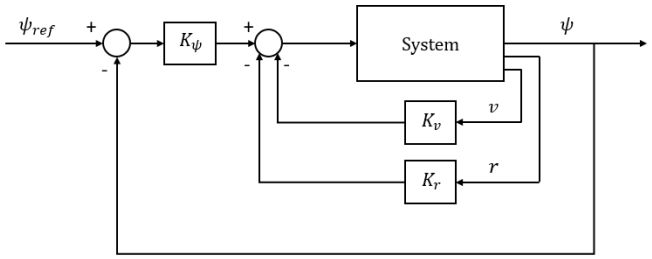


Figure 3.1: LQR state feedback controller with reference.

Using the following Q and R weighting matrices, after some iterations, the results are as shown in figure 3.2.

$$Q = \begin{bmatrix} 4.00 & 0 & 0 \\ 0 & 1.56 & 0 \\ 0 & 0 & 11.11 \end{bmatrix} \quad R = 0.01 \tag{3.9}$$

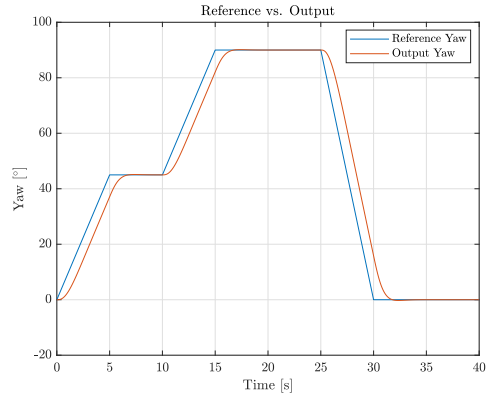
Resulting in,

$$K = \begin{bmatrix} -5.09 & 25.15 & 33.33 \end{bmatrix} \tag{3.10}$$

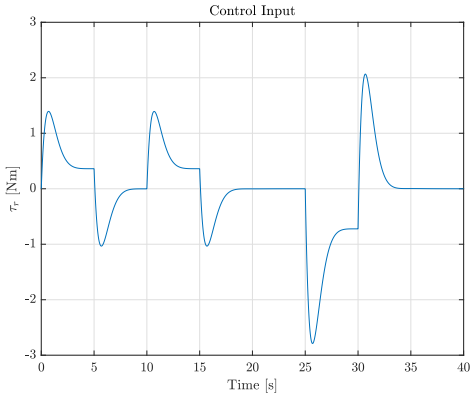
Using Kalman's Criterion, it is possible to prove that the pairs (A, B) and (A, C) are controllable and observable, respectively, with $C = \sqrt{Q}$. In fact,

$$\begin{aligned} \operatorname{rank}\begin{bmatrix} B & AB & A^2B \end{bmatrix} &= \operatorname{rank}\begin{bmatrix} 0 & -0.0406 & 0.0606 \\ 0.1032 & -0.0441 & 0.0398 \\ 0 & 0.1032 & -0.0441 \end{bmatrix} = n = 3 \\ \operatorname{rank}\begin{bmatrix} C & CA & CA^2 \end{bmatrix}^T &= \operatorname{rank}\begin{bmatrix} 2.0000 & 0 & 0 \\ 0 & 1.2500 & 0 \\ 0 & 0 & 3.3333 \\ -2.1277 & -0.7872 & 0 \\ -0.6450 & -0.5341 & 0 \\ 0 & 3.3333 & 0 \\ 2.6697 & 1.1738 & 0 \\ 0.9617 & 0.4821 & 0 \\ -1.7200 & -1.4241 & 0 \end{bmatrix} = n = 3 \end{aligned} \tag{3.11}$$

As a result, the system is controllable and observable and there exists a unique and stabilizable solution to the LQR problem, with $P > 0$.



(a) Comparison between the output and the reference.



(b) Commanded input.

Figure 3.2: Simulation of the state feedback heading controller.

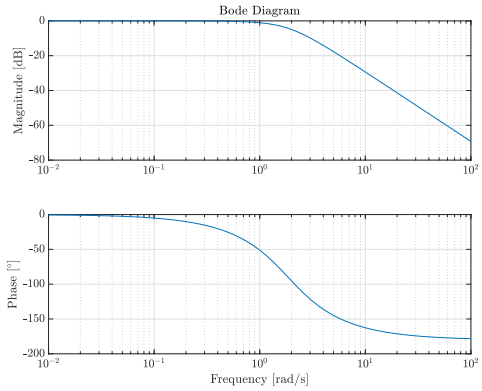


Figure 3.3: Bode diagram of the closed-loop system - from the yaw reference to the yaw output.

The obtained output is stable, with the following closed-loop eigenvalues:

$$\lambda_1 = -1.061, \quad \lambda_{2,3} = -1.513 \pm j1.077, \tag{3.12}$$

and the response behaves as desired, with unit steady-state gain.

State-Space Model Simplification

In reality, measuring the sway linear velocity, v, is expensive. As a result, one could either use state estimation methods to have a measure of v, or try to simplify the model by neglecting it. The latter approach is more appealing for this case, where the v is usually a small value and it wouldn't be possible to benefit much from a state estimation approach.

As a result, since the control is done considering the state variable ψ, it is possible to simplify the state-space model by taking out the sway linear velocity, v, from the state vector. This is equivalent to making it zero, $v = 0$, and removing the corresponding row and column from the state matrix A, as well

as from the input and output vectors B and C respectively. Later, results that support this simplification will be shown.

As a result, the simplified state-space model satisfies

$$\begin{aligned} \begin{bmatrix} \dot{r} \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} \frac{N_r}{m_r} & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} r \\ \psi \end{bmatrix} + \begin{bmatrix} \frac{1}{m_r} \\ 0 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} r \\ \psi \end{bmatrix} \end{aligned} . \tag{3.13}$$

Therefore, the v feedback will no longer be represented in the new controlled system block diagram, as it is in figure 3.1.

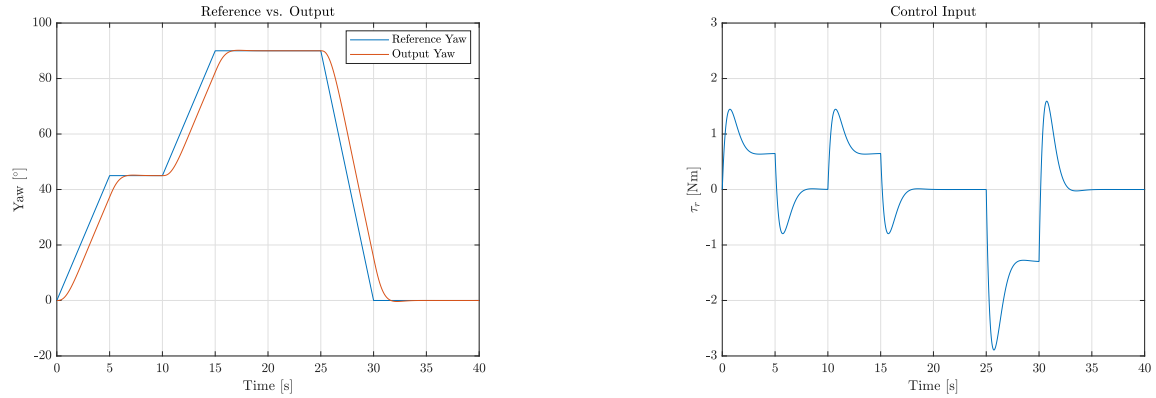
The following results were obtained after the model simplification:

$$Q = \begin{bmatrix} 1.56 & 0 \\ 0 & 11.11 \end{bmatrix} \quad R = 0.01 \tag{3.14}$$

Resulting in,

$$K = \begin{bmatrix} 24.49 & 33.33 \end{bmatrix} \tag{3.15}$$

and:



(a) Comparison between the output and the reference.

(b) Commanded input.

Figure 3.4: Simulation of the state feedback heading controller after state-space model simplification.

The obtained output is stable, with the following closed-loop eigenvalues:

$$\lambda_{1,2} = -1.477 \pm j1.122. \tag{3.16}$$

Since the output is almost identical to the previously obtained in figure 3.2 and the closed-loop eigenvalues are similar to the previously obtained in (3.12), one can accept the state-space simplification discussed in this section.

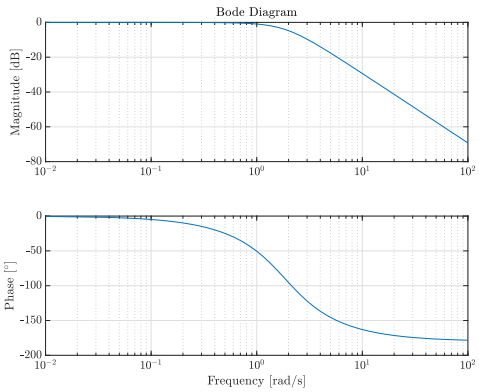


Figure 3.5: Bode diagram of the closed-loop system after state-space model simplification - from the yaw reference to the yaw output.

3.3 Integral Effect

To complete the design of the heading controller, an integral effect can be added in order to reject constant disturbances (constant external inputs affecting the command of the thrusters). To achieve this, one has to consider an additional integrator placed before the disturbance, thus, integrating the error with respect to time as in

$$\epsilon = \int_0^t \psi_{ref} - \psi \quad dt. \tag{3.17}$$

Adding this effect implies the addition of ϵ to the state vector, as a state variable with a corresponding gain in the K vector. The previous state-space model is, therefore, adapted taking into account the dynamics of ϵ, given by $\dot{\epsilon} = \psi_{ref} - \psi$, resulting in the new state matrix:

$$A = \begin{bmatrix} \frac{N_r}{m_r} & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \tag{3.18}$$

With the updated state-space model, one can go back to the LQR controller and recalculate the K vector, in order to obtain the additional K_ϵ gain. In figure 3.6 is represented the additional integrator in a block diagram, with the new design allowing for disturbance rejection at the controller output.

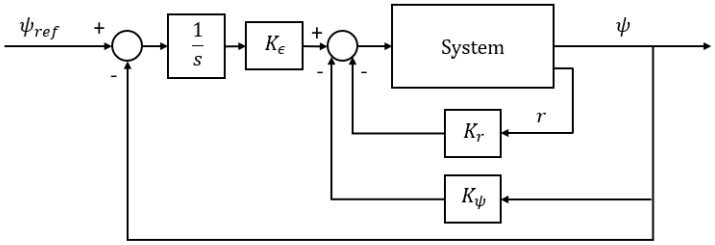


Figure 3.6: Block diagram of the controlled system with integral effect.

However, by using this configuration (integrating the error), the system is at risk of experiencing

integrator wind-up. This effect happens, assuming that at some point the actuators saturate – the input saturates and the integral of the error keeps increasing. In turn, when the error decreases, the large integral value prevents the controller from resuming "normal operations" quickly, so the response is delayed. To solve this problem, one can adopt an anti-windup design that stops integrating the error once the input saturates.

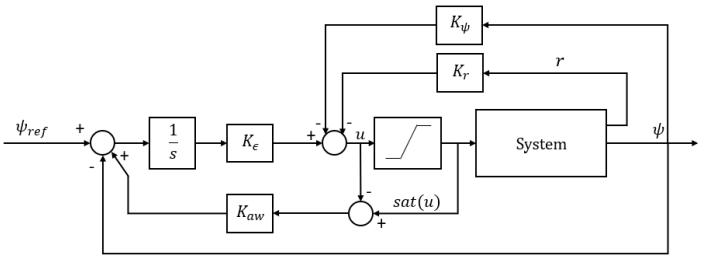


Figure 3.7: Anti-windup design with integral effect.

The anti-windup scheme represented in figure 3.7 prevents divergence of the integral error when the control cannot keep up with the reference, maintaining the integral errors "small" (avoids error buildup). The thought process to reach this design starts with the comparison of the actual input with the commanded input: if they are the same there's no saturation, otherwise the integral error must be reduced by a constant multiplied by the difference between the saturated and commanded inputs, as in

$$u_{aw} = \int K_\epsilon \left[e + K_{aw}(\mathrm{sat}(u) - u) \right] dt, \tag{3.19}$$

noticing that when $\mathrm{sat}(u) - u = 0$, there's no anti-windup effect, because the input is not saturated, falling back to the situation represented in figure 3.6.

The issue now is that, for the anti-windup scheme to be implemented and produce a quick action, the integrator must be right next to the plant, which is not the case in figure 3.7. More importantly, placing the integrator next to the plant results in a Bumpless Transfer method, allowing the controller to be transitioned from manual mode to automatic mode without disrupting the process, since the integrator placed at the control signal output is capable of smoothing out the transition bump.

In order to accomplish this, one can look into the theory presented in the paper Kaminer et al. [7]. This paper proves that it is possible to place the integrator after the state feedback through the gain vector K, by first differentiating the state, while preserving the closed-loop eigenvalues as well as the input-output properties of the original linear closed-loop system locally. Therefore, the new configuration represented in figure 3.8 is proven possible and results in the same controlled system output, while making the anti-windup action effective and allowing bumpless transfer.

The method represented in figure 3.8 is possible because the state derivatives have physical meaning, otherwise the required differentiation couldn't be done in practice. Instead, the differentiation operator would have to be replaced by a causal system with transfer function $s/(\alpha s + 1)$, with $\alpha > 0$, while having the linearisation property recovered asymptotically as $\alpha \to 0$ - as proven in the paper by Kaminer.

In figure 3.9 are represented simulation results of the implementation of the integral effect designs –

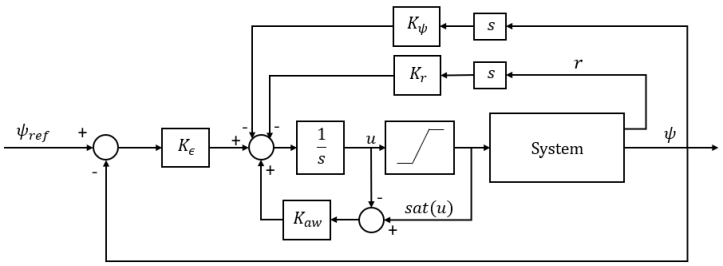
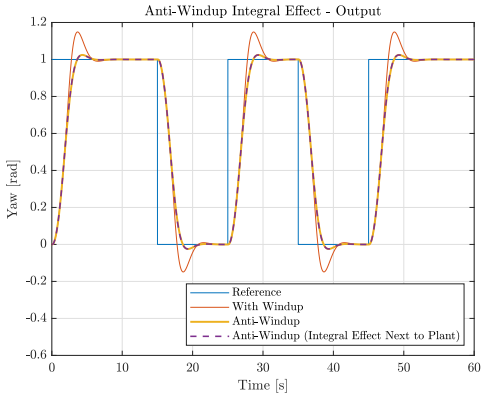
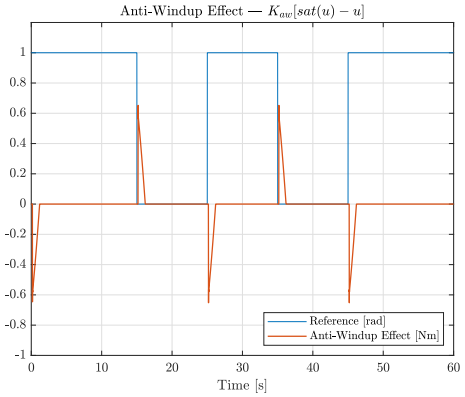


Figure 3.8: Anti-windup design with integral effect next to the plant.

with and without anti-windup. To do this, a discontinuous reference signal (square wave) was used, in order to saturate the actuators at its discontinuities. The K_{aw} gain was tuned in order to produce the best response: less delayed by the integral effect and closer to the output obtained with a reference that doesn't saturate the actuators.



(a) Controller output with and without anti-windup.



(b) Time representation of the anti-windup effect.

Figure 3.9: Integral effect simulation results with and without anti-windup.

These results prove that the anti-windup design with integral effect next to the plant, and corresponding state derivatives, produces the same output as the design represented in figure 3.7, as expected. The time representation of the anti-windup effect, figure 3.9 (b), shows that the anti-windup effect is null when the actuators are not being saturated, as desired.

Chapter 4

Path Following

In the previous chapter, an inner-loop was designed to control the vehicle's heading. However, this is not sufficient to drive a vehicle along a desired path. To do this, an outer-loop must be designed, one that can generate yaw angle references knowing the path and making use of the vehicle kinematics. The goal is to make the vehicle converge to and follow the path at a desired speed, that can be path dependent.

This suggests that an inner-outer loop decoupling strategy be implemented in order to control the motion of a single vehicle. Since the kinematics have already been addressed, the only outer-loop missing component is the path following controller, which will feed reference yaw angles to the inner-loop. To do this, a path parametrised in space must be known, as this is the starting point of any path following approach.

That being said, this chapter will address the completion of the design of the outer-loop, making use of the kinematics and feeding a yaw reference to the inner-loop. There are many different ways to obtain reference yaw angles from a path, in this chapter, three different path following approaches will be discussed.

4.1 Line of Sight

The Line of Sight (LOS) algorithm is the simplest path following algorithm to implement. The main goal of this algorithm is to compute the steering angle taking into consideration a lookahead distance, starting from the path's point that is closest to the vehicle.

In this section, the LOS algorithm will be first derived for a straight line path following scenario and then it'll be extended for a circumferential path.

Line of Sight for Straight Lines

In figure 4.1 is represented the configuration for the path following of a straight line, considering that the straight line exists between two waypoints p_k and p_{k+1} and that it can be interpreted as one of the segments that belongs to a longer path composed of straight line segments.

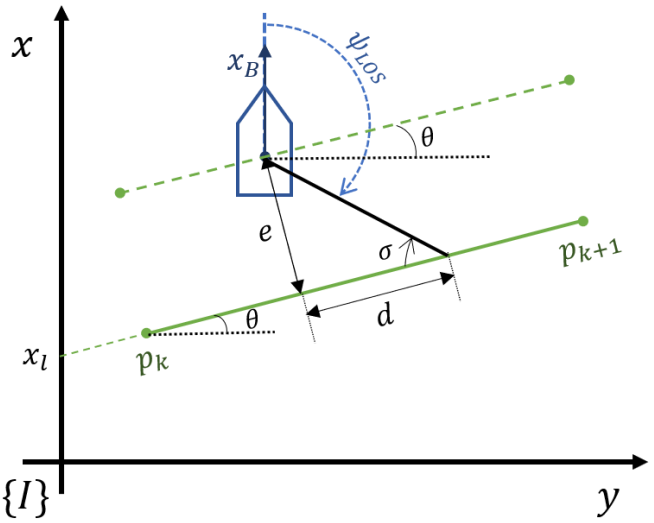


Figure 4.1: Line of Sight configuration for straight lines.

As can be observed in figure 4.1, the lookahead distance is represented by d and the distance of the vehicle to the closest point belonging to the path is represented by e (cross-track error). A waypoint is expressed as $\boldsymbol{p_{k+1}} = [x_k, y_k]^T$ and the position of the centre of gravity of the vehicle is expressed as $\boldsymbol{p} = [x, y]^T$, obtained from the kinematic equations.

For a general straight line, with any inclination, the desired reference yaw angle, given by the LOS algorithm, can be computed as

$$\psi_{LOS} = \frac{\pi}{2} - \theta + \sigma, \tag{4.1}$$

where $\pi/2$ comes from the fact that the yaw is measured relatively to the north direction, θ is used to account for the line's inclination (for the simpler case of an horizontal line this term is zero) and σ accounts for the definition of the LOS algorithm.

The σ angle can be computed as

$$\sigma = \arctan\left(\frac{e}{d}\right). \tag{4.2}$$

In addition, the straight line inclination, θ, recalling that x points north, is obtained from

$$\theta = \arctan\left(\frac{x_{k+1} - x_k}{y_{k+1} - y_k}\right). \tag{4.3}$$

In order to compute the angle σ, the only unknown is the cross-track error e, because the lookahead distance d is a tuning parameter of the LOS algorithm – so it is initially set with a certain value. The cross-track error term can be computed, for any straight line, with the help of a rotation matrix defined as

$$\bar{R}_I^F(\theta) = \begin{bmatrix} \sin(\theta) & \cos(\theta) \\ \cos(\theta) & -\sin(\theta) \end{bmatrix}. \tag{4.4}$$

This rotation matrix, $\bar{R}_I^F(\theta)$, is obtained from switching the columns of $R_I^F(\theta)$, as in

$$R_I^F(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \quad \Rightarrow \quad \bar{R}_I^F(\theta) = \begin{bmatrix} \sin(\theta) & \cos(\theta) \\ \cos(\theta) & -\sin(\theta) \end{bmatrix}. \tag{4.5}$$

The columns of $R_I^F(\theta)$ are switched to consider the adopted convention for the axes positioning. This can be observed in figure 4.1, where the x-axis is oriented north, instead of the y-axis, as it is usually done in a traditional convention.

In view of the above, if one transforms the position of the vehicle to the frame $\{F\}$ – rotated in accordance to the inclination of the straight line and with the x-axis coincident with the straight line –, one can obtain the cross-track error, considering a vertical translation due to the intercept x_l, by computing

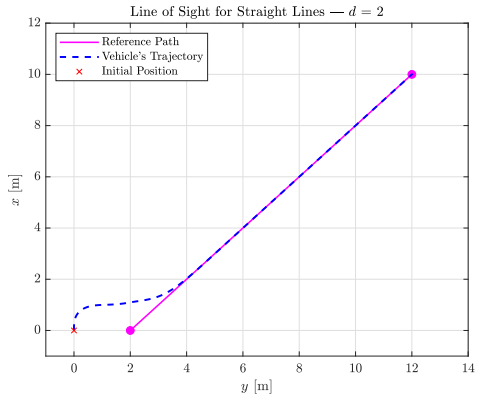
$$\begin{bmatrix} d_x \\ e \end{bmatrix} = \bar{R}_I^F(\theta) \begin{bmatrix} x - x_l \\ y \end{bmatrix} \qquad . \tag{4.6}$$
$$e = (x - x_l)\cos(\theta) - y\sin(\theta)$$

Now that the cross-track error can be computed, it is possible to finally compute ψ_{LOS}:

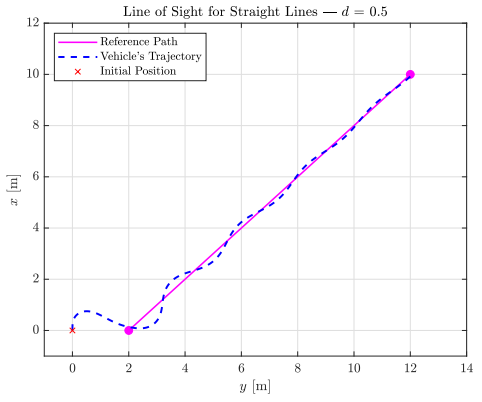
$$\psi_{LOS} = \frac{\pi}{2} - \theta + \arctan\left[\frac{(x - x_l)\cos(\theta) - y\sin(\theta)}{d}\right]. \tag{4.7}$$

The approach of computing e through a rotation matrix eases the adaption of the LOS algorithm to the next scenario, where the path is a circumference, since the reasoning will be similar.

The implementation of this algorithm yields the following results:



(a) Simulation with LOS lookahead distance $d = 2$.



(b) Simulation with LOS lookahead distance $d = 0.5$.

Figure 4.2: Simulation of the LOS algorithm for straight lines with different lookahead distances.

As can be deduced from the results, a smaller lookahead distance yields an oscillatory response of the cross-track error convergence. The lookahead distance value must be chosen as to ensure the fastest, non-oscillatory, convergence of the cross-track error.

Line of Sight for Circumferences

In figure 4.3 is represented the configuration for the path following of a circumference of radius R and centred at $\boldsymbol{C} = [C_x, C_y]^T$, along with the representation of a new frame $\{F\}$, similar to the one used in the LOS approach for straight lines.

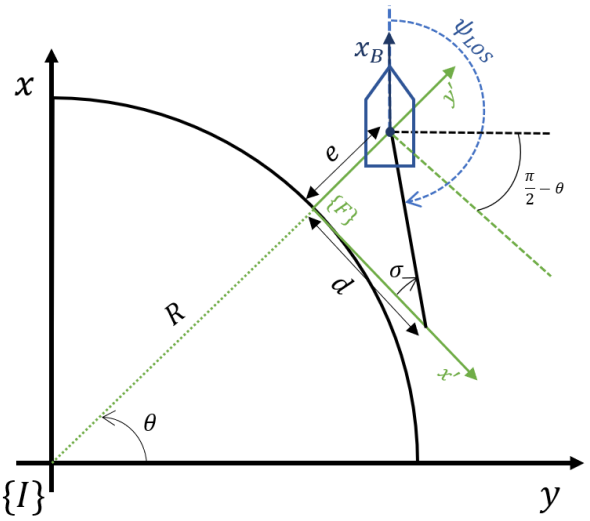


Figure 4.3: Line of Sight configuration for circumferential path following.

The notation used in this case is the same as the one used in the straight line configuration, with the exception of θ, which now represents the inclination of the line that crosses the origin and the centre of gravity of the vehicle, given by

$$\theta = \arctan\left(\frac{x - C_x}{y - C_y}\right). \tag{4.8}$$

It is important to recall the adopted frame convention, with the x-axis oriented north. On top of that, the C_x and C_y are used to compensate for the translation of the configuration due to where the circumference is centred at. Moreover, the four-quadrant version of the inverse tangent function should be assumed.

Under this configuration, the desired yaw reference is given by

$$\begin{aligned} \psi_{LOS} &= \frac{\pi}{2} + \left(\frac{\pi}{2} - \theta\right) + \sigma \\ &= \pi - \theta + \sigma \end{aligned}, \tag{4.9}$$

with $\sigma = \arctan\left(\frac{e}{d}\right)$, as previously computed in (4.2).

Similarly to the straight line LOS algorithm, d is a known tuning parameter and e is the only unknown that must be computed, in order to obtain σ. The cross-track error will also be computed with the help of a rotation matrix, just like in the straight line scenario, with $\bar{R}_I^F(\phi)$ defined as in (4.4). The only differences now are that $\bar{R}_I^F(\phi)$ changes as the vehicle moves (there's a new ϕ every time the vehicle moves) and, in order to transform coordinates in $\{I\}$ to coordinates in $\{F\}$, a new translation vector is considered, one that places the frame $\{F\}$ on top of the circumference:

$$T = \begin{bmatrix} R\sin(\theta) \\ R\cos(\theta) \end{bmatrix}. \tag{4.10}$$

The rotation matrix is defined as a function of ϕ, given by

$$\phi = -\Big(\frac{\pi}{2} - \theta\Big) = \theta - \frac{\pi}{2}, \tag{4.11}$$

instead of as a function of θ, because ϕ represents the clockwise rotation relatively to the x-axis, as desired in this case. If it was θ instead, the transformations would be done counter-clockwise and, thus, they would follow the symmetric movement of the vehicle.

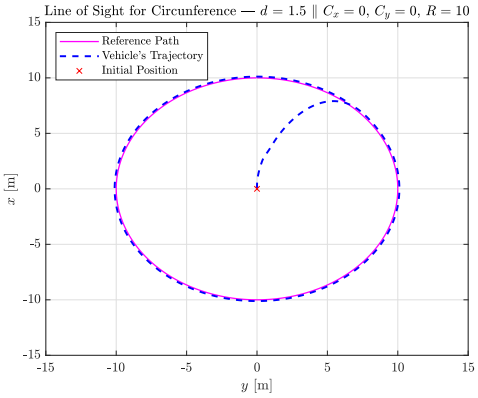
By following the above procedure, the problem is transformed into one expressed in $\{F\}$, which follows the vehicle's movement and moves on top of the circumference. The lookahead distance coincides with the x'-axis of the $\{F\}$ frame and describes the tangent to the circumference with origin in the orthogonal projected position of the vehicle on the circumference. Moreover, the lookahead distance is defined on top of a tangent straight line, instead of being defined on an curving arc of the path, and this is an approximation used in this LOS algorithm for a circumferential path. Under these conditions, the cross-track error is obtained from

$$\begin{bmatrix} d_x \\ e \end{bmatrix} = \bar{R}_I^F(\phi)\left(\begin{bmatrix} x - C_x \\ y - C_y \end{bmatrix} - \begin{bmatrix} R\sin(\theta) \\ R\cos(\theta) \end{bmatrix}\right) \\ e = \Big[x - C_x - R\sin(\theta)\Big]\cos(\phi) - \Big[y - Cy - R\cos(\theta)\Big]\sin(\phi) \tag{4.12}$$

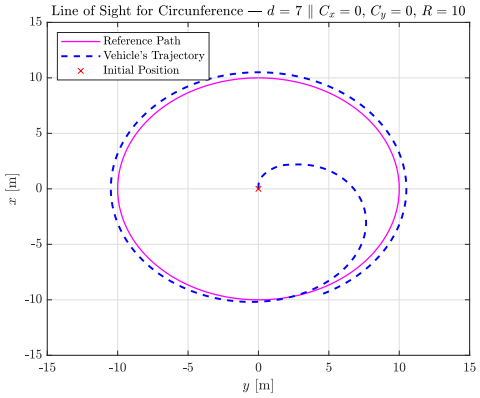
Recalling (4.9), the desired ψ_{LOS} can be obtained from

$$\psi_{LOS} = \pi - \theta + \arctan\left(\frac{\Big[x - C_x - R\sin(\theta)\Big]\cos(\phi) - \Big[y - Cy - R\cos(\theta)\Big]\sin(\phi)}{d}\right). \tag{4.13}$$

This wraps up the LOS approach for path following. An important property of this strategy is illustrated in figure 4.4: notice that the use of a larger lookahead distance results in a larger offset between the reference path and the vehicle's trajectory. If a smaller lookahead distance, $d < 1.5$, were used, the oscillatory response seen in figure 4.2 would be obtained again.



(a) Simulation with LOS lookahead distance $d = 1.5$.



(b) Simulation with LOS lookahead distance $d = 7$.

Figure 4.4: Simulation of the LOS algorithm for circumferences with different lookahead distances.

In figure 4.5 is represented the result of the LOS implementation for a reference path composed of straight line segments and half circumferences, where there's still some offset when the vehicle follows the curved segments.

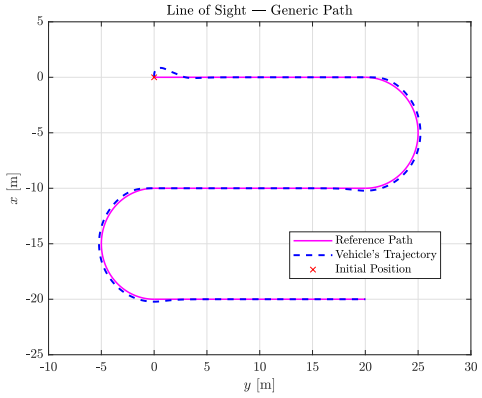


Figure 4.5: LOS results for a generic path.

4.2 P. Maurya's Guidance Method

This path following method was proposed by P. Maurya, A. Pedro Aguiar and A. Pascoal in [1]. The method considers a strategy of inner-outer loop control, with the guidance algorithm designed as a controller that drives the cross-track error to zero, commanding ψ. This method also has the advantage of explicitly accounting for constant ocean currents, without having to know or estimate them explicitly.

In figure 4.6 is represented a straight line path following scenario, with $\boldsymbol{V_w}$ representing the velocity of the vehicle with respect to the water, expressed in $\{B\}$, $\boldsymbol{V_c}$ is the velocity of the current expressed in $\{I\}$ and $\boldsymbol{V}$ is the total inertial velocity, also expressed in $\{I\}$. The sideslip angle is denoted by β, relatively to the velocity vector $\boldsymbol{V_w}$. Although the considered path following scenario is a straight line, it is possible to generalise this method to arbitrary paths by just computing the correspondent transformations – a method also used in the LOS algorithm.

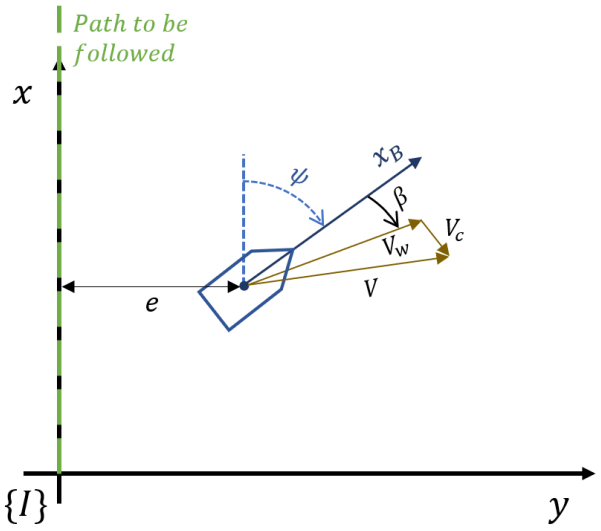


Figure 4.6: Cross-track error for straight line following.

If p denotes the position of the centre of mass of the vehicle and $R(\psi)$ denotes the rotation matrix from $\{B\}$ to $\{I\}$, parametrised by ψ, the variation of the position of the vehicle in $\{I\}$ is given by

$$\dot{\boldsymbol{p}} = R(\psi)\boldsymbol{V_w} + \boldsymbol{V_c}. \tag{4.14}$$

Equivalently, if one considers the sideslip angle β in the rotation matrix, $\boldsymbol{V_w}$ is aligned with the longitudinal body axis x_B, this results in

$$\dot{\boldsymbol{p}} = R(\psi + \beta)\begin{bmatrix}\|\boldsymbol{V_w}\| & 0\end{bmatrix}^T + \boldsymbol{V_c}. \tag{4.15}$$

This way, the evolution of the cross-track error satisfies

$$\dot{e} = \sin(\psi + \beta)\,\|\boldsymbol{V_w}\| + V_{cy}, \tag{4.16}$$

where V_{cy} denotes the component of $\boldsymbol{V_c}$ along the unit vector y, of $\{I\}$.

Therefore, the outer-loop controller, used to make e converge to zero, can be accomplished by, firstly, assuming that the heading angle can be commanded instantaneously, designing an outer-loop with heading as a command variable so that e converges to zero (virtual control), then, secondly, by using the actual inner-loop heading controller to track the commanded angle.

For the sake of simplicity, one can assume that the sideslip angle is zero and that the total velocity $\|\boldsymbol{V_w}\|$ can be held constant and equal to $U > 0$:

$$\dot{e} = U\sin(\psi) + V_{cy}, \tag{4.17}$$

which, in the absence of V_{cy}, results in

$$\dot{e} = Uu, \tag{4.18}$$

where $u = \sin(\psi)$. Assuming that it is possible to manipulate the variable u, one can define a control law that yields asymptotic exponential convergence of e to zero. This control law must satisfy

$$u = -\frac{K_1}{U}e, \tag{4.19}$$

which can be replaced in (4.18), resulting in the following solution, as desired: $e(t) = e(0)\exp(-K_1 t)$, $t \to +\infty$: $e(t) \to 0$.

Furthermore, the existence of a fixed bias V_{cy} motivates the introduction of an integral term in the virtual input u, tackling the existence of ocean currents, which means that it can be re-written as

$$u = -\frac{1}{U}\left(K_1 e + K_2 \int_0^t e(\tau)d\tau\right). \tag{4.20}$$

As a result, the cross-track error dynamics becomes

$$\dot{e} + K_1 e + K_2 \int_0^t e(\tau)d\tau = 0, \tag{4.21}$$

letting $\rho = \int_0^t e(\tau)d\tau$, then the following second order system is obtained: $\ddot{\rho} + K_1\dot{\rho} + K_2\rho = 0$. The second order system gains, K_1 and K_2, can be chosen in accordance to the desired natural frequency, ω_n, and damping factor, ζ. These shall be chosen as to comply with the previously mentioned assumption of instantaneous heading angle command (the inner-loop bandwidth has to be larger).

Having now a controller that generates the signal u, the above virtual control suggests that the desired command for heading is written as

$$\psi_{Maurya} = \arcsin(u) = \arcsin\left[\mathrm{sat}\left(-\frac{1}{U}\left[K_1 e + K_2 \int_0^t e(\tau)d\tau\right]\right)\right], \tag{4.22}$$

where $\mathrm{sat}(\cdot)$ is given by

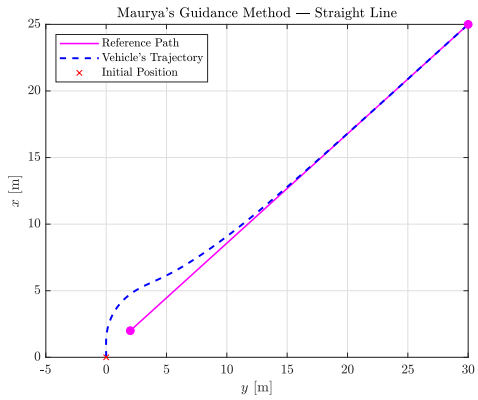
$$\mathrm{sat}(u) = \begin{cases} u & \text{if } |u| < l_s \\ l_s & \text{if } u \geq l_s \\ -l_s & \text{if } u \leq -l_s \end{cases}, \tag{4.23}$$

$0 < l_s < 1$, and this saturation function is introduced to guarantee that the argument of $\arcsin(\cdot)$ lies in the interval $[-1, +1]$.

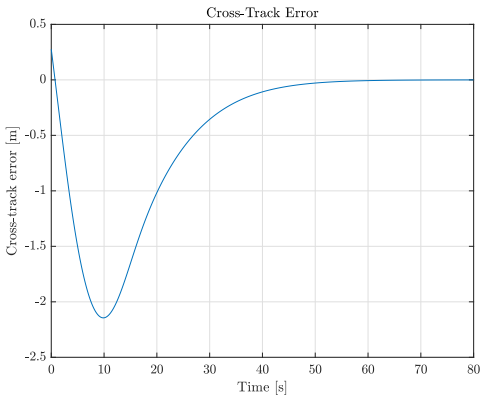
In order to obtain the cross-track error for a general straight line, for instance, one can use the same rotation matrix used in the LOS algorithm, defined in (4.4). However, in order to have the same representation as in figure 4.6, the cross-track error corresponds to the symmetric value of the y coordinate given by the rotation. This representation corresponds to having the x-axis of the transformed frame on top of the line and the y-axis describing a clockwise 90° angle with the x-axis. The same method can be applied to other arbitrary paths.

The above considerations conclude the explanation of the path following method proposed by Maurya. Implementing this algorithm yields the following results:

The gains $K_1 = 2\zeta\omega_n = 0.14$ and $K_2 = \omega_n^2 = 0.01$, were defined with $\zeta = 0.7$ and $\omega_n = 0.1$ rad/s.

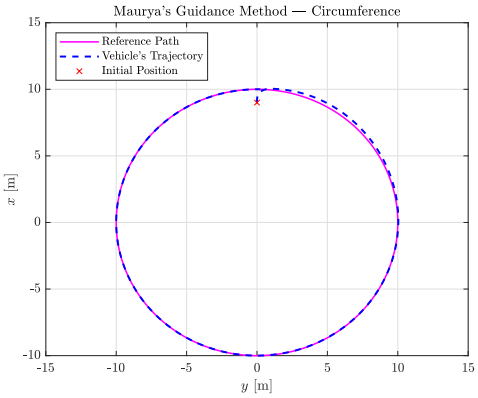


(a) Maurya's path following of a straight line.

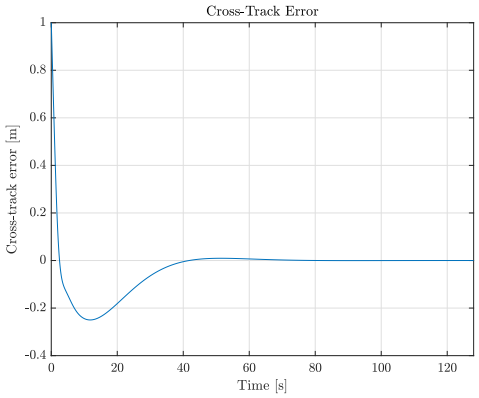


(b) Cross-track error evolution.

Figure 4.7: Simulation of the Maurya's guidance method for a straight line scenario.



(a) Maurya's path following of a circumference.



(b) Cross-track error evolution.

Figure 4.8: Simulation of the Maurya's guidance method for a circumferential path scenario.

Regarding the result for a generic path, figure 4.9, in comparison to the LOS algorithm, the Maurya's guidance method takes more time to converge, but shows better results following the curved segments (without offset).

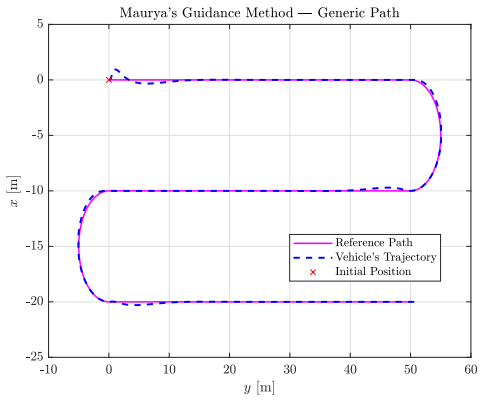


Figure 4.9: Maurya's guidance method for a generic path.

4.3 A. Micaelli and C. Samson's Guidance Method

This path following method is proposed by A. Micaelli and C. Samson in [2]. The method can be applied to an arbitrary path, using a Frenet frame $\{T\}$ defined on the curve, with the orthogonal projection of the vehicle's position onto the path, $\boldsymbol{P}$, as its origin.

The path following configuration is represented in figure 4.10. The position of the vehicle is represented by $\boldsymbol{M}$, with coordinates $[0, e]^T$ in the frame $\{T\}$, and $[X, Y]^T$ in the fixed inertial frame $\{I\}$. The signed curvilinear abscissa of $\{T\}$ along the curve is denoted as s and represents the traveled distance of this frame's origin measured on top of the path. The position of the vehicle can be equivalently expressed in the $\{T\}$ frame by the couple of variables $[s, e]^T$.

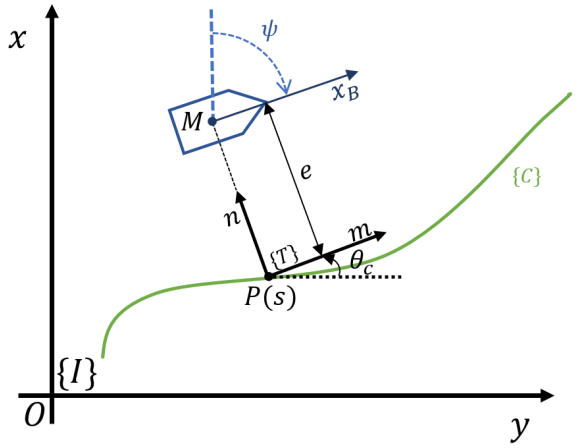


Figure 4.10: Configuration for the A. Micaelli and C. Samson's guidance method.

The rotation matrix from $\{I\}$ to $\{T\}$ is given by:

$$R_I^T(\theta_c) = \begin{bmatrix} \sin(\theta_c) & \cos(\theta_c) & 0 \\ \cos(\theta_c) & -\sin(\theta_c) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.24}$$

assuming the previously mentioned convention of the x-axis as a longitudinal coordinate and the y-axis as a transversal coordinate of the vehicle, thus, the columns are switched relatively to the traditional convention adopted in the paper.

In fact, considering the difference between the axis positioning convention adopted here and the one used in the paper, the θ_c used here expresses the rotation of the $\{T\}$ frame in the opposite direction when compared to the one in the paper. Due to this issue, $\beta = \pi/2 - \theta_c$ will replace θ_c, which is in fact more relevant considering the control objectives that will be addressed later on. Thus, let the rotation rate of frame $\{T\}$ be expressed in terms of this new variable, as in

$$\dot{\theta}_c = -c_c(s)\dot{s} \quad \Leftrightarrow \quad (\dot{\frac{\pi}{2}} - \beta) = -c_c(s)\dot{s} \quad \Leftrightarrow \quad \dot{\beta} = c_c(s)\dot{s}. \tag{4.25}$$

A classical law of mechanics gives

$$\left[\frac{d\overrightarrow{OM}}{dt}\right]_I = \left[\frac{d\overrightarrow{OP}}{dt}\right]_I + \left[\frac{d\overrightarrow{PM}}{dt}\right]_T + (\overrightarrow{w_c} \times \overrightarrow{PM}), \tag{4.26}$$

with

$$[\overrightarrow{\mathrm{PM}}]_T = \begin{bmatrix} 0 \\ e \\ 0 \end{bmatrix} \quad \text{and} \quad [\vec{w}_c]_I = \begin{bmatrix} 0 \\ 0 \\ \dot{\beta} = c_c(s)\dot{s} \end{bmatrix}, \tag{4.27}$$

where $[\vec{w}_c]_I$ is the rotation velocity vector of frame $\{T\}$ with respect to $\{I\}$, $\left(\frac{d}{dt}\right)_I$ means time differentiation with respect to the frame $\{I\}$ and $c_c(s)$ is the path's curvature at $\{T\}$.

One gets from (4.26)

$$\begin{bmatrix} \dot{s} \\ \dot{e} \\ 0 \end{bmatrix} = R_I^T \left(\frac{\pi}{2} - \beta\right) \begin{bmatrix} \dot{X} \\ \dot{Y} \\ 0 \end{bmatrix} + \begin{bmatrix} ec_c(s)\dot{s} \\ 0 \\ 0 \end{bmatrix} \tag{4.28}$$

and thus

$$\begin{cases} \dot{s} = \frac{\dot{X}\sin\left(\frac{\pi}{2}-\beta\right)+\dot{Y}\cos\left(\frac{\pi}{2}-\beta\right)}{1-c_c(s)e} \\ \dot{e} = \dot{X}\cos\left(\frac{\pi}{2}-\beta\right) - \dot{Y}\sin\left(\frac{\pi}{2}-\beta\right) \end{cases} \tag{4.29}$$

Considering the simplified kinematic model of the vehicle (with $v = 0$),

$$\begin{cases} \dot{x} = u\cos\psi \\ \dot{y} = u\sin\psi \\ \dot{\psi} = r \end{cases}, \tag{4.30}$$

along with the corresponding substitutions in (4.29), one can get the expression of the vehicle's position in frame $\{T\}$ as follows:

$$\begin{cases} \dot{s} = \frac{u\sin\left(\frac{\pi}{2}+\psi-\beta\right)}{1-c_c(s)e} \\ \dot{e} = u\cos\left(\frac{\pi}{2}+\psi-\beta\right) \\ \dot{\psi} = r \end{cases}. \tag{4.31}$$

Ultimately the goal is to make $\theta = \psi - \beta$ converge to zero (the orientation of the Frenet frame relatively to the vehicle's), in other words, to align the vehicle's main axis with the tangent to the curve – this is why β became more relevant than θ_c. As a result, (4.31) must be expressed now in terms of θ,

$$\begin{cases} \dot{s} = \frac{u\cos(\theta)}{1-c_c(s)e} \\ \dot{e} = -u\sin(\theta) \\ \dot{\psi} = r \end{cases}. \tag{4.32}$$

The result (4.32) is different from the one reached in [2] because the method had to accommodate the adopted conventions, mainly due to the difference in the inertial axis positioning.

Now that the kinematic model of the vehicle has been fully addressed, one can focus on introducing a nonlinear closed-loop control law to steer the dynamic model. This can be accomplished by using the kinematic model in (4.32) and by embodying the control objectives in a Lyapunov candidate function – Lyapunov-oriented control design. These control objectives essentially consist of driving e and θ to zero, considered in the following Lyapunov function

$$V = \frac{1}{2}\left[f^2(e) + \frac{1}{\lambda_\theta}(\theta - \delta(e,u))^2\right], \tag{4.33}$$

where it is assumed that

$A_1 : f(\pm a) = \pm\infty$

$A_2 : f(0) = \delta(0,u) = 0, \forall u$

$A_3 : f'_e(e) > 0, \forall e$

$A_4 : uf(e)\sin(\delta(e,u)) \geq 0, \forall e, \forall u.$

The functions f and δ are introduced in order to broaden the control stability domain and achieve additional user's objectives. In this Lyapunov function, the first term penalises the cross-track error, while the second one is concerned with how the vehicle should be oriented. The δ function can interpreted as the desired value of the orientation θ during transients, allowing for orientation flexibility especially when, for instance, the vehicle is considerably distant from the curve, letting it approach further the curve before being specifically concerned with the relative orientation to the Frenet frame (A_4) – only after this should the vehicle's main axis be tangent to the path, when e finally approaches zero (A_2).

With this in mind, the goal is to find r capable of fulfilling the control objectives and $\dot{V} \leq 0$, ensuring that the vehicle actually converges to the path, while considering the aforementioned assumptions. Therefore, looking into the theory exposed in the paper, one can assume that the following r will fulfil the control objectives while keeping $\dot{V} \leq 0$:

$$r = c_c\frac{u\cos\theta}{1 - c_c e} - \delta'_e u\sin\theta + \delta'_u\dot{u} + \lambda_\theta f f'_e u\frac{\sin\theta - \sin\delta}{\theta - \delta} - k\lambda_\theta|u|(\theta - \delta); \quad k > 0, \lambda_\theta > 0. \tag{4.34}$$

In the paper by Micaelli and Samson [2], the chosen control variable is the angular velocity r, instead of ψ, as has been considered so far. This implies that, for now, in order to test this algorithm, one has to derive a new controlled system with r as a control variable. Knowing r, one can finally steer the vehicle to the path at a desired speed.

Regarding the choice of the approach angle $\delta(e,u)$, one can look to the suggestion presented in [2], $\delta(e,u) = \mathrm{sign}(u)\theta_a\tanh(e)$, however it is not differentiable with respect to u at $u = 0$, so

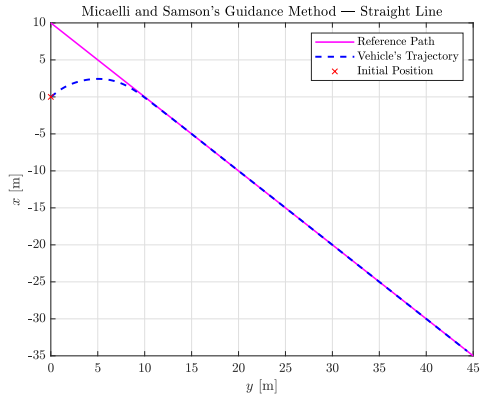
$$\delta(e,u) = \theta_a\tanh(k_\delta e u), \tag{4.35}$$

where $0 < \theta_a < \pi/2$ and k_δ is an arbitrary positive gain, was considered instead.

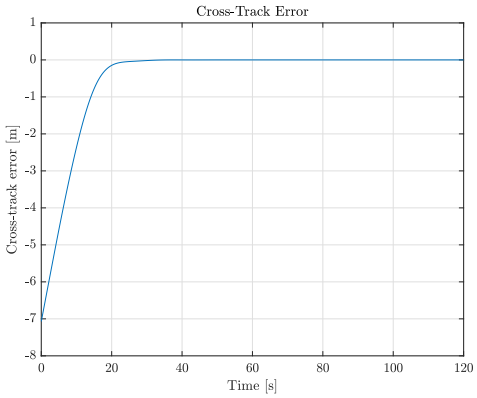
Finally, at this point it is important to notice that the position of the vehicle $\boldsymbol{M}$ in $\{T\}$, given by $[0,e]^T$ (since the location of point $\boldsymbol{P}$ is defined by the projection of $\boldsymbol{M}$ on the path), has a null coordinate that

causes $1 - c_c(s)e$ to appear in the denominator of $\dot{s}$ in (4.32), thus creating a singularity at $e = 1/c_c$. As a result, the derived control law requires that the initial position of M be restricted to a tube around the path, the radius of which must be less than $1/c_{c,max}$, where $c_{c,max}$ denotes the maximum curvature of the path. Clearly, this constraint is very conservative, imposing a rather strict constraint on the initial vehicle's position (affecting only the initial position because, once inside the accepted region, the vehicle converges to the path). However, the upcoming and last guidance method to be studied will explore an improvement to Micaelli and Samson's method, fixing this singularity issue.

Lastly, implementing the Micaelli and Samson's method, keeping in mind the restriction to the initial vehicle's position (within a radius of $1/c_{c,max}$) and a constant vehicle speed, yields the following results:

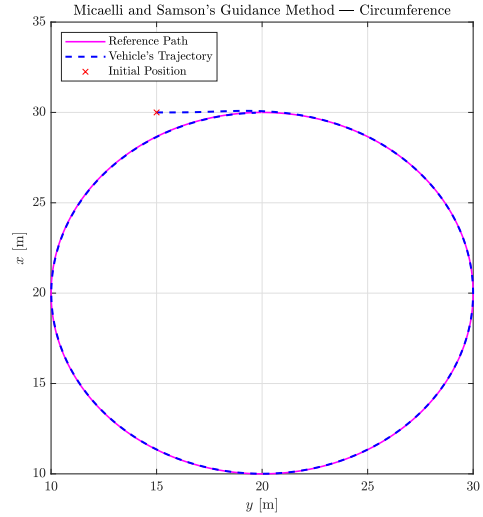


(a) Micaelli and Samson's path following of a straight line.

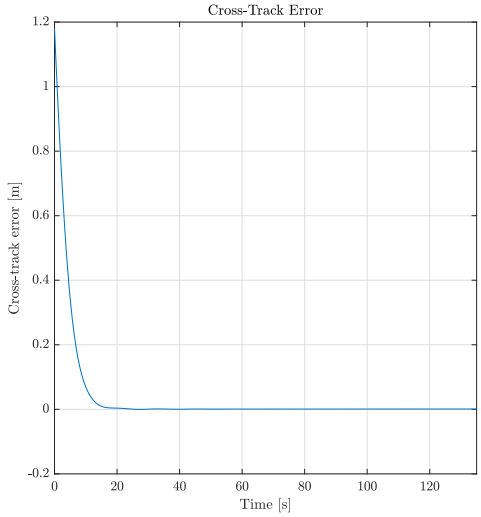


(b) Cross-track error evolution.

Figure 4.11: Simulation of the Micaelli and Samson's guidance method for a straight line scenario.



(a) Micaelli and Samson's path following of a circumference.

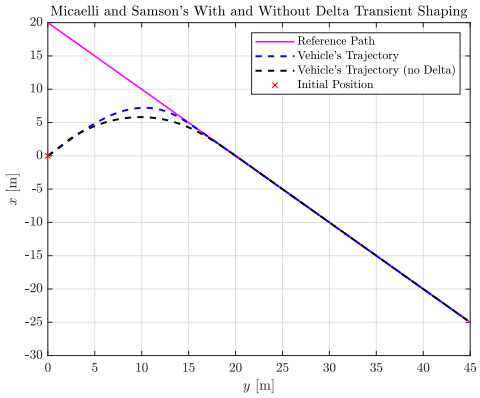


(b) Cross-track error evolution.

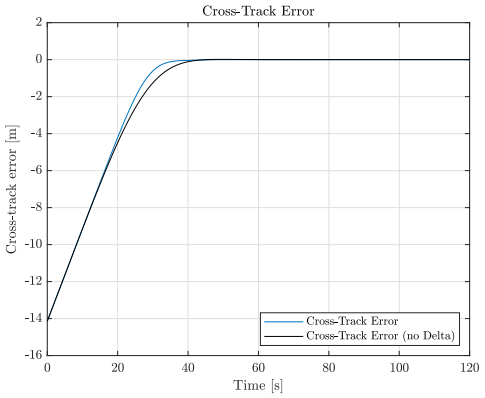
Figure 4.12: Simulation of the Micaelli and Samson's guidance method for a circumferential path scenario.

The previous results benefited from shaping the transient manoeuvrers during the path approach phase, introduced by the choice of the $\delta(e, u)$ function defined in (4.35). The next results explore what

happens if no $\delta(e, u)$ were used.



(a) Micaelli and Samson's path following of a line with and without the $\delta(e, u)$ effect.



(b) Cross-track error evolution.

Figure 4.13: Simulation of the Micaelli and Samson's guidance method for a straight line path scenario with and without the $\delta(e, u)$ effect shaping the transient manoeuvre.

As it could be expected, using an appropriate $\delta(e, u)$ allows the vehicle to enter an approach phase, where it can prioritize approaching first the path and only afterwards make its main axis tangent to the path, as e gets closer to zero. This results in the behaviour that can be observed in figure 4.13, where with the $\delta(e, u)$ effect, the vehicle converges faster to the path due to the shaping of the transient manoeuvre during the path approach phase. In short, this Lyapunov-oriented control design allowed for the introduction of extra control objectives that benefit substantially the end result, such as the $\delta(e, u)$ function.

Nevertheless, this method still has the singularity issue as a major disadvantage, imposing a strict constraint on the initial vehicle's position, as explained before. As a result, in [2] only local convergence of the actual path of the vehicle to the desired path can be proven. Therefore, one should look into the theory of the next path following method, which aims to improve upon Micaelli and Samson's work.

4.4 L. Lapierre's Guidance Method

This path following method is proposed by L. Lapierre, D. Soetanto, and A. Pascoal in [8]. It aims to improve upon Micaelli and Samson's work [2], overcoming the stringent initial condition constraint present in the previously studied path following method.

The initial condition constraint is solved by controlling explicitly the rate of progression of a "virtual target" to be tracked along the path, thus bypassing the problems that arise when the position of the virtual target is simply defined by the projection of the actual vehicle on that path, as explained in [8]. This procedure avoids the singularity that occurs when the distance to the path is not well defined and allows for a proof of global convergence of the actual path of the vehicle to the desired path, contrarily to the Micaelli and Samson's method where only local convergence has been proven.

Considering the advantages that these Lyapunov-oriented control design methods bring to the path following problem ([2], [8]), such as the additional control objectives that have proven to improve sub-

stantially the results, and also being able to use these methods for arbitrary paths; moving forward, L. Lapierre's method will be the chosen path following solution to be ultimately implemented to steer the vehicles. As a result, this section will explore the method in a more complete manner, comparatively to what has been done with the previous methods.

4.4.1 Path Following Control Design

The path following configuration is represented in figure 4.14 – adapted from the previous method. The position of the vehicle is represented by the centre of mass $\boldsymbol{M}$, with coordinates $[s_1, y_1]^T$ in the frame $\{T\}$, and $[X, Y]^T$ in the fixed inertial frame $\{I\}$. The signed curvilinear abscissa of $\boldsymbol{P}$ along the path is denoted s, which is associated with the Serret-Frenet frame $\{T\}$.

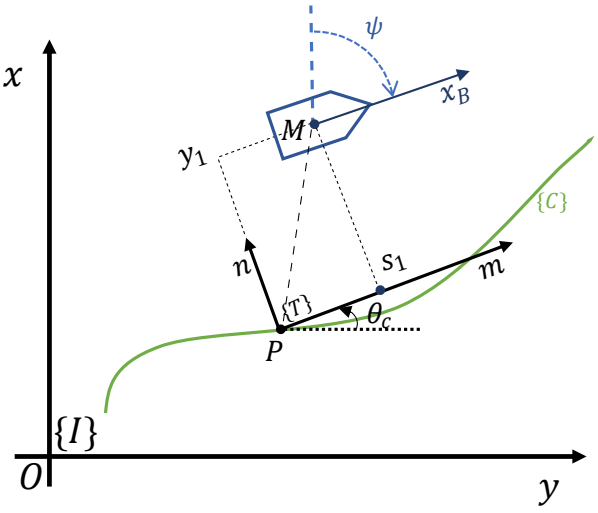


Figure 4.14: Configuration for the L. Lapierre's guidance method.

One significant difference between this method and the previous one, by Micaelli and Samson, is that now the Frenet frame is not attached to the point on the path that is closest to the vehicle. Instead, the origin of $\{T\}$ along the path is made to evolve according to a conveniently defined function of time, effectively yielding an extra controller design parameter, seemingly allowing to lift the initial condition constraint. The Frenet frame $\{T\}$ now plays the role of the body axis of a "virtual target vehicle" that should be tracked by the real vehicle.

Once again, let

$$R_I^T\left(\frac{\pi}{2}-\beta\right) = \begin{bmatrix} \sin\left(\frac{\pi}{2}-\beta\right) & \cos\left(\frac{\pi}{2}-\beta\right) & 0 \\ \cos\left(\frac{\pi}{2}-\beta\right) & -\sin\left(\frac{\pi}{2}-\beta\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.36}$$

be the rotation matrix from $\{I\}$ to $\{T\}$, parametrized locally by the angle β (considering that $\theta_c = \pi/2 - \beta$), instead of θ_c, because β will be more useful to express the control objective of driving a certain $\theta = \psi - \beta$ to zero, under this different axis positioning convention, when compared to the configuration present in [8]. This control objective aims to align the vehicle's main body axis with the tangent to the curve.

Moreover, the rotation rate of the frame $\{T\}$ is given by (4.25), expressed in terms of β as well.

Using the same classical law of mechanics defined in (4.26) with the relations

$$[\overrightarrow{\mathrm{OM}}]_I = \begin{bmatrix} X \\ Y \\ 0 \end{bmatrix}, \quad [\overrightarrow{\mathrm{OP}}]_I = \begin{bmatrix} s \\ 0 \\ 0 \end{bmatrix}, \quad [\overrightarrow{\mathrm{PM}}]_T = \begin{bmatrix} s_1 \\ y_1 \\ 0 \end{bmatrix}, \tag{4.37}$$

and

$$\vec{w}_c \times \overrightarrow{PM} = \begin{bmatrix} 0 \\ 0 \\ \dot{\beta} = c_c(s)\dot{s} \end{bmatrix} \times \begin{bmatrix} s_1 \\ y_1 \\ 0 \end{bmatrix} = \begin{bmatrix} -c_c(s)\dot{s}y_1 \\ c_s(s)\dot{s}s_1 \\ 0 \end{bmatrix}, \tag{4.38}$$

where $[\vec{w}_c]_I$ is the rotation velocity vector of frame $\{T\}$ with respect to $\{I\}$ and $c_c(s)$ is the path's curvature at $\{T\}$, one gets

$$R_I^T\left(\frac{\pi}{2} - \beta\right) \begin{bmatrix} \dot{X} \\ \dot{Y} \\ 0 \end{bmatrix} = \begin{bmatrix} \dot{s}\left[1 - c_c(s)y_1\right] + \dot{s}_1 \\ \dot{y}_1 + c_c(s)\dot{s}s_1 \\ 0 \end{bmatrix}. \tag{4.39}$$

Solving for $\dot{s_1}$ and $\dot{y_1}$ yields

$$\begin{cases} \dot{s_1} = \dot{X}\sin\left(\frac{\pi}{2} - \beta\right) + \dot{Y}\cos\left(\frac{\pi}{2} - \beta\right) - \dot{s}\left(1 - c_c y_1\right) \\ \dot{y_1} = \dot{X}\cos\left(\frac{\pi}{2} - \beta\right) - \dot{Y}\sin\left(\frac{\pi}{2} - \beta\right) - c_c\dot{s}s_1 \end{cases}. \tag{4.40}$$

At this point it is important to notice that by making s_1 not necessarily equal to zero (as it happens in the Micaelli and Samson's method), a virtual target that is not coincident with the projection of the vehicle on the path is created, thus introducing an extra degree of freedom for controller design. By specifying how fast the target moves, the occurrence of a singularity at $y_1 = 1/c_c$ is removed.

Substituting the simplified kinematic model of the vehicle, expressed in (4.30), in (4.40) and introducing the variable $\theta = \psi - \beta$, results the kinematic model of the vehicle in the (s_1, y_1) coordinates given by

$$\begin{cases} \dot{s}_1 = -\dot{s}\left(1 - c_c y_1\right) + u\cos(\theta) \\ \dot{y}_1 = -c_c\dot{s}s_1 - u\sin(\theta) \\ \dot{\theta} = r - c_c\dot{s} \end{cases}, \tag{4.41}$$

where $r = \dot{\psi}$.

Now that the kinematic model of the vehicle in terms of (s_1, y_1) coordinates has been addressed, one can start fulfilling the main objective of the path following control law that is to drive y_1 and θ to zero, using a nonlinear controller design strategy based on this kinematic model. These control objectives can be embodied in a Lyapunov candidate function

$$V_1 = \frac{1}{2}\left(s_1^2 + y_1^2\right) + \frac{1}{2\gamma}\left(\theta - \delta\left(y_1, u\right)\right)^2, \tag{4.42}$$

where it is assumed that

A.1: $\delta(0, u) = 0$.

A.2: $y_1 u \sin\delta\,(y_1, u) \geq 0, \forall y \forall u$.

A.3: $\lim_{t\to\infty} u(t) \neq 0$.

In the adopted V_1 Lyapunov function, the first term captures the distance between the vehicle and the path, which must be driven to zero. The second term aims to shape the approach angle $\theta = \psi - \beta$ of the vehicle to the path, by forcing it to follow a desired orientation profile given by the function δ.

Assumption A.1 imposes the condition that the vehicle's main axis must be tangent to the path when the distance to it, y_1, reaches zero. In turn, assumption A.2 provides an adequate reference sign definition in order to steer the vehicle to the path, considering when it is on the right side relatively to the path or when it is on its left side. Finally, assumption A.3 states that the vehicle must not tend to a state of rest. These assumptions are needed to ensure $\dot{V}_1 \leq 0$ and, thus, to ensure that the vehicle actually converges to the path.

The derivative of V_1 gives

$$\begin{aligned}\dot{V}_1 &= s_1\dot{s}_1 + y_1\dot{y}_1 + \frac{1}{\gamma}(\theta-\delta)(\dot{\theta}-\dot{\delta}) \\ &= s_1\left(u\cos\theta - \dot{s}\left(1 - c_c y_1\right) - \dot{s}c_c y_1\right) - y_1 u\sin\theta + \frac{1}{\gamma}(\theta-\delta)(\dot{\theta}-\dot{\delta}) \\ &= s_1(u\cos\theta - \dot{s}) - y_1 u\sin\delta + \frac{1}{\gamma}(\theta-\delta)\left(\dot{\theta} - \dot{\delta} - \gamma y_1 u\frac{\sin\theta - \sin\delta}{\theta-\delta}\right)\end{aligned} \quad . \tag{4.43}$$

Letting the ideal kinematic control laws for s and θ, given by the authors of [8], be defined as

$$\begin{aligned}\dot{s} &= u\cos\theta + k_1 s_1 \\ \dot{\theta} &= \dot{\delta} + \gamma y_1 u\tfrac{\sin\theta - \sin\delta}{\theta-\delta} - k_2(\theta-\delta)\end{aligned} \quad , \tag{4.44}$$

where k_1 and k_2 are positive gains. Then,

$$\dot{V}_1 = -k_1 s_1^2 - y_1 u\sin\delta - k_2\frac{(\theta-\delta)^2}{\gamma} \leq 0, \tag{4.45}$$

where the second term, $y_1 u\sin\delta$, respects the inequality through assumption A.2. As a result, these control laws can be used to achieve the convergence of the vehicle to the path at a desired speed, hence accomplishing path following.

Given that ultimately the vehicle is steered by receiving values of $r = \dot{\psi}$ and recalling (4.41),

$$\begin{aligned}r &= \dot{\theta} + c_c\dot{s} \\ &= \dot{\delta} + \gamma y_1 u\frac{\sin\theta - \sin\delta}{\theta-\delta} - k_2(\theta-\delta) + c_c\dot{s}.\end{aligned} \tag{4.46}$$

The choice of the $\delta(y_1, u)$ function is instrumental in shaping the transient manoeuvrers during the path approach phase, as explained before in the context of the Micaelli and Samson's method. Looking again into the references [2] and [8], it is recommended the usage of

$$\delta\,(y_1, u) = \theta_a \tanh\left(k_\delta y_1 u\right), \tag{4.47}$$

where $0 < \theta_a < \pi/2$, k_δ is an arbitrary positive gain and $\delta(y_1, u)$ is differentiable with respect to u at $u = 0$.

4.4.2 Path Generation and Parametrization

Now that path following can be accomplished using the method proposed in [8], one should address the path generation problem. This is required in order to feed measures of (s_1, y_1), $c_c(s)$ and θ_c (or $\beta = \pi/2 - \theta_c$) to the aforementioned virtual control laws, as illustrated in figure 4.15.

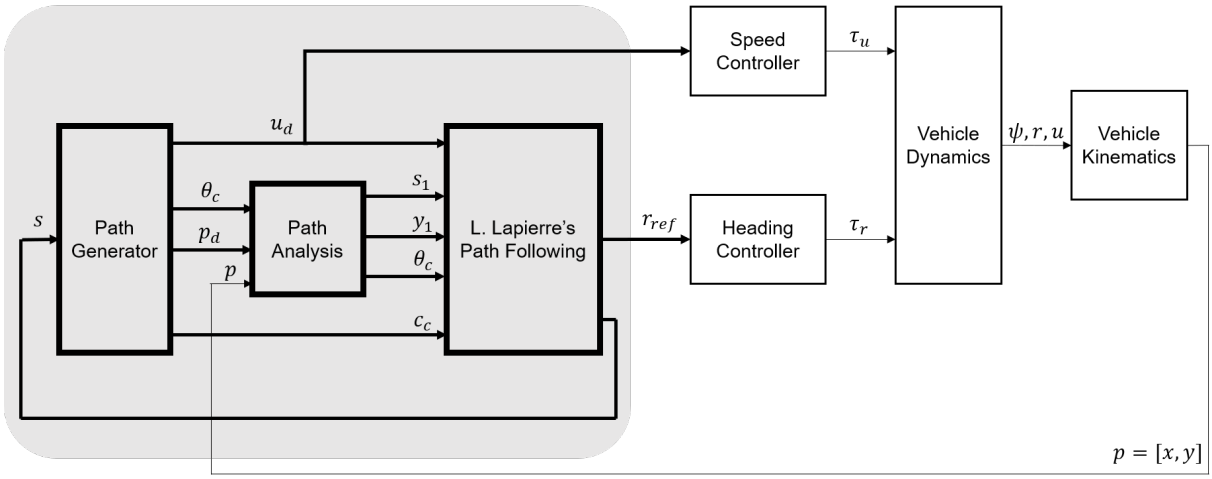


Figure 4.15: Detailed view of the path following block diagram (single vehicle).

The path generation block requires a priori knowledge of the path the vehicle has to follow, as desired by the user. Moving forward, the ultimate goal is to make the vehicle converge to a lawnmower-type path, which is composed of straight line segments and half-circumferences that make the vehicle describe a curve in between straight lines (as has been explored in previous methods). Therefore, one has to tackle the path generation of a straight line and, consecutively, the path generation of a circumference, before moving forward to the lawnmower path, which uses both of these elements interchangeably.

The generation method uses the arc-length distance s to determine the desired point of the path to where the vehicle has to converge, $\boldsymbol{p_d} = [x_d, y_d]^T$, as well as the inclination of the tangent to that point, θ_c. Determining p_d makes it possible to compute (s_1, y_1) (knowing $\boldsymbol{p} = [x, y]^T$); and determining θ_c makes it possible to compute the rotation matrix $R_I^T(\theta_c)$ and the angle $\beta = \pi/2 - \theta_c$, which is used, in turn, to compute $\theta = \psi - \beta$ – required by the virtual path following control laws.

It is assumed access to a speed controller that is responsible for driving the speed of the vehicle to a certain value. In this case, a constant desired speed is considered for all the path segments, u_d, which serves as an input to the path following algorithm and the speed controller, given by the path generation block. Moreover, the results benefit from the usage of a previously acquired nonlinear model of the MEDUSA-class vehicles, to which the previously designed heading controller was added (in this case, for r).

Acknowledging that $\boldsymbol{\alpha} = [x_d(s), y_d(s)]^T$ is a proper arc-length parametric representation of a twice differentiable curve (where $\frac{d\alpha}{ds}$ is differentiable and different from zero), then the signed curvature can be

defined as

$$c_c(s) = \frac{x'_d y''_d - y'_d x''_d}{\left(\sqrt{x'^2_d + y'^2_d}\right)^3}, \tag{4.48}$$

where $(\cdot)'$ means $\frac{d\cdot}{ds}$ and $(\cdot)''$ means $\frac{d^2\cdot}{ds^2}$ (from reference [9, Chapter 13.3]).

However, it is well known that a straight line has no curvature, so $c_c(s) = 0$ for a straight line, and it is also well known that the curvature of a circumference is given by

$$c_c(s) = \frac{1}{R}, \tag{4.49}$$

where R denotes the radius of the circumference. This curvature takes a negative sign if the goal is to achieve a counter-clockwise rotation. As a result, the path dependent curvature has been addressed and, consequently, the path generator's c_c output as well.

Although $\boldsymbol{\alpha} = [x_d(s), y_d(s)]^T$ is a possible arc-length parametrization, one should prefer $\boldsymbol{\alpha} = [x_d(\gamma), y_d(\gamma)]^T$ as a parametric representation of the curve, where γ corresponds to the normalized arc-length. The preference for γ, the normalized arc-length, instead of for s, the arc-length, will become more evident when matters of Cooperative Control are addressed later on.

In general, for a straight line, $\boldsymbol{p_d}$ can be given by

$$\boldsymbol{p_d} = \begin{bmatrix} x_d(\gamma) \\ y_d(\gamma) \end{bmatrix} = \begin{bmatrix} x_0 + \gamma l \sin(\theta_l) \\ y_0 + \gamma l \cos(\theta_l) \end{bmatrix}, \tag{4.50}$$

where l denotes the length of the straight line segment, the normalized arc-length is given by $\gamma = s/l$, θ_l denotes the inclination of the desired straight line, and $[x_0, y_0]^T$ is the line's starting point. For instance, for a vertical straight line segment: $\boldsymbol{p_d} = [x_0 + \gamma l, y_0]^T$, with $\theta_l = \pi/2$; and for a horizontal straight line segment: $\boldsymbol{p_d} = [x_0, y_0 + \gamma l]^T$, with $\theta_l = 0$.

Considering a circumferential path, p_d can be given by

$$\boldsymbol{p_d} = \begin{bmatrix} x_d(\gamma) \\ y_d(\gamma) \end{bmatrix} = \begin{bmatrix} x_0 + R\sin(2\pi\gamma) \\ y_0 + R\left[1 - \cos(2\pi\gamma)\right] \end{bmatrix}, \tag{4.51}$$

where R denotes the radius, the normalized arc-length is given by $\gamma = s/(2\pi R)$, and $[x_0, y_0]^T = [x_c + R\sin(\pi), y_c + R\cos(\pi)]^T$ (where $[x_c, y_c]^T$ is the centre of the circumference). The argument of the sine in $x_0 + R\sin(2\pi\gamma)$ should be negative if a counter-clockwise rotation is desired. With this in mind, the path generator block is now able to compute $\boldsymbol{p_d}$ for both scenarios.

The next step is to be able to compute θ_c and from there to be able to compute $\beta = \pi/2 - \theta_c$ as well. For a straight line with inclination θ_l, the angle θ_c is simply given by

$$\theta_c = \theta_l, \text{ with } \beta = \frac{\pi}{2} - \theta_l. \tag{4.52}$$

For a circumference, θ_c, in terms of γ, can be given by

$$\theta_c = \arctan\left[\frac{\cos(2\pi\gamma)}{\sin(2\pi\gamma)}\right], \text{ with } \beta = \frac{\pi}{2} - \arctan\left[\frac{\cos(2\pi\gamma)}{\sin(2\pi\gamma)}\right]. \tag{4.53}$$

However, this last result has to be compensated with $-\pi$ should a counter-clockwise rotation be desired. As result, the path generator is now able to compute θ_c and β. The equations that compute θ_c were derived from the general expression of the unit tangent vector (from reference [9, Chapter 13.2])

$$\boldsymbol{T}(\gamma) = \frac{\boldsymbol{p'_d}(\gamma)}{|\boldsymbol{p'_d}(\gamma)|}, \tag{4.54}$$

where $\boldsymbol{p'_d}$ means $\frac{dp_d}{d\gamma}$, which indicates the direction of the curve at γ. Knowing this unit tangent vector, θ_c is simply computed by taking the arctangent of its x component over its y component, recalling the adopted axis conventions – this is from where one gets the equations (4.52) and (4.53).

All of this information can now be used to address the last missing component: being able to compute $[s_1, y_1]^T$, knowing the position of the vehicle $\boldsymbol{p} = [x, y]^T$ (given by the kinematics), the desired position $\boldsymbol{p_d} = [x_d, y_d]^T$, and the inclination of the tangent to the curve θ_c. This can be achieved by computing

$$\begin{bmatrix} s_1 \\ y_1 \\ 0 \end{bmatrix} = R_I^T(\theta_c) \begin{bmatrix} x - x_d \\ y - y_d \\ 0 \end{bmatrix}, \tag{4.55}$$

recalling the correct definition of $R_I^T(\theta_c)$ in (4.24), considering the adopted conventions.

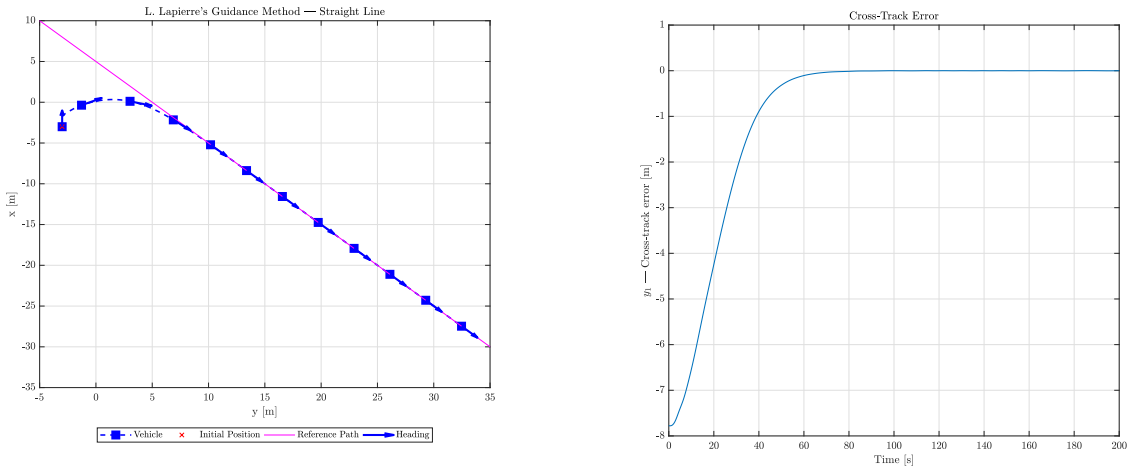
Finally, the path generator is complete and the outer-loop path following control has access to all the required input quantities. This way, one is finally able to compute r_{ref} and feed it to the heading controller, as well as output the desired constant speed of the vehicle to the speed controller, as previously illustrated in figure 4.15.

Considering the lawnmower path, one just has to adapt the previous equations to the desired lawnmower configuration and use them interchangeably. Switching between segments should be based on the inspection of s, transitioning when s reaches certain desired values. The state γ should be computed given s and the length of each segment, additionally it should evolve continuously from zero to one for the first segment, then from one to two for the second segment, and so on ($\gamma = [0...1...2...n]$, where n is the total number of segments composing the path). Naturally, considering this continuous profile of the evolution of γ, the previous equations should be compensated by subtracting the normalized arc-length of the previous segment to the current value of γ. This definition of γ will make it easier when matters of Cooperative Control are addressed later on and it also avoids unwanted discontinuities.

4.4.3 Results

Lastly, implementing the L. Lapierre's method, with the path generation and parametrization as explained before and a constant vehicle speed, yields the following results:

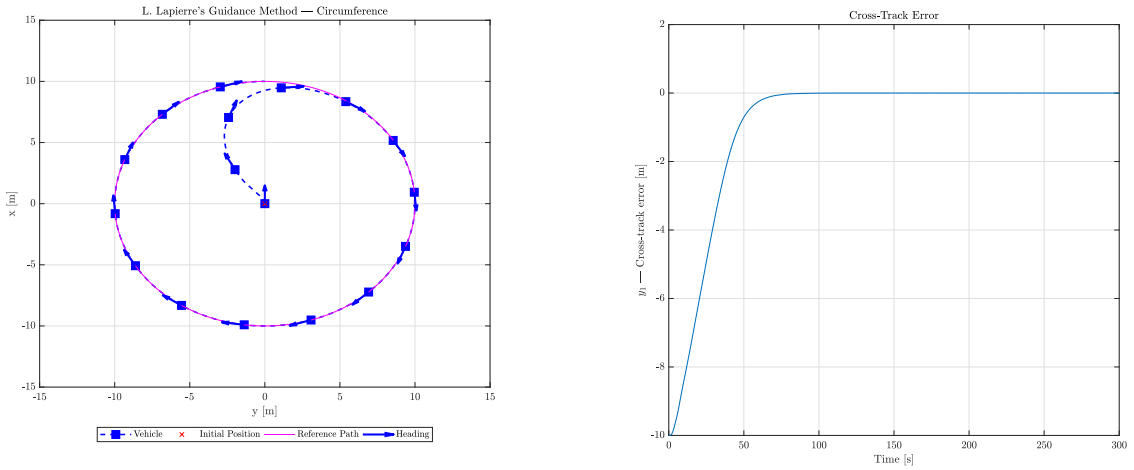
Observing the results in figures 4.16 and 4.17, the L. Lapierre's path following method achieves proper path following for a straight line and circumferential path scenarios, taking advantage of the



(a) L. Lapierre's path following of a straight line path.

(b) Cross-track error evolution.

Figure 4.16: Simulation of the L. Lapierre's guidance method for a straight line path scenario.



(a) L. Lapierre's path following of a circumferential path.

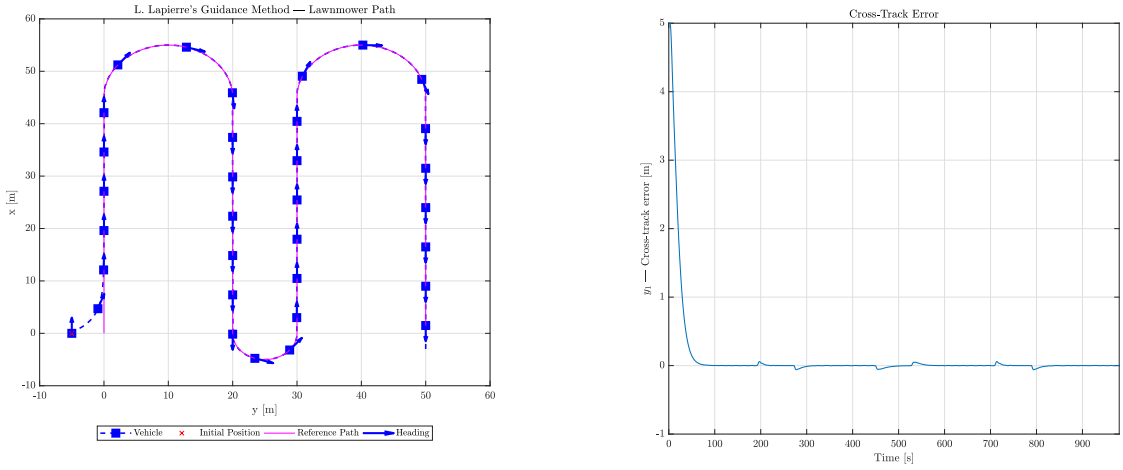
(b) Cross-track error evolution.

Figure 4.17: Simulation of the L. Lapierre's guidance method for a circumferential path scenario.

approach phase with the shaping of the transient manoeuvres imposed by $\delta(y_1, u)$. It is important to notice, however, that for the circumferential path in figure 4.17, the initial position can be set at the centre of the circumference. This wouldn't be possible with the Micaelli and Samson's method, because of the occurrence of the singularity at $y_1 = 1/c_c$, which raises an initial position restriction: the initial position has to be within a radius of $1/c_{c,max}$ relatively to the path (which in this case means that it must start within a "tube" with the radius of the circumference around the path, excluding the upper and lower limits of $\pm R$). That being said, the result in figure 4.17 proves that, in fact, L. Lapierre's method solves the singularity and the initial position restriction issues, as initially proposed.

Moving forward to the simulation results in figures 4.18 and 4.19, these prove that the method can achieve proper path following for arbitrary paths, just as long as one can parametrize the paths as it was explained in the previous section. The result in figure 4.19 even goes a step further than the lawnmower path, by introducing a sine wave segment in the end, which had to be parametrized and had to use the

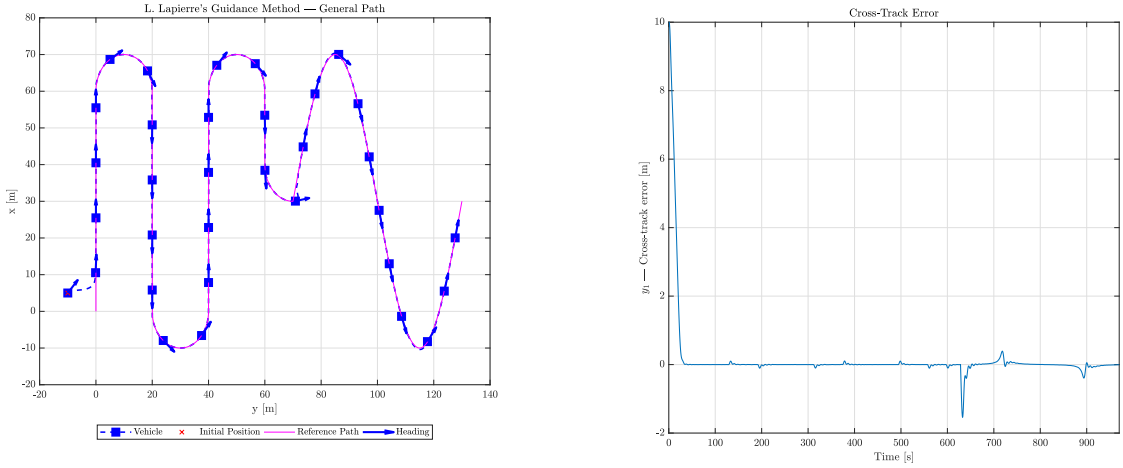
general equations of the curvature, (4.48), and of the inclination of the tangent to the path, (4.54).



(a) L. Lapierre's path following of a lawnmower path.

(b) Cross-track error evolution.

Figure 4.18: Simulation of the L. Lapierre's guidance method for a lawnmower path scenario.



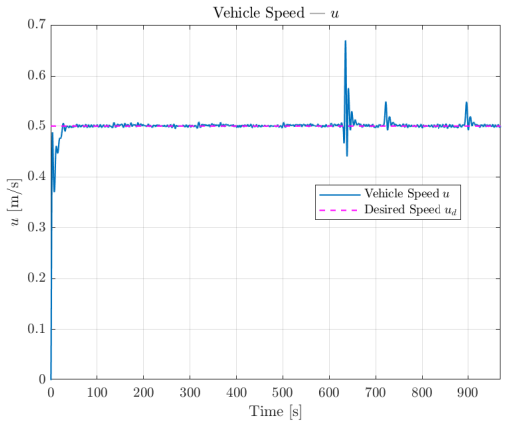
(a) L. Lapierre's path following of a general path.

(b) Cross-track error evolution.

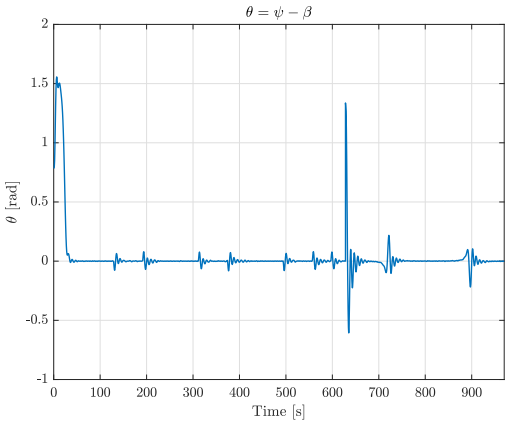
Figure 4.19: Simulation of the L. Lapierre's guidance method for a general path scenario.

Regarding the path following scenario represented in figure 4.19, figure 4.20 shows the speed of the vehicle converging to the desired value and $\theta = \psi - \beta$ converging to zero, as desired. When the vehicle transitions from one segment to the other, the convergence of u and θ is disturbed, especially when the vehicle transitions to the sine wave (because the change is more abrupt in this case), as expected.

This concludes the results section of the L. Lapierre's path following method. As explained before, moving forward, this is the chosen method to achieve path following with the underwater vehicles, recalling the advantages that it brings in terms of being able to use it with arbitrary paths and giving the additional flexibility (such as the approach phase to the path), with proven global convergence of the actual path of the vehicle to the desired path (without initial position restrictions).



(a) Convergence of the vehicle speed to the desired value.



(b) Convergence of θ to zero.

Figure 4.20: Evolution of the vehicle speed and the value of θ during the simulation of a general path following scenario (lawnmower + sine wave).

4.5 Path Following with Ocean Currents

This section will explore a method to deal with ocean currents while path following is achieved. Simply put, the method feeds heading control references that are meant to align the total velocity vector of the vehicle with the trajectory, instead of the vehicle's main body axis. To achieve this, a current observer, in the fixed inertial frame, is used so that it is possible to then compensate the heading references with the intent of canceling the perpendicular component of the velocity of the current to the trajectory.

In order to consider the influence of ocean currents, the inner-loop heading controller must accept yaw angle references, ψ_{ref}, instead of yaw rate references, r_{ref}, as the L. Lapierre's path following method considers. This means that the outer-loop path following virtual control laws must be adapted to yaw angle references and that the inner-loop controller must be designed as explained in chapter 3, knowing that the goal is to compensate the heading angle of the vehicle.

L. Lapierre's Path Following Method with Ocean Currents

Firstly, it is important to consider the velocity notation represented in figure 4.21.

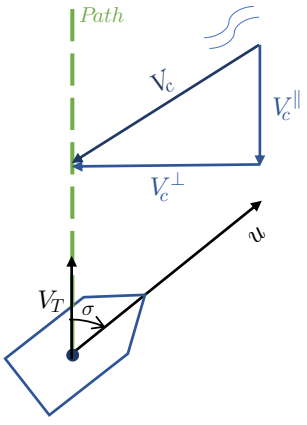


Figure 4.21: Ocean currents velocity notation.

Where:

- $\boldsymbol{V_c}$ – velocity vector of the ocean current;
- $V_c^{\parallel}$ – parallel component of the ocean current to the trajectory;
- $V_c^{\perp}$ – perpendicular component of the ocean current to the trajectory.

The total velocity vector of the vehicle is denoted by $\boldsymbol{V_T}$, the goal being to align this velocity vector with the trajectory, which causes an angle mismatch, σ, relatively to the vehicle's heading – course angle. The surge velocity, along the vehicle's main body axis, is depicted by u, the vessel's velocity relatively to the water.

Going back to the kinematic model of the vehicle in the Serret-Frenet frame, in (4.41), given by the L. Lapierre's method, and taking the ocean currents into account, the new kinematic model is given by

$$\begin{cases} \dot{s}_1 = -\dot{s}\left(1 - c_c y_1\right) + u\cos(\theta) + V_c^{\parallel} \\ \dot{y}_1 = -c_c \dot{s} s_1 - u\sin(\theta) + V_c^{\perp} \end{cases}. \tag{4.56}$$

Using the same methodology as before, it is possible to apply the Lyapunov's direct method now to design the outer-loop path following controller, by choosing a control law that makes the Lyapunov function decrease, thus guaranteeing stability as well. Unfortunately, the second term of the previously used Lyapunov function, in (4.42), won't be considered this time, because the goal is to derive now a control law for the heading angles, ψ, and not for the heading rate, r, and it becomes troublesome to find a virtual control law while still considering the second term of the previous Lyapunov function. For this reason, the new Lyapunov function is defined as

$$V_1 = \frac{1}{2}\left(s_1^2 + y_1^2\right), \tag{4.57}$$

considering the new kinematic model defined in (4.56), the derivative of the Lyapunov function gives

$$\begin{aligned} \dot{V}_1 &= s_1\dot{s}_1 + y_1\dot{y}_1 \\ &= s_1[-\dot{s} + u\cos(\theta) + V_c^{\parallel}] + y_1[-u\sin(\theta) + V_c^{\perp}] \end{aligned}. \tag{4.58}$$

The ideal kinematic control laws for s and θ are now given by

$$\begin{aligned} \dot{s} &= u\cos\theta + k_1 s_1 + V_c^{\parallel} \\ \theta &= \arcsin\left[\mathrm{sat}\left(k_2 y_1 + \tfrac{V_c^{\perp}}{u}\right)\right] \end{aligned}, \tag{4.59}$$

where k_1 and k_2 are positive gains and u is set to be constant and positive. Therefore,

$$\dot{V}_1 = -k_1 s_1^2 - u k_2 y_1^2 \leq 0, \tag{4.60}$$

proving that the chosen virtual control law for θ will make the cross-track error converge to zero, as intended. Recalling that $\theta = \psi - \beta$,

$$\psi = \arcsin\left[\text{sat}\left(k_2 y_1 + \frac{V_c^{\perp}}{u}\right)\right] + \beta \ . \tag{4.61}$$

As a result, the L. Lapierre's path following method is now defined to give yaw references to the inner-loop, ψ_{ref}, while taking into account the ocean currents, assuming that it is possible to know the perpendicular and parallel components of the velocity of the current (through an observer).

It is important to consider that if $V_c^{\perp}$ is bigger than u, then the vehicle's speed won't be enough to compensate the current and stabilize the system. Moreover, when y_1 is big enough to saturate θ, the vehicle will be approaching the path perpendicularly, taking the values of $\pm 90^{\circ}$, which is a reasonable behavior for when the vehicle is too far away.

Considering the added offset $V_c^{\perp}/u$, due to the ocean current, the heading reference should align the vehicle's total velocity vector with the trajectory, achieving the intended goal.

Ocean Current Observer

Following the same reasoning as in [10] and looking at the filter theory contemplated in [11], it is possible to propose an observer to estimate the ocean current disturbance, assuming that a position system is available to provide measurements of the position of the vehicle, $\boldsymbol{p} = [x, y]^T$. Moreover, since the influence of the ocean currents is not a major topic of this thesis, it is also assumed that the vehicle's speed relatively to the fluid, u, is measured by a **DVL** (Doppler Velocity Log).

The structure of the observer, fixed in the world frame $\{I\}$, is based on the vehicle kinematics, ignoring the sway speed v,

$$\begin{cases} \dot{x} = u\cos(\psi) + v_{c_x} \\ \dot{y} = u\sin(\psi) + v_{c_y} \end{cases} , \tag{4.62}$$

where v_{c_x} and v_{c_y} denote the components of the ocean current disturbance in $\{I\}$.

Taking into account the measured quantities, the kinematic model and defining the notation $\hat{x}$ as an estimate of the inertial x and $\tilde{x} := x - \hat{x}$ as the error of the estimate (the same notation is used for y and v_{c_i}), an observer for the component v_{c_x} is

$$\begin{cases} \dot{\hat{x}} = u\cos\psi + \hat{v}_{c_x} + k_{x_1}\tilde{x} \\ \dot{\hat{v}}_{c_x} = k_{x_2}\tilde{x} \end{cases} . \tag{4.63}$$

Clearly, the estimate errors $\tilde{x}$ and $\tilde{v}_{c_x}$ are asymptotically exponentially stable if all the roots of the characteristic polynomial $p(s) = s^2 + k_{x_1}s + k_{x_2}$ associated with the system

$$\begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{v}}_{c_x} \end{bmatrix} = \begin{bmatrix} -k_{x_1} & 1 \\ -k_{x_2} & 0 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{v}_{c_x} \end{bmatrix} \tag{4.64}$$

have strictly negative real parts.

The observer for the component v_{c_y} can be written in an analogous manner, to yield

$$\begin{cases} \dot{\tilde{y}} = u \sin\psi + \hat{v}_{c_y} + k_{y_1}\tilde{y} \\ \dot{\hat{v}}_{c_y} = k_{y_2}\tilde{y} \end{cases} . \tag{4.65}$$

Likewise, the estimate errors $\tilde{y}$ and $\tilde{v}_{c_y}$ are asymptotically exponentially stable if all the roots of the characteristic polynomial $p(s) = s^2 + k_{y_1}s + k_{y_2}$ associated with the system

$$\begin{bmatrix} \dot{\tilde{y}} \\ \dot{\tilde{v}}_{c_y} \end{bmatrix} = \begin{bmatrix} -k_{y_1} & 1 \\ -k_{y_2} & 0 \end{bmatrix} \begin{bmatrix} \tilde{y} \\ \tilde{v}_{c_y} \end{bmatrix} \tag{4.66}$$

have strictly negative real parts.

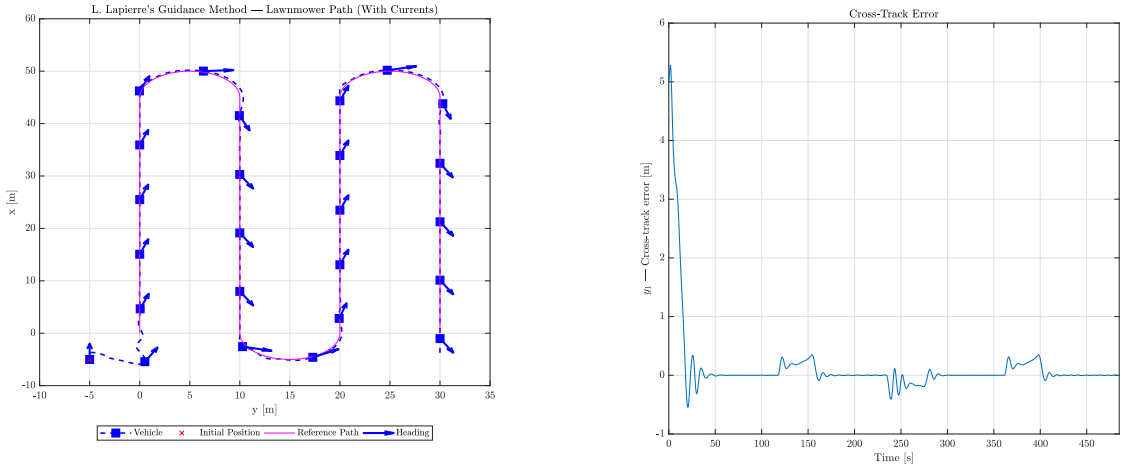
In order to obtain the perpendicular and parallel components of the ocean current disturbance relatively to the trajectory, one should recall the previously defined rotation matrix $R_I^T(\theta_c)$ to obtain these components in the Serret-Frenet frame:

$$\begin{bmatrix} V_c^{\parallel} \\ V_c^{\perp} \\ 0 \end{bmatrix} = R_I^T(\theta_c) \begin{bmatrix} v_{c_x} \\ v_{c_y} \\ 0 \end{bmatrix} . \tag{4.67}$$

As a result, it is now possible to estimate the ocean current disturbance, which is, in turn, used to compensate the heading so that the total velocity vector of the vehicle is aligned with the trajectory.

Results

Implementing the adapted L. Lapierre's path following controller, with current compensation and the ocean current observer, yields the following results, for an ocean current of $v_{cx} = -0.15$ m/s and $v_{cy} = -0.2$ m/s, and a desired vehicle speed of $u = 0.5$ m/s:



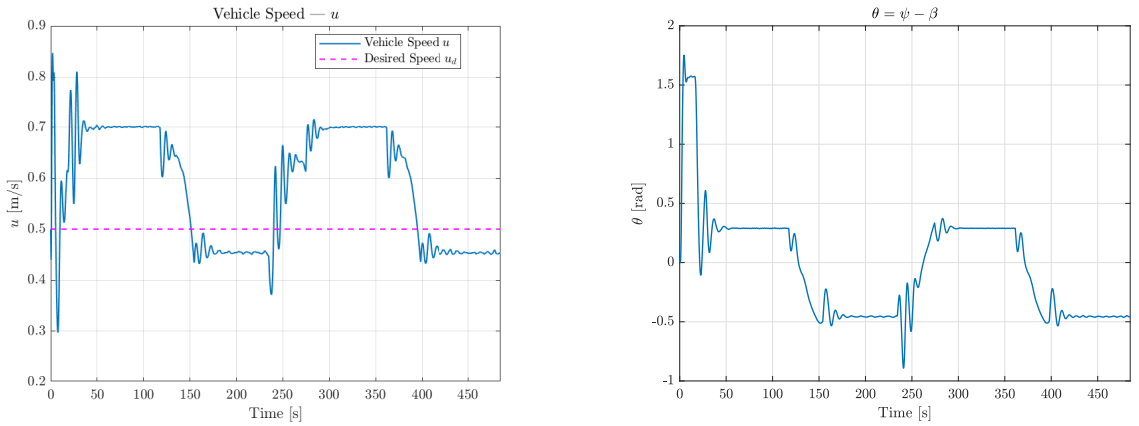
(a) L. Lapierre's path following of a lawnmower path wih currents.

(b) Cross-track error evolution.

Figure 4.22: Simulation of the L. Lapierre's guidance method for a lawnmower path scenario, with ocean currents: $v_{cx} = -0.15$ m/s and $v_{cy} = -0.2$ m/s.

Regarding the implementation of the new path following control law for ψ references, it is possible to observe in 4.22 that path following is, in fact, accomplished. However, using this new control law during the curved segments, a small offset is noticeable, captured by the evolution of the cross-track error. Nevertheless, current compensation works properly by orienting the heading angle in such a way that the perpendicular component of the current is canceled and the total velocity vector of the vehicle follows the trajectory, as intended.

In figure 4.23 it is possible to observe the evolution of the vehicle's speed and of $\theta = \psi - \beta$. This time, the vehicle's speed convergence is naturally affected by the ocean current and θ no longer converges to zero because the goal is not to align the vehicle's main body axis with the trajectory, but instead to align the vehicle's total velocity vector, which depends on where the vehicle must be heading to and the value of the parallel and perpendicular components of the ocean current.



(a) Vehicle's speed u (affected by the current).

(b) Evolution of θ (affected by the current)

Figure 4.23: Evolution of the vehicle speed and the value of θ during the simulation of a lawnmower path following scenario, with ocean currents: $v_{c_x} = -0.15$ m/s and $v_{c_y} = -0.2$ m/s.

Regarding the current observer, it is possible to assess in 4.24 that the observer is actually able to estimate the values of the constant ocean current disturbance.

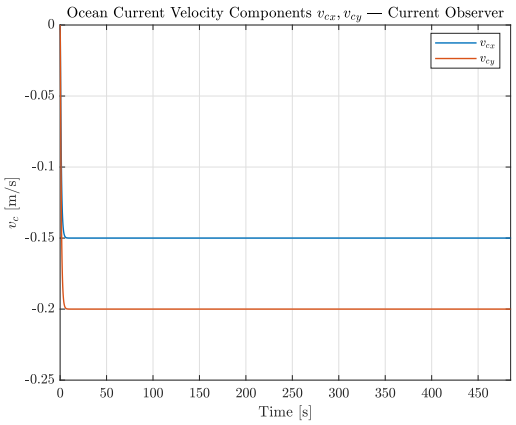


Figure 4.24: Ocean current velocity components v_{cx} and v_{cy}, given by the current observer.

Chapter 5

Multiple Vehicle Coordination Control

Now that single vehicle motion control is well studied, it is desired to achieve coordination between multiple vehicles. This chapter will describe a general architecture for multiple vehicle cooperative control, following a cooperative path following maneuver with simulation results. Some of the multiple vehicle coordination control issues are studied from references [3] and [12].

Beyond the issue of coordination, underwater communications between vehicles must be taken into account, which take place in a discrete manner. However, to start the coordination issue, continuous communications will be assumed and only after this will discrete communications take place. Since the vehicles will be using acoustic modems to broadcast their coordination states, data transference is bound to be slow, which is why one of the purposes of this thesis is to make faster optical communications viable in an underwater cooperative motion control scenario. As a result, a mechanism in which the vehicles only need to exchange data with their neighbors when necessary, in accordance with an appropriately defined criterion (time-dependent or state-dependent) will be proposed, in order to take into account the discrete communications and the slow data transference aspects. Ultimately, the underlying idea is that if any agent can produce good estimates of the neighboring states, then there is no need to communicate continuously among the vehicles. Therefore, each vehicle should be able to produce estimates of its own state and its neighbor's. This logic communications issue is studied from [13].

In order to begin studying the coordination issues, one should look first into some graph theory, so that it is possible to represent a communication scheme mathematically, referring to [14] for a comprehensive introduction to graph theory applied to the consensus problems, [15] for basic graph definitions, and [16] for algebraic graph theory.

5.1 Graph Theory

An undirected graph can be used in this case to model the communications network between the multiple underwater vehicles, where edges have no direction.

Definition 1 (Undirected Graph)**.** *A graph is a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that elements of $\mathcal{E}$ consist of two elements from $\mathcal{V}$, i.e. $\mathcal{E} \subseteq [\mathcal{V}]^2$. The elements of $\mathcal{V}$ are called vertices, or nodes of $\mathcal{G}$ and the elements $\mathcal{E}$*

are its edges, or links. One edge connects two vertices.

In the multiple vehicle coordination control scenario, the graph edges represent the communication link between a pair of vehicles, which are undirected, allowing for communications in both directions: from node n_0 to n_1 and from n_1 to n_0 (each node represents a vehicle), for example.

Additionally, $\mathcal{N}^{[i]}$ represents the set of neighboring nodes of node i with which this node communicates, among other basic definitions:

- Two nodes are adjacent, or neighbors, if there is an edge connecting them;
- The degree of a node is the number of edges associated with the node, that is equal to the number of neighbors, $\left|\mathcal{N}^{[i]}\right|$;
- A node with degree zero is isolated.

With this in mind, now it is important to represent the graph by means of matrices – algebraic graph theory. In this context, two main definitions are relevant: the adjacency matrix and the degree matrix.

The adjacency matrix of the graph, denoted A, is a square matrix with rows and columns indexed by the nodes such that the i, j entry of A is 1 if $j \in \mathcal{N}^{[i]}$ and zero otherwise – also formally defined as:

Definition 2 (Adjacency Matrix)**.** *The adjacency matrix $A = (a_{ij})_{N\times N}$ of $\mathcal{G}$ in which its elements are defined as*

$$a_{ij} := \begin{cases} 1 & \textit{if } n_i n_j \in \mathcal{E} \\ 0 & \textit{otherwise} \end{cases} . \tag{5.1}$$

The degree matrix D of the graph is a diagonal matrix where the i, j entry equals $\left|\mathcal{N}^{[i]}\right|$, the cardinality of $\mathcal{N}^{[i]}$, in short, it gives the number of neighbors to which the i-th node is connected in a diagonal matrix – also formally defined as:

Definition 3 (Degree Matrix)**.** *Let $\mathcal{G}$ be a graph. The degree matrix, $D = (d_{ij})_{N\times N}$ of $\mathcal{G}$ can have each element of the matrix defined as*

$$d_{ij} := \begin{cases} d\left(n_i\right) = \left|\mathcal{N}^{[i]}\right| & \textit{if } i = j \\ 0 & \textit{otherwise} \end{cases} . \tag{5.2}$$

From the two latter definitions, it is possible to define the Laplacian, L, of the undirected graph to represent it. The expression of the Laplacian of an undirected graph is defined as

$$L = D - A. \tag{5.3}$$

It is well known that if $\mathcal{G}$ is undirected, then L is symmetric and $L\mathbf{1} = \mathbf{0}$, where $\mathbf{1} := [1]_{N\times 1}$ and $\mathbf{0} := [0]_{N\times 1}$, with N being the total number of nodes.

One last modification can be made to the Laplacian form of a graph to obtain the normalized Laplacian

$$L_D = D^{-1}(D - A). \tag{5.4}$$

Example

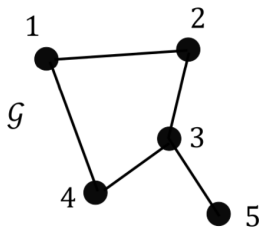


Figure 5.1: Undirected graph example (from [3]).

Considering the undirected graph represented in figure 5.1, the corresponding adjacency and degree matrices are given by

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.5}$$

With these matrices A and D, associated to this specific graph, one can easily compute the Laplacian, L, or the normalized Laplacian, L_D, using (5.3) and (5.4) respectively. Moving forward, the normalized Laplacian, L_D, will be useful to express the coordination error vector of a certain vehicle formation.

5.2 Coordination Control

In order to achieve cooperative path following, a group of **AUVs** is required to follow a series of pre-defined paths, while holding a certain formation pattern at a desired formation speed. For this reason, each vehicle has a path following algorithm defined as in section 4.4, which is needed to individually achieve the path following of its own pre-defined path. With this in mind, the only issue left is the one of coordination control, to ensure the formation pattern at a desired formation speed. The coordination problem statement, referring to Ghabcheloo et al. [17], can be formally defined as

Problem 1 (Coordination Problem)**.** *For vehicle $i = 1, \ldots, n$ derive a control law for $\dot{\gamma}^{[i]}$ as a function of local states and the variables $\gamma^{[j]}, j \in \mathcal{N}^{[i]}$ such that $\gamma^{[i]} - \gamma^{[j]}, \forall i, j$ approach a small neighborhood of zero as $t \to \infty$ and the formation travels at the speed $u_d(t)$, that is, $\dot{\gamma}^{[i]} \to u_d \forall i$.*

With the stated coordination problem, it is important to define the coordination state $\gamma^{[i]}$. In fact, this definition has been tackled before, in the path generation and parametrization subsection 4.4.2, where $\gamma^{[i]}$ corresponds to the normalized arc-length. The defined path parametrization, with the normalized arc-length as the coordination state, is appropriate to reach an in-line vehicle formation, where the vehicles follow their paths side-by-side (in a line) when their states are equal (coordination is being achieved).

The normalized arc-length coordination state is indeed very convenient, especially to properly achieve a radially aligned formation (in-line formation during a curve), where the goal is to follow a number of concentric circumferences (for each vehicle), due to the normalized length aspect: when the states are equal, radial alignment is reached regardless of the different radii and, thus, regardless of the different path lengths. In other words, for this concentric circumferences scenario and the established coordination state definition, if all vehicles have the same $\gamma^{[i]} = 0.5$, this means that all of the vehicles have already followed half a circumference, regardless of their own different radii, clearly being radially aligned whenever their states are equal.

In order to generate a different vehicle formation (for instance, a triangular formation), the desired target positions, $\boldsymbol{p_d}^{[i]}$, also defined in the path generation subsection 4.4.2, should be adapted for each vehicle, in order to offset intentionally the target positions as dictated by the desired formation.

Going back to the coordination problem, the distributed coordination controllers, designed to make $\gamma^{[i]}$ reach consensus, will yield a correction term, $u_c^{[i]}$, that is added to the desired speed of each vehicle, with the goal of driving the coordination error to zero. Thus, the desired speed that enters the inner-loop is

$$u_d^{[i]} = u_L^{[i]} + u_c^{[i]}. \tag{5.6}$$

This adjustment to the linear speeds of the vehicles allows the controllers to make the states reach consensus, by increasing or decreasing the speed of a certain vehicle, depending on whether it is falling behind or it is too upfront, based on a properly defined coordination error that can be used for control purposes.

With this in mind, it is possible to define the architecture represented in figure 5.2 for each vehicle.

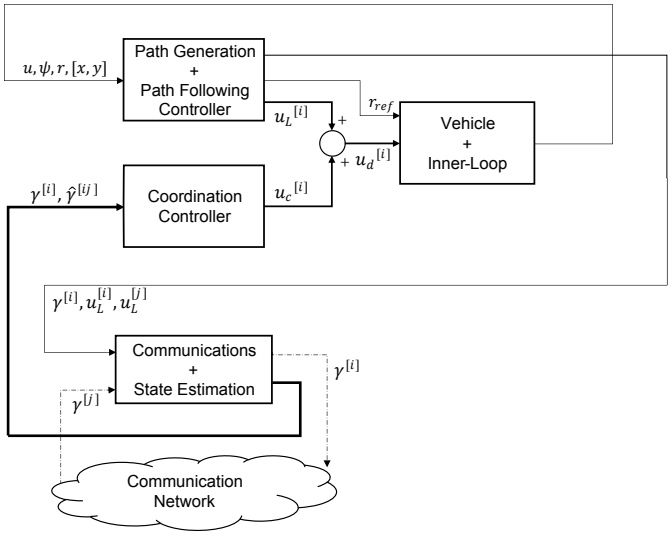


Figure 5.2: General control architecture considering coordination control.

Knowing that the communication network is modeled by an undirected graph, the previously defined normalized Laplacian, L_D, can be used to help define the coordination error vector:

$$\boldsymbol{\xi} = L_D \boldsymbol{\gamma}, \tag{5.7}$$

where $\boldsymbol{\xi}$ is the coordination error vector and $\boldsymbol{\gamma} = [\gamma^{[1]}, ..., \gamma^{[N]}]^T$ is the state vector containing the coordination state of each vehicle. This coordination error definition represents the difference between the coordination state of the vehicle itself and the mean coordination state of the vehicles with which it communicates. Each entry of $\boldsymbol{\xi}$ is, in fact, equal to

$$\xi^{[i]} = \gamma^{[i]} - \frac{1}{\left|\mathcal{N}^{[i]}\right|} \sum_{j \in \mathcal{N}^{[i]}} \gamma^{[j]}. \tag{5.8}$$

Assuming, for now, continuous communications, a distributed control law for $\boldsymbol{u_c}$ that drives ξ_i to zero is given by

$$\boldsymbol{u_c} = -k_\xi \tanh\left(L_D \boldsymbol{\gamma}\right), \tag{5.9}$$

where k_ξ is a positive constant and the hyperbolic tangent is used to bound $\boldsymbol{u_c}$, in order to prevent the speed reference from becoming negative. Proving that the origin of the coordination error vector is globally asymptotically stable is shown in the aforementioned literature and deemed not important to expose here.

As a result, it is possible now to use the defined distributed control law to reach consensus, thus achieving coordination. Moving forward, one should then take into account the discrete communications among vehicles.

5.3 Intuition Behind the ETC Mechanism

Before describing the Event-Triggered Communication (ETC) mechanism for the general consensus problem stated in the previous section, one can consider a simple application of the general setup to illustrate the underlying idea behind the proposed ETC mechanism.

This application involves the cooperative control of three autonomous underwater vehicles, and it is illustrated in figure 5.3.

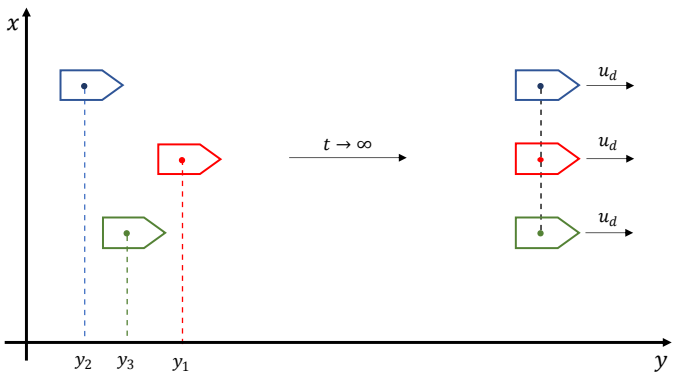


Figure 5.3: Cooperative control of three AUVs.

For simplicity of exposition, it is assumed that the vehicles only maneuver along the inertial axis y. Let $y_i \in \mathbb{R}$ and $u_d^{[i]} \in \mathbb{R}$ denote the coordinate and the speed of the vehicles along y, respectively, where $i \in \{1, 2, 3\}$. For notation consistency, $\gamma^{[i]} = y_i$ in this application. As a result, each vehicle can be considered as a point mass whose motion is described by a model given by

$$\dot{y}_i = u_d^{[i]}, \quad \forall i \in \{1, 2, 3\}. \tag{5.10}$$

The cooperative control problem involves finding $u_d^{[i]}$ to make the vehicles achieve coordination and travel with a common constant desired speed $u_L^{[i]}$, yielding a correction term denoted by $u_c^{[i]}$, as in

$$\begin{aligned} u_d^{[i]} &= u_L^{[i]} + u_c^{[i]} \\ \dot{y}_i &= u_L^{[i]} + u_c^{[i]}, \quad \forall i \in \{1, 2, 3\}. \end{aligned} \tag{5.11}$$

In this example, the AUVs exchange state information via acoustic communications, with the communication network being represented by an undirected graph as in figure 5.4.

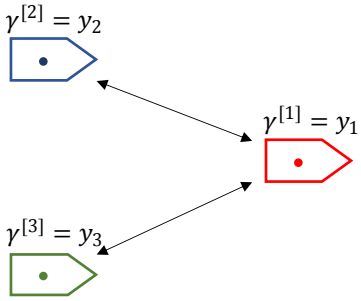


Figure 5.4: Graph representation of the communication network.

Recalling the distributed control law defined in (5.9), $u_c^{[i]}$ can be written explicitly for all vehicles as

$$\begin{aligned} u_c^{[1]} &= -k_\xi \tanh\left(y_1 - \frac{y_2}{2} - \frac{y_3}{2}\right) \\ u_c^{[2]} &= -k_\xi \tanh\left(y_2 - y_1\right) \\ u_c^{[3]} &= -k_\xi \tanh\left(y_3 - y_1\right). \end{aligned} \tag{5.12}$$

However, this control law implies that, in order to compute $u_c^{[i]}$, the vehicles must update the state of their neighbors continuously. The proposed ETC mechanism aims to prevent this costly requirement by triggering the state exchange only when necessary, while still guaranteeing coordination performance.

In the proposed ETC mechanism, the vehicles predict the states of their neighbors and use them in (5.12). This implies that the vehicles don't need to transmit their states to their neighbors continuously; instead, the transmission only happens when found necessary, so that their neighbors can correct the predictions. Let $\left\{t_k^{[i]}\right\}; k \in \mathbb{N}$ denote the sequences of time instants at which the vehicles transmit their states, the main goal of the ETC mechanism is to specify these sequences.

For simplicity of exposition, the ETC mechanism will be described with focus on the transmission of the state y_1, that is, how $\left\{t_k^{[1]}\right\}; k \in \mathbb{N}$ is specified for the vehicle 1. To this end, let $\hat{\gamma}^{[21]} = \hat{y}_1^{[2]}$ and $\hat{\gamma}^{[31]} = \hat{y}_1^{[3]}$ be the estimate of y_1 predicted by the vehicles 2 and 3, respectively. Neglecting communication

delays, to predict the state of vehicle 1, the neighbors 2 and 3 can adopt the following simple estimator:

$$\hat{y}_1^{[i]} := \begin{cases} \dot{\hat{y}}_1^{[i]} = u_L^{[1]} \\ \hat{y}_1^{[i]}\left(t_k^{[1]}\right) = y_1\left(t_k^{[1]}\right) \end{cases} \quad i = 2, 3 \tag{5.13}$$

This estimator can be interpreted as follows:

- The first equation of (5.13) is obtained from the fact that if all vehicles get coordinated, then all would travel with the same desired speed u_L.
- The second equation of (5.13) implies that the estimates of y_1 are corrected whenever vehicles 2 and 3 receive the latest state of vehicle 1.

The communication graph, considering the estimators, is represented in figure 5.5 for better comprehension.

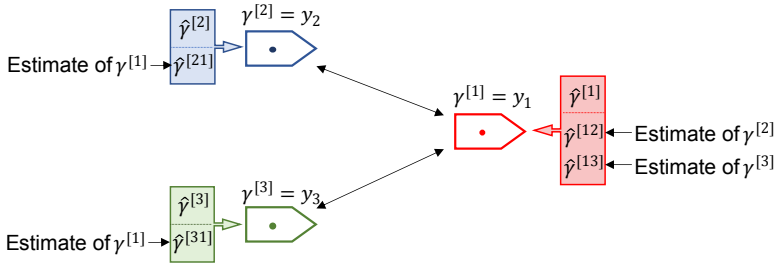


Figure 5.5: Adapted graph representation of the communication network.

With the neighbors 2 and 3 running estimators of y_1, their speed correction term is given by

$$\begin{aligned} u_c^{[2]} &= -k_\xi \tanh\left(y_2 - \hat{y}_1^{[2]}\right) \\ u_c^{[3]} &= -k_\xi \tanh\left(y_3 - \hat{y}_1^{[3]}\right). \end{aligned} \tag{5.14}$$

To facilitate the analysis, $u_c^{[2]}$ and $u_c^{[3]}$ can be rewritten as

$$\begin{aligned} u_c^{[2]} &= -k_\xi \tanh\left(y_2 - y_1 - e_1^{[2]}\right) \\ u_c^{[3]} &= -k_\xi \tanh\left(y_3 - y_1 - e_1^{[3]}\right), \end{aligned} \tag{5.15}$$

where

$$e_1^{[2]} := \hat{y}_1^{[2]} - y_1, \quad \text{and} \quad e_1^{[3]} := \hat{y}_1^{[3]} - y_1 \tag{5.16}$$

are the estimation errors of the state y_1 at vehicles 2 and 3, respectively. The underlying idea behind the ETC mechanism is to find a way to control these estimation errors, forcing them to be bounded in a "small" region in order to guarantee that the neighbors always have good predictions of the agent's state. In order to have triggered state broadcasts, vehicle 1 must send its state whenever these estimation errors exceed a certain threshold, and to accomplish this it must run itself an identical estimator, so that

it is able to evaluate the threshold. As a result, vehicle 1 runs a copy of $\hat{y}_1^{[i]}$ simply denoted $\hat{y}_1$, defined as

$$\hat{y}_1 := \begin{cases} \dot{\hat{y}}_1 = u_L^{[1]} \\ \hat{y}_1\left(t_k^{[1]}\right) = y_1\left(t_k^{[1]}\right) \end{cases} . \tag{5.17}$$

It follows that

$$\hat{y}_1(t) = \hat{y}_1^{[2]}(t) = \hat{y}_1^{[3]}(t) \quad \forall t, \tag{5.18}$$

implying that $\hat{y}_1$ can help monitor how good the estimations being performed by the neighbors are, thus letting the vehicle 1 broadcast its state whenever

$$|\hat{y}_1 - y_1| \geq \epsilon, \tag{5.19}$$

where $\epsilon \geq 0$ is called a triggering threshold. Therefore, the estimation errors $e_1^{[i]}$ are bounded by ϵ, as desired. Therefore, by choosing a proper ϵ, it is possible to control the transmission frequency of the state of vehicle 1. Formally, the sequence $\left\{t_k^{[1]}\right\}$, specified by the ETC mechanism, is given by

$$t_{k+1}^{[1]} = \inf\left\{t > t_k^{[1]} : |\hat{y}_1(t) - y_1(t)| \geq \epsilon\right\}. \tag{5.20}$$

The sequences $\left\{t_k^{[2]}\right\}$ and $\left\{t_k^{[3]}\right\}$ can be obtained similarly, after analogously defining the estimators for the remaining vehicles' states $\left\{\hat{y}_2^{[1]}, \hat{y}_2\right\}$ and $\left\{\hat{y}_3^{[1]}, \hat{y}_3\right\}$. This concludes the comprehensive explanation of the underlying idea behind the ETC mechanism, before formally generalizing.

5.4 Event-Triggered Communication Mechanism

Continuous communications among the vehicles are impossible to meet because the practical communication systems exchange data at discrete instants of time. Therefore, an Event-Triggered Communication (ETC) mechanism is proposed, in which the vehicles only need to exchange data with their neighbors when necessary, in accordance with a properly defined triggering function that contains all the local information that vehicle i has at a certain instant of time (referring to [12] and [13]).

In this mechanism, instead of using the true neighboring states, $\gamma^{[j]}; j \in \mathcal{N}^{[i]}$, the previously defined control law (5.9) uses their estimates – if any agent can produce good estimates of the neighboring states, then there is no need to communicate continuously. Letting $\hat{\gamma}^{[ij]}$ be an estimate of $\gamma^{[j]}$ computed by vehicle i and $\hat{\gamma}^{[i]}$ is an estimate of the vehicle's state itself, the event-triggered distributed control law is given by

$$u_c^{[i]} = -k_\xi \tanh\left(\hat{\gamma}^{[i]} - \frac{1}{|\mathcal{N}^{[i]}|}\sum_{j\in\mathcal{N}^{[i]}} \hat{\gamma}^{[ij]}\right); i \in \mathcal{N}. \tag{5.21}$$

Letting $\left\{t_k^{[i]}\right\}; k \in \mathbb{N}$ be the sequence of time instants at which vehicle i sends its current value of

$\gamma^{[i]}(t_k^{[i]})$ to its neighbors, during the interval $\mathcal{T}_k^{[i]} := \left[t_k^{[i]}, t_{k+1}^{[i]}\right)$ and for $t \in \mathcal{T}_k^{[i]}$, an estimator for $\hat{\gamma}^{[i]}$ can be defined as

$$\begin{aligned} \dot{\hat{\gamma}}^{|i]}(t) &= \bar{u}_L^{[i]} \\ \hat{\gamma}^{[i]}\left(t_k^{[i]}\right) &= \gamma^{[i]}\left(t_k^{[i]}\right), \end{aligned} \tag{5.22}$$

knowing that the state is the normalized arc-length, then $\bar{u}_L^{[i]}$ corresponds to the normalized desired speed given by the path generation block. The second equation implies that whenever vehicle i broadcasts $\gamma^{[i]}$ to its neighbors, the initial condition for $\hat{\gamma}^{[i]}$ will be reset.

Likewise, letting $\left\{t_k^{[i]}\right\}$ be the sequence of time instants at which vehicle j receives the state of vehicle i, the estimator for $\hat{\gamma}^{[ji]}; j \in \mathcal{N}^{[i]}$, during the interval $\mathcal{T}_k^{[i]} := \left[t_k^{[i]}, t_{k+1}^{[i]}\right)$ and for $t \in \mathcal{T}_k^{[i]}$, can be defined as

$$\begin{aligned} \dot{\hat{\gamma}}^{|ji]}(t) &= \bar{u}_L^{[ji]} \\ \hat{\gamma}^{[ji]}\left(t_k^{[ji]}\right) &= \gamma^{[i]}\left(t_k^{[i]}\right). \end{aligned} \tag{5.23}$$

Similarly, the second equation implies that whenever vehicle j receives the state of vehicle i, the initial condition for $\hat{\gamma}^{[ji]}$ will be reset.

As a result, each vehicle can estimate now its own state and the neighbor's states, which are needed to accomplish the ETC distributed control defined in (5.21). Moreover, appropriate reset of the estimates is defined, which happens after the triggering function satisfies a desired criterion.

Having defined the new distributed control law, which considers the estimates of the states, the next step is to define the triggering function that dictates when should there be a state broadcast, influencing when the estimates are reset. This triggering function will be defined for two types of events: time-dependent and state-dependent events.

5.4.1 Time-Dependent Triggering Events

In general, a triggering function can contain all the local information that agent i has at time t, that is,

$$h^{[i]}(t) = h^{[i]}\left(e^{[i]}(t), \gamma^{[i]}(t), \hat{\gamma}^{[i]}(t), \hat{\gamma}^{[ij]}(t), t; \quad j \in \mathcal{N}^{[i]}\right), \tag{5.24}$$

where $e^{[i]}(t) = \hat{\gamma}^{[i]}(t) - \gamma^{[i]}(t)$ is the local estimation error of vehicle i itself.

For a purely time-dependent triggering event, to ensure that the estimation error is bounded, vehicle i should transmit $\gamma^{[i]}$ whenever $e^{[i]}$ hits a designed threshold $\eta^{[i]}$ that is dependent on time. Mathematically, the triggering function for vehicle i is given by

$$h^{[i]}(t) = \left|e^{[i]}(t)\right| - \eta^{[i]}(t), \tag{5.25}$$

where $\eta^{[i]}(t)$ belongs to a class of non-negative functions $\mathcal{C}$ defined by $\mathcal{C} := \{f : R_{\geq 0} \to R_{\geq 0} \mid\leq\ 0 \leq f(t) \leq c_u\}$ for all $i \in \mathcal{N}$. For example, $\eta^{[i]}(t) = c_1 + c_2 e^{-\alpha t}$ with a proper choice of c_1, c_2 and α is a typical function belonging to $\mathcal{C}$. With this definition, vehicle i will send its state to its neighbors whenever $h^{[i]}(t) \geq 0$.

5.4.2 State-Dependent Triggering Events

Regarding state-dependent triggering events, it is possible to introduce another triggering function for each vehicle that depends on the information about its state estimate and the state estimates of the neighboring vehicles that communicate with it.

With the local information of the states, a new threshold is defined for $(e^{[i]})^2(t)$, resulting in the following triggering function:

$$h^{[i]}(t) = (e^{[i]})^2(t) - \left(\theta^{[i]} \frac{\sigma}{\sigma^{[i]}} \sum_{j \in \mathcal{N}^{[i]}} a_{ij} \left(\hat{\gamma}^{[i]}(t) - \hat{\gamma}^{[ij]}(t) \right)^2 + \epsilon_0 \right), \tag{5.26}$$

where a_{ij} represents the adjacency matrix A entries, which take the value of 1 or 0. The other constants: $\sigma, \sigma^{[i]}, \theta^{[i]}$, and ϵ_0 are defined in [13] and, for the sake of simplicity, were deemed not relevant to show here.

Likewise, with the above definition, vehicle i will send its state to its neighbors whenever $h^{[i]}(t) \geq 0$.

5.5 Results

Having applied the above definitions and the appropriate coordination control law, this section will show some simulation results of a group of underwater vehicles accomplishing coordination using continuous communications and discrete communications (both with time-dependent and state-dependent triggering events).

The simulations that were performed consider a group of three underwater vehicles in an in-line and triangular formations. The middle vehicle, vehicle 2, communicates with both neighbors 1 and 3. Vehicles 1 and 3 only communicate with 2. This communication graph is represented by the following Laplacian, adjacency, and degree matrices:

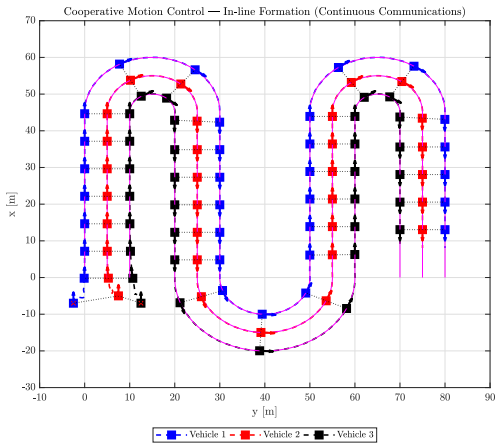
$$A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad L = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}. \tag{5.27}$$

As expected, the Laplacian matrix is symmetric and the sum of the elements of each line is zero – characteristic of an undirected graph representation.

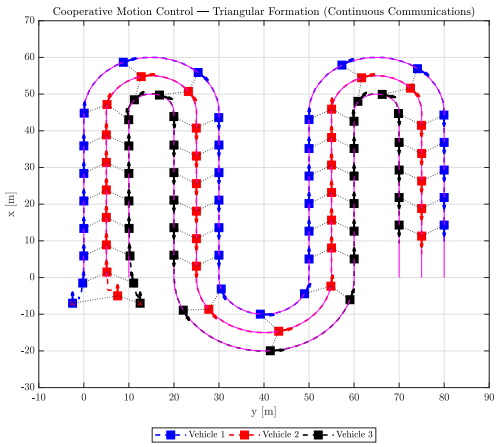
Continuous Communications

Figure 5.6 shows the cooperative motion control simulation for a series of lawnmower paths and the two different formations. It can be observed that coordination is properly reached among the vehicles with continuous communications, showing that the previously proposed distributed coordination control laws make the system achieve consensus.

Figure 5.7 shows the evolution of the state errors and the vehicles' speed. The state errors are defined as $\tilde{\gamma}_{ij} = \gamma^{[i]} - \gamma^{[j]}$, which must tend to zero due to the effect of the coordination controllers, as it



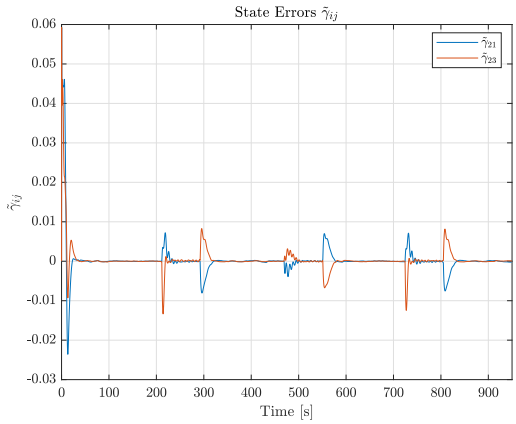
(a) Coordination of an in-line formation.



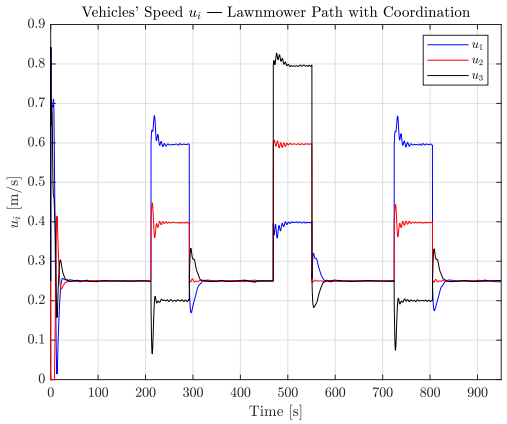
(b) Coordination of a triangular formation.

Figure 5.6: Cooperative motion control simulation of a group of three vehicles using continuous communications.

is shown. The vehicles' speed must be actively corrected in order to reach consensus and, in general, for concentric curves, vehicles that describe a curve with a bigger radius must travel at higher speeds to keep up with the rest of the agents, as it is shown.



(a) State errors $\tilde{\gamma}_{ij}$.



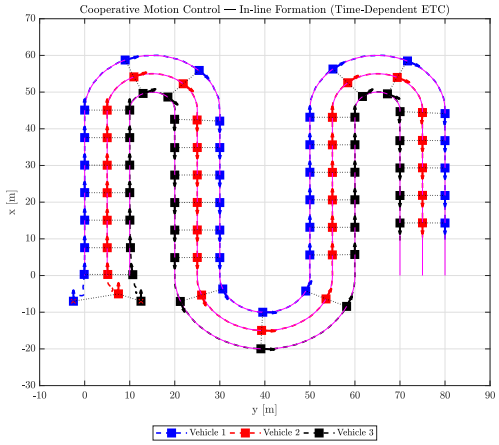
(b) Vehicles' speed u_i.

Figure 5.7: Evolution of the state errors and the surge speed of each vehicle, corrected by the coordination controller (in-line formation).

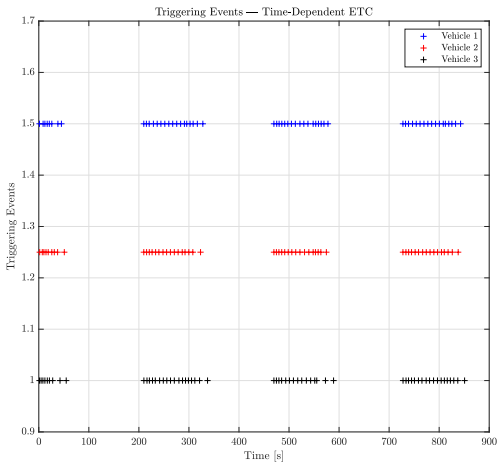
Time-Dependent Discrete Communications

Figure 5.8 shows that coordination is still achieved even when discrete communications are taking place. Moreover, it is possible to observe when each vehicle transmits its state, based on the time-dependent triggering function.

As it is shown, considering that each vehicle estimates the state of its neighbors, the agents don't need to be communicating continuously to achieve coordination, however, every now and then, they should reset their estimates in order to keep the errors from growing. Additionally, using a triggering



(a) Coordination of an in-line formation.



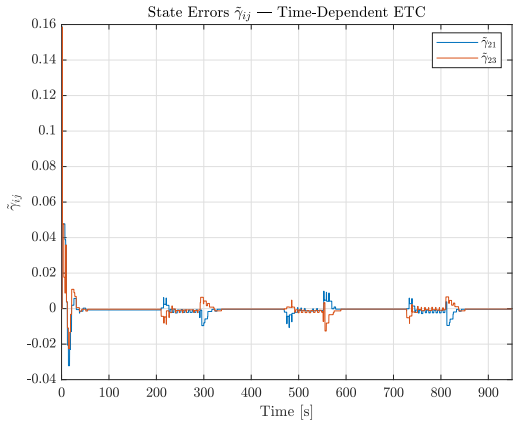
(b) Time-dependent triggering events over time.

Figure 5.8: Cooperative motion control simulation of a group of three vehicles using time-dependent ETC.

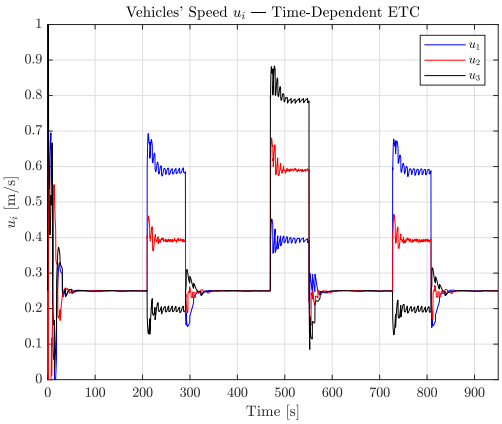
function avoids communication congestion when one considers that underwater acoustic communications have slow data transference.

Regarding time-dependent events, triggering happens throughout time without considering any more local information that a certain vehicle may have access to. Triggering is especially noticeable during the beginning of the simulation and while the vehicles describe a curve, both being situations when more estimation resets are needed. Observing figure 5.9, the states are still synchronized asymptotically, with the state errors tending to a neighborhood of zero. Corrections to the speed of the vehicles happen now at discrete instants of time, which is why the signals look less smooth than the previously obtained ones with continuous communications.

In general, it may be possible to observe some instances where coordination holds some errors for a certain period of time, causing vehicle formation misalignment during that period due to the discrete nature of the communications.



(a) State errors $\tilde{\gamma}_{ij}$.

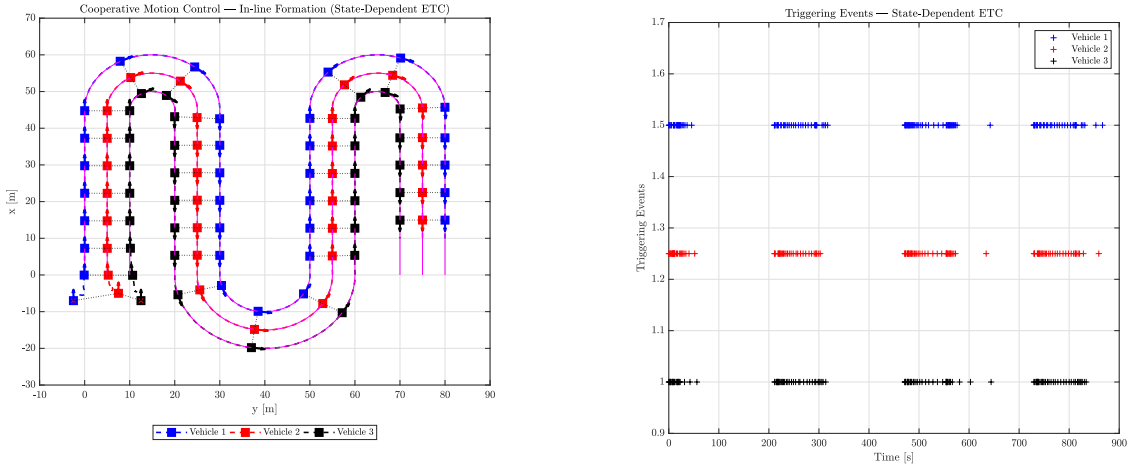


(b) Vehicles' speed u_i.

Figure 5.9: Evolution of the state errors and the surge speed of each vehicle, using time-dependent ETC.

State-Dependent Discrete Communications

Likewise, with state-dependent event triggered communications, coordination is also achieved, as it can be observed in figure 5.10. However, compared to the latter situation, state-dependent triggering happens more times during transitions between segments, instead of being more or less evenly spread out during a curve, which may be more desirable since transitioning between segments can be a critical maneuver during path following, requiring more speed corrections.

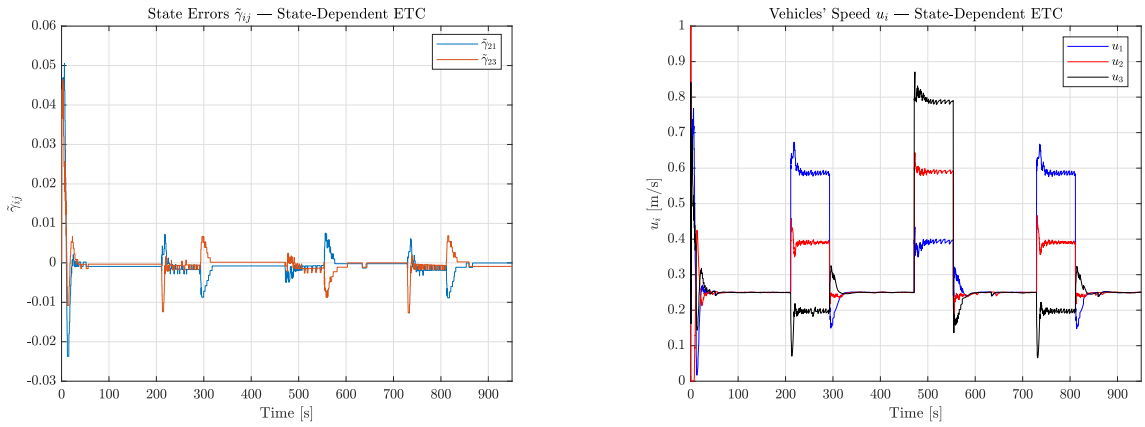


(a) Coordination of an in-line formation.

(b) State-dependent triggering events over time.

Figure 5.10: Cooperative motion control simulation of a group of three vehicles using state-dependent ETC.

In general, also considering the results in figure 5.11, it seems that better results are obtained using state-dependent triggering. This can be related to the fact that the triggering function, in this case, uses more local information about the agents it communicates with (the estimates of the neighbors), allowing for more triggering during critical maneuvering (such as path segment transitions). Similarly, the states are still synchronized asymptotically.



(a) State errors $\tilde{\gamma}_{ij}$.

(b) Vehicles' speed u_i.

Figure 5.11: Evolution of the state errors and the surge speed of each vehicle, using state-dependent ETC.

Chapter 6

Optical Communications

Thus far, single vehicle motion control has been studied, having designed an inner-loop heading controller and an outer-loop path following controller based on the kinematics. Additionally, cooperative motion control has been addressed considering that the vehicles use underwater acoustic communications to transmit their states among themselves, which happens at discrete instants of time.

Now to address more directly the point of this dissertation, the goal is to make a hybrid communication network viable during cooperative missions, where the vehicles still use acoustic communications to send their states, while optical communications are introduced for other purposes, such as transmitting mission data (maps, images, etc). Optical modems allow faster data transference, which is why they are more appropriate to transmit this kind of "heavier" data. The MEDUSA autonomous underwater vehicles and the BlueRay optical modem, to be mounted on these vehicles, are shown in figure 6.1.

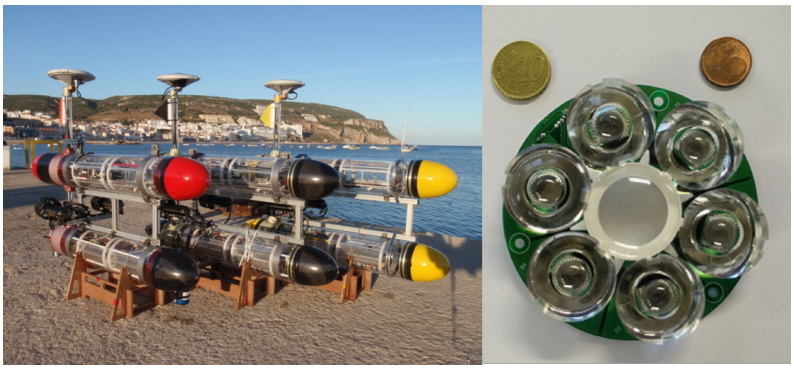


Figure 6.1: MEDUSA autonomous underwater vehicles and the BlueRay optical modem.

Nevertheless, optical communications have narrow directivity patterns, which make the issue challenging when they're added on moving agents, whose speeds are consistently being corrected in order to reach consensus. With this in mind, the major goal of this chapter is to devise a plan to facilitate optical beam alignment between a pair of vehicles. To accomplish this, each vehicle should be able to estimate the position of its neighbors along the path they are following, based on the information given by the coordination states. Knowing these positions, one can derive an algorithm that allows beam alignment between a pair of vehicles, assuming that the optical modems have independent rotation axes.

Beam alignment is planned to be accomplished in two phases: rough and refined alignment phases. Rough alignment aims to rotate the beams based on estimates of the positions of each vehicle, which may not lead to proper alignment due to the errors of the estimates. Refined alignment aims to apply small deviations to the rough alignment angle until the beams align, locking that beam orientation.

6.1 Rough Alignment Phase

The rough alignment phase aims to rotate the optical beams of a pair of vehicles assuming only that one vehicle knows its position and the position of the other one (along the path that is being followed).

Let $\boldsymbol{p_t} = [x_t, y_t]^T$ denote the position of the center of mass of the transmitting vehicle, with a_t denoting the central axis of the transmitting beam and α denoting the angle at which the transmitting beam is oriented. Likewise, let $\boldsymbol{p_r} = [x_r, y_r]^T$ denote the position of the center of mass of the receiving vehicle, with a_r denoting the central axis of the receiving beam, with orientation given by σ. Figure 6.2 represents the geometry of the beam alignment problem.

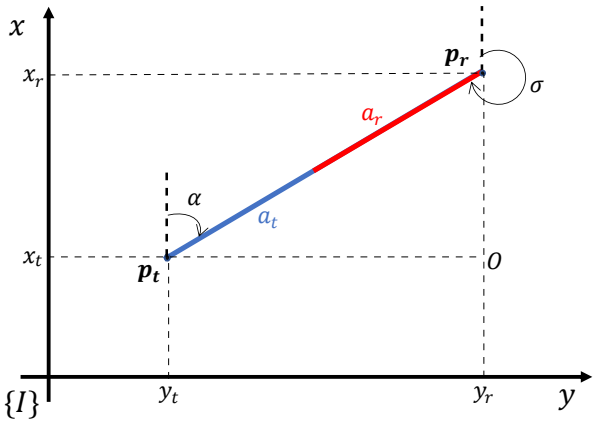


Figure 6.2: Geometric representation of the optical beam alignment problem.

For now, it is assumed that each vehicle knows exactly $\boldsymbol{p_t}$ and $\boldsymbol{p_r}$ (and not their estimates). With the goal of overlapping a_t and a_r, a geometric analysis of the situation represented in 6.2 yields the following equations for α:

$$\alpha = \begin{cases} \frac{\pi}{2} - \arcsin(k), & y_r \geq y_t \\ -\frac{\pi}{2} + \arcsin(k), & y_r < y_t \end{cases}, \tag{6.1}$$

where k is given by

$$k = \frac{\mathrm{d}(\boldsymbol{O}, \boldsymbol{p_r})}{\|\boldsymbol{p_r} - \boldsymbol{p_t}\|}. \tag{6.2}$$

The numerator $\mathrm{d}(\boldsymbol{O}, \boldsymbol{p_r})$ denotes the signed distance between $\boldsymbol{O}$ and $\boldsymbol{p_r}$. As a result,

$$k = \frac{x_r - x_t}{\sqrt{(y_r - y_t)^2 + (x_r - x_t)^2}}. \tag{6.3}$$

Similarly, the equations for σ are defined as

$$\sigma = \begin{cases} \frac{3\pi}{2} - \arcsin(k), & y_r \geq y_t \\ \frac{\pi}{2} + \arcsin(k), & y_r < y_t \end{cases} . \tag{6.4}$$

In reality, each vehicle has local information about its own coordination state and the states of the neighboring vehicles, which, for the sake of discrete communications, are estimated over time and are only reset after triggering occurs. With this in mind, the orientation of the beams will be affected by uncertainty, which motivates the design of a refined alignment phase. However, one still has to compute $\boldsymbol{p_t}^{[i]}$ and $\boldsymbol{p_r}^{[ij]}$ using $\gamma^{[i]}(t)$ and $\hat{\gamma}^{[ij]}(t)$, both for each vehicle of the pair, hence the superscripts $[i]$, $[ij]$.

Knowing each trajectory and the parametrization defined in equations (4.50) and (4.51), for a straight line segment, $\boldsymbol{p_t}^{[i]}$ can be given by

$$\boldsymbol{p_t}^{[i]}(\boldsymbol{t}) = \begin{bmatrix} x_0 + \gamma^{[i]}(t) l \sin(\theta_l) \\ y_0 + \gamma^{[i]}(t) l \cos(\theta_l) \end{bmatrix} , \tag{6.5}$$

and, for a circumferential segment of the path, by

$$\boldsymbol{p_t}^{[i]}(\boldsymbol{t}) = \begin{bmatrix} x_0 + R \sin(2\pi\gamma^{[i]}(t)) \\ y_0 + R\left[1 - \cos(2\pi\gamma^{[i]}(t))\right] \end{bmatrix} . \tag{6.6}$$

Assuming that agent i is the one transmitting and agent j the one receiving. Likewise, $\boldsymbol{p_r}^{[ij]}$ can be given by

$$\boldsymbol{p_r}^{[ij]}(\boldsymbol{t}) = \begin{bmatrix} x_0 + \hat{\gamma}^{[ij]}(t) l \sin(\theta_l) \\ y_0 + \hat{\gamma}^{[ij]}(t) l \cos(\theta_l) \end{bmatrix} , \tag{6.7}$$

for a straight line segment and by

$$\boldsymbol{p_r}^{[ij]}(\boldsymbol{t}) = \begin{bmatrix} x_0 + R \sin(2\pi\hat{\gamma}^{[ij]}(t)) \\ y_0 + R\left[1 - \cos(2\pi\hat{\gamma}^{[ij]}(t))\right] \end{bmatrix} , \tag{6.8}$$

for a circumferential segment of the path. Using these equations, each vehicle can estimate where the neighbor is and where the vehicle itself is, on top of their respective trajectories and in terms of inertial coordinates. As a result, α and σ can be finally computed for each vehicle, under the uncertainty afflicting the estimates $\hat{\gamma}^{[i]}(t)$ and $\hat{\gamma}^{[ij]}(t)$.

Nevertheless, this rough alignment phase, executed by determining α and σ, should always yield a rough optical beam alignment that needs only refinement, taking into account that using these state estimates actually yields good coordination results, meaning that they don't stray off too much from the real values. If one observes the coordination state errors in figures 5.9 and 5.11, these average around 10^{-4} of normalized path length.

6.2 Refined Alignment Phase

The refined alignment phase aims to improve the previously designed rough alignment. This is done by adding a correction term to the angles determined before.

A refinement is needed because, under uncertainty, one vehicle may have a wrong idea of the position of its neighbor during a period of time, requiring a small correction to the rough orientations. The problem becomes more challenging when one considers that cooperative missions require that vehicles travel from 2 to 5 meters apart, which means that, at these ranges, a small orientation offset at the origin of the beam causes a large beam misalignment at the receiver end (the orientation offset is amplified by how far the agents are from each other).

The correction term applied to α and σ will work as a sweeping mechanism: sweeping a small neighborhood of each roughly determined angle until the beams align, locking on the corrected orientation.

Power Variation with Beam Misalignment

One way of determining that the refinement phase was successful and that beam locking was achieved is by measuring the power of the acoustic signal received at each modem. The power of the signal is largely impacted by how misaligned are the optical beams (and it is also affected by attenuation and dispersion caused by the fluid). As a result, if the rough alignment phase causes a low received signal power, it means that the refinement phase must begin.

To this end, one should be able to study how does the power of the received signal vary with beam misalignment. Referring to [18], which studies the optical modems to be mounted on the vehicles, this variation follows a normal curve, as represented in figure 6.3.

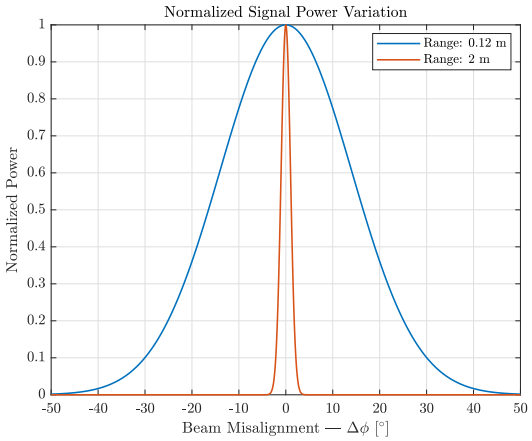


Figure 6.3: Variation of the received signal power with the beam misalignment for two different ranges.

It can be observed that for a communication range of 2 meters, a small misalignment impacts greatly the variation of the power of the received signal, in contrast to a much shorter range. Based on this curve, one can evaluate at a simulation level if beam alignment has been achieved. In a simulation environment it is possible to know the actual orientation of the beams that achieves proper alignment, by resorting to information that each vehicle, in reality, doesn't have access to. This way, it is possible to

evaluate the refinement results in a simulation environment (this information is only used for evaluation purposes).

Therefore, knowing the current orientation of the beams, given by the rough alignment phase, α and σ, and comparing these values with the actual ones, one can compute the misalignment:

$$\Delta\phi = \alpha_{real} - \alpha \quad \text{or} \quad \Delta\phi = \sigma_{real} - \sigma. \tag{6.9}$$

Consecutively, taking into account the known curve that models the variation of the signal power, one can compute the normalized power, having computed already $\Delta\phi$, and from this result infer whether or not the refinement reached its goal, by establishing a reasonable user defined threshold for the power of the received signal (for instance, more than 90% of the maximum average power).

Sweeping Mechanism

Now that it is possible to evaluate if the proposed goal is reached or not, the sweeping mechanism must be defined. As mentioned before, sweeping should occur on a small neighborhood of the previously computed values of α and σ. Letting α_c and σ_c denote the corrected angles, defined as

$$\begin{aligned} \alpha_c &= \alpha + A\sin(2\pi f_\alpha) \\ \sigma_c &= \sigma + A\sin(2\pi f_\sigma), \end{aligned} \tag{6.10}$$

where f_α and f_σ denote the oscillation frequencies of the correction term in hertz and A denotes the amplitude of the oscillations in degrees, one is capable of sweeping, using a sine wave with a certain frequency and amplitude as a sweeping term.

It may be problematic to use the same frequency for α_c and σ_c, depending on the phase of the oscillation, due to the fact that there may occur situations where the sweeping keeps the misalignment as the oscillations happen with the same frequency, not ever giving the sweeping mechanism a chance to make the beams overlap. Therefore, one should consider defining $f_\alpha = 2f_\sigma$, increasing the chances of beam alignment per period of the sweeping signal with the smallest frequency.

Additionally, the amplitude of the sweeping oscillations should start small (for instance, one degree) and be increased as needed. Considering that the system has no way of actually knowing how misaligned the beams are, increasing the amplitude as needed may be useful to overcome situations where sweepings of one degree are not enough. This requires the definition of a criterion that triggers the amplitude increase.

This criterion can simply take into account how long has it been since the refinement started and beam alignment was unsuccessful. To be more specific, if for every two periods of sweeping, beam alignment is not achieved, then the amplitude of the sweeping term should increase by a certain amount (for instance, three degrees).

With these two guidelines on the frequencies of the sweeping signals and their amplitudes, one should be able to avoid instances where there is no chance of ever making the optical beams overlap and where a certain amplitude is not enough to overcome how misaligned the beams are.

6.3 Discussion on Expected Results

If one implements both phases as explained before it is expected that optical beam alignment is reached between a pair of vehicles because, at some point in time, it will always be possible to make the beams overlap with the sweeping mechanism, as shown in figure 6.4.

However, the proposed algorithm assumes that the beams can be rotated instantaneously, which is a reasonable assumption considering that, in reality, the motion of the beams must be much faster than the vehicles', at least for a first approach on the subject.

Nevertheless, taking into account the results that were obtained in the coordination with discrete communications section 5.5, the vehicles' speeds have some fluctuation during discrete periods of time. This can raise another problem, one where the refined alignment phase can be constantly activated, which may not be desired at all times. One should keep this in mind and lean more to the conservative side when defining the acceptable threshold of the received signal power, for instance. However, if the estimates are good enough, the rough alignment phase should overcome some of these situations, which helps avoid sweeping at all occurrences of misalignment.

In order to determine if this proposed alignment algorithm is actually making optical communications viable, it was decided that it should be possible to notice an improvement on the average of the received signal power over time when the refinement phase aims to improve the angles obtained during the rough alignment phase. In other words, without refinement, the average received signal power should be lower because there will be periods of time where the estimate errors are such that beam alignment is not reached (only with the rough alignment phase activated). With this in mind, it should be possible to check whether or not the refinement mechanism actually improves the results.

6.4 Results

Implementing the proposed algorithm for optical beam alignment, with the two aforementioned phases (rough and refined alignment), yields the following results:

On a first scenario, the alignment algorithm was applied to a group of two vehicles with continuous communications that are following a straight line. However, the idea that one vehicle has of where the other is was intentionally adapted to force the situation where the agents think they are aligning the optical beams, but, in fact, they aren't and so the refinement phase must be activated and it must correct the orientation of the beams. This situation is represented in figure 6.4.

As can be observed, the beams are initially misaligned and then the refined alignment introduces the sweeping mechanism that ultimately makes the optical beams overlap. This was a specifically crafted situation, where vehicle 2 is intentionally offset by 50 centimeters along the path, in order to test the refinement phase. This refinement is better observed in figure 6.5, where the evolution of the corrected angles is represented.

It can be observed that the refinement is done by using sweeping oscillating terms that have different frequencies ($f_\alpha = 2f_\sigma$) and whose amplitudes starts small and gradually increases for every two periods

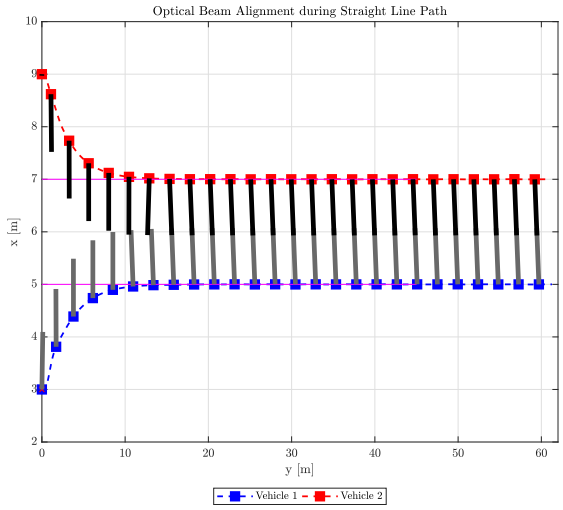
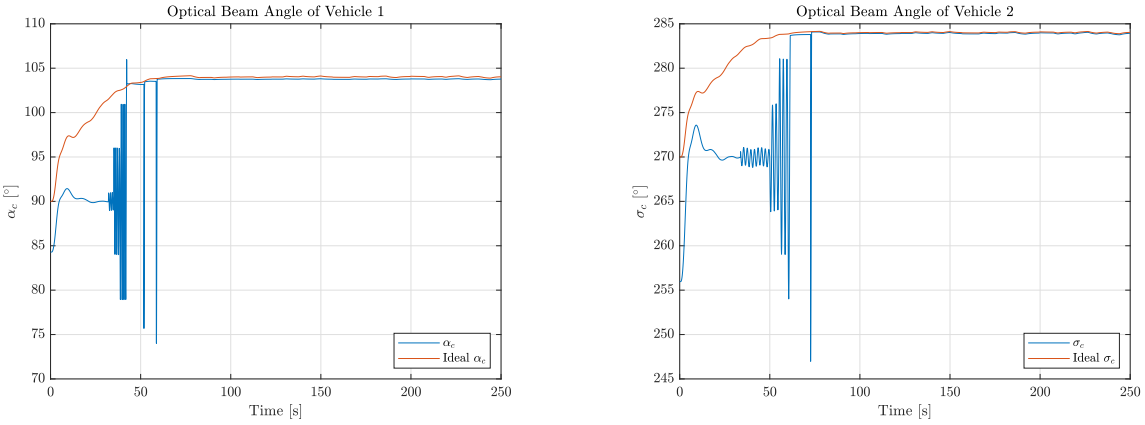


Figure 6.4: Optical beam alignment for a straight line path (with an intentional position offset).



(a) Evolution of α_c.

(b) Evolution of σ_c.

Figure 6.5: Evolution of the corrected angles α_c and σ_c.

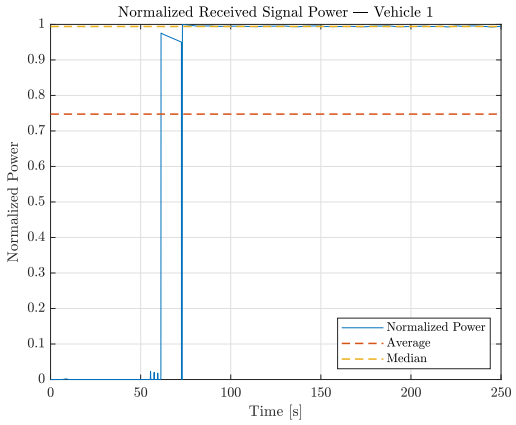
of unsuccessful alignment, as desired.

In terms of the normalized received signal power, one can observe figure 6.6. In fact, the refinement phase does make the beams align according to a user defined power threshold.

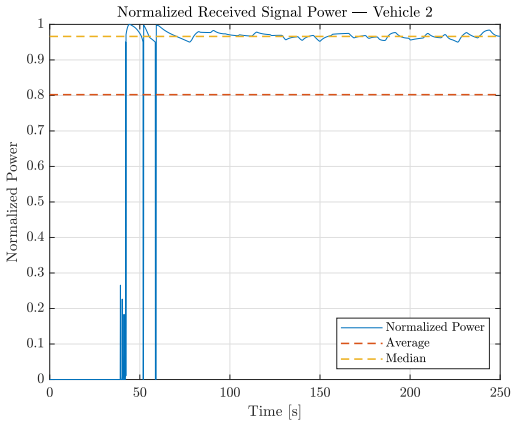
Nevertheless, this corresponds to a situation where the estimate errors are forced to be uncommonly large and, thus, the rough alignment phase is bound to fail by a large margin. This may represent the worst case scenarios during a regular cooperative mission. However, for this scenario, continuous communications were used to achieve consensus, which doesn't add the issue of the vehicles' speed fluctuation that one can find with discrete communications.

The second scenario to be simulated is one where there is a group of two vehicles in an in-line formation, following a lawnmower path with discrete communications (state-dependent ETC). The beam alignment result is represented in figure 6.7.

It can be observed that the proposed algorithm achieves optical beam alignment for a cooperative mission, using discrete acoustic communications to transmit the states among the vehicles. In fact, most

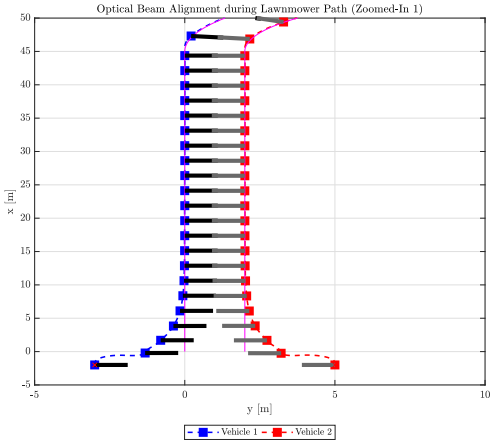


(a) Evolution of the received signal power of vehicle 1.

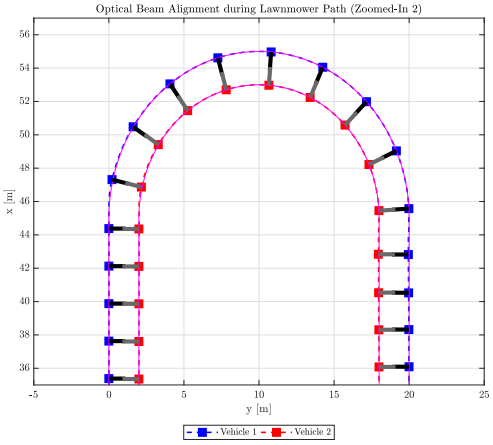


(b) Evolution of the received signal power of vehicle 2.

Figure 6.6: Evolution of the received signal powers.



(a) Zoomed-in portion of the path 1.



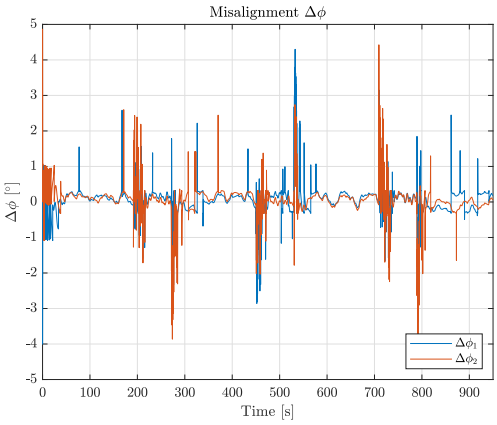
(b) Zoomed-in portion of the path 2.

Figure 6.7: Optical beam alignment for a lawnmower path zoomed-in portions (state-dependent ETC).

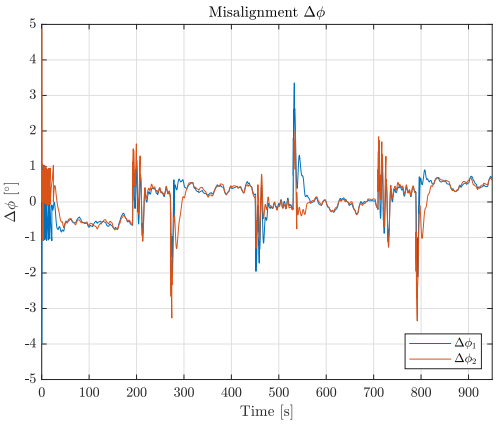
of the time, the rough alignment phase already produces a close enough orientation of the beams and only a small sweeping is required, as expected, observing the alignment errors in figure 6.8.

However, due to the range of communication (2 meters), the power curve is very narrow, as represented before in figure 6.3, which means that even a small deviation can produce a dramatic impact on the value of the received signal power, which is why, in figure 6.9, the normalized power shows sudden drops. Nevertheless, looking at the average and median values, most of the time, optical communications aim to be within the user defined power threshold.

Comparing the variation of the received signal power to a situation where only the rough alignment phase is activated, shown in figure 6.10, the average and median values show that the refinement phase is essential and works to improve the results, as expected (higher values are obtained with the refinement). Moreover, the refinement phase makes the alignment errors be more constrained to a neighborhood of zero, as shown in figure 6.8.

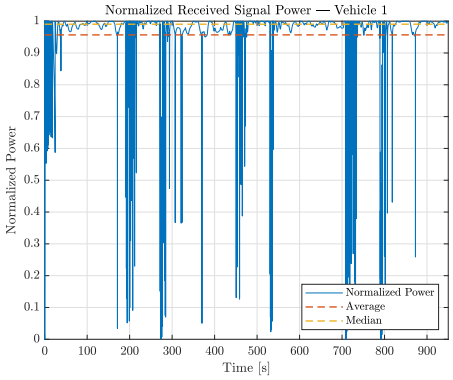


(a) Evolution of the misalignment $\Delta\phi$.

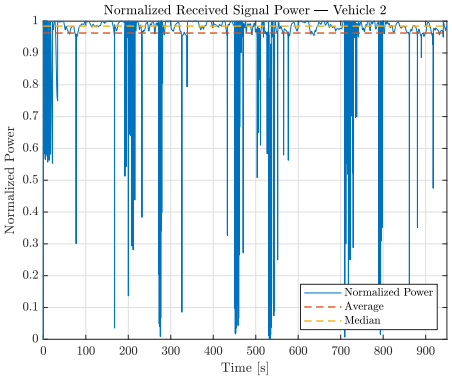


(b) Evolution of the misalignment $\Delta\phi$ without refinement.

Figure 6.8: Evolution of the misalignment $\Delta\phi$ with and without refinement.

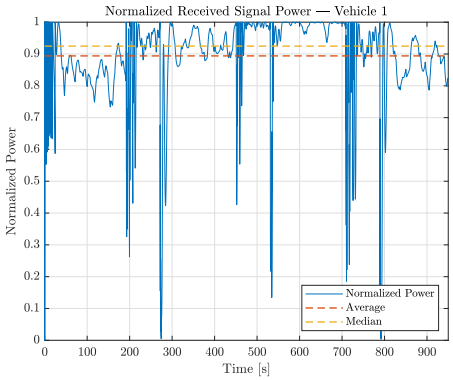


(a) Evolution of the received signal power of vehicle 1.

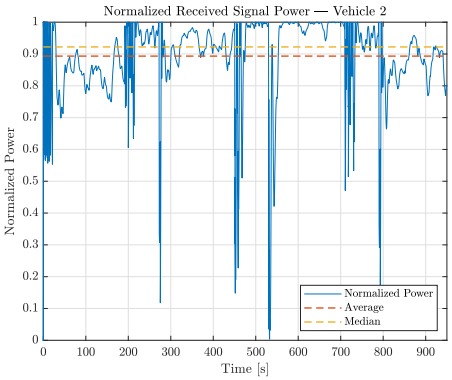


(b) Evolution of the received signal power of vehicle 2.

Figure 6.9: Evolution of the received signal powers (lawnmower path + discrete communications).



(a) Evolution of the received signal power of vehicle 1 (without refinement).



(b) Evolution of the received signal power of vehicle 2 (without refinement).

Figure 6.10: Evolution of the received signal powers without the refinement phase.

The next results correspond to the simulation of a group of three vehicles in a in-line formation that use optical communications in pairs, interchangeably.

The reasoning with more than two vehicles is to allow beam alignment between one pair of agents during some time and then, after some time, make a different pair of agents reach beam alignment. In

this case, every 5 minutes a different pair of vehicles is considered to achieve optical beam alignment. However, instead of using time to decide when should each pair communicate optically, one can define segments of the path where each pair should be using optical communications.

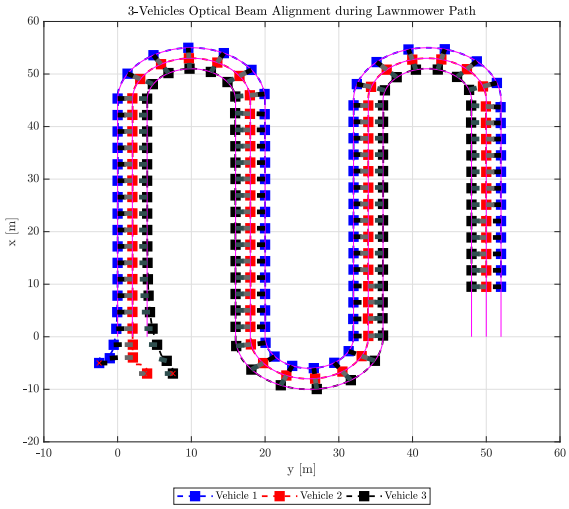
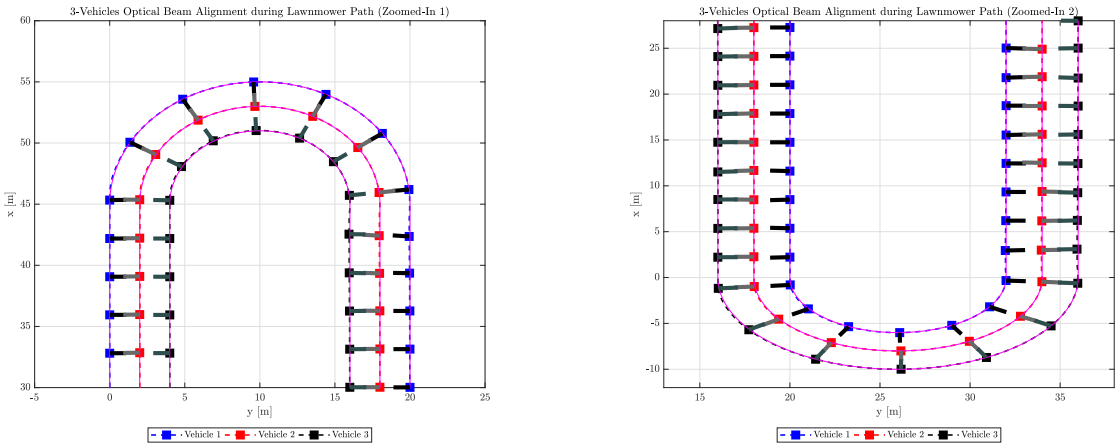


Figure 6.11: Optical beam alignment for a lawnmower path using three vehicles.



(a) Zoomed-in portion of the path 1.

(b) Zoomed-in portion of the path 2.

Figure 6.12: Optical beam alignment for a lawnmower path using three vehicles (zoomed-in portions).

It can be observed in figures 6.11 and 6.12 the instances when the alignment problem switches from one pair of vehicles (vehicles 1 and 2) to the other (vehicles 2 and 3). During interchangeable periods of 5 minutes, it is expected that the received signal power of vehicles 1 and 3 is zero. Moreover, the idle vehicle activates only the rough alignment phase, while the others are optically communicating.

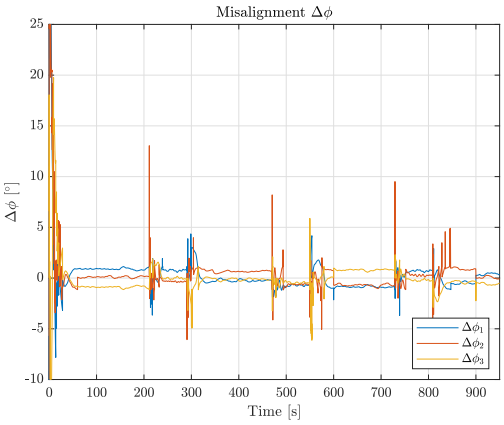
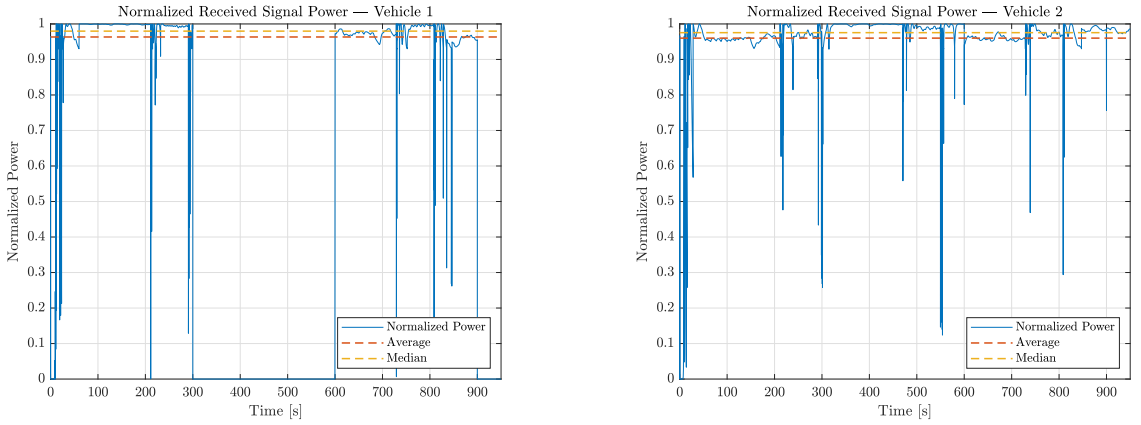


Figure 6.13: Evolution of the misalignment $\Delta\phi$.



(a) Evolution of the received signal power of vehicle 1.

(b) Evolution of the received signal power of vehicle 2.

Figure 6.14: Evolution of the received signal powers using three vehicles.

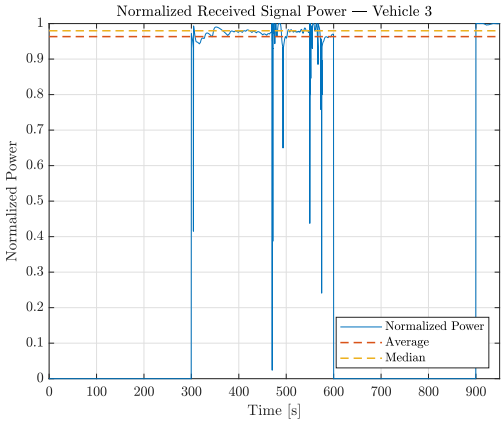


Figure 6.15: Evolution of the received signal power of vehicle 3.

Finally, the proposed algorithm has illustrated that using optical communications can be viable, which is the goal of this dissertation, however, it is very affected by small errors and, thus, there may be some intermittency.

It is important to take into account that some sweepings last longer in a simulation environment due

to the clock step that is used. If the clock step isn't small enough, then sweeping may occur continuously without ever reaching alignment, because the sweeping term depends on time, and if the time step is too large, then only a few points of each period of the sine wave are taken into account. Hopefully, in a real scenario, this process may work better, since, in practice, it should take advantage of its analogical nature.

Chapter 7

Conclusions

The purpose of this dissertation was to ultimately show that using a hybrid communication network, during cooperative missions, is viable. Acoustic communications are commonly used with underwater autonomous vehicles to broadcast the state of the vehicles. However, optical communication modems are just now breaking through in this field and so the problems regarding their narrow directivity patterns boil down to reaching optical beam alignment between a pair of vehicles, which has been shown to be viable.

Nevertheless, due to the complexity of this optical beam alignment problem using moving cooperative agents, some communication intermittency is to be expected, still the proposed algorithm may just be enough to transmit mission data (such as maps and images) among the vehicles using the faster optical communications pillar of the hybrid network.

At the first stage, the MEDUSA-class vehicles were modeled to allow the design of an inner-loop state feedback heading controller. Next, single vehicle motion control was accomplished by studying a variety of path following algorithms, based on the kinematics of the problem. Path following was ultimately accomplished using the L. Lapierre's method, which uses the Lyapunov's direct method for control design purposes of the outer-loop path following controller. It is an interesting approach, because one can come up with the virtual control laws that make the Lyapunov function decrease, proving stability as well. Finally, cooperative motion control was accomplished by designing a coordination controller that drives a certain coordination error to zero, reaching state consensus.

When matters of cooperative motion control are studied, communication among the vehicles plays a major role, namely the fact that the communication scheme needs a mathematical representation (graph) and that, in reality, communications among the vehicles take place at discrete instants of time. For this reason, the concept of triggering functions for discrete communications was studied.

To study and implement all of these tools was essential to ultimately address the optical communications issue. This problem was solved by resorting to two stages: a rough beam alignment phase followed by a refined one, which basically boiled down to implementing a sweeping mechanism until the optical beams overlapped.

As for future work, it would be important to test the beam alignment algorithm in real vehicles per-

forming real cooperative missions, recalling that ocean currents have to be considered as studied in this dissertation as well. Moreover, it could be interesting to study the impact of the system that rotates the optical beams, as it was disregarded in this dissertation. Lastly, if this optical beam alignment algorithm shows promising results in real tests, then one could consider using the optical communications to transmit the vehicles' coordination states as well.

Bibliography

[1] P. Maurya, A. P. Aguiar, and A. M. Pascoal. Marine vehicle path following using inner-outer loop control. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 42(18):38–43, 2009.

[2] A. Micaelli and C. Samson. Trajectory tracking for unicycle-type and two-steering-wheels mobile robots. *Rapports de recherche - INRIA*, (RR-2097), 1993.

[3] G. M. V. Sanches. Sensor-based formation control of autonomous marine robots. Master's thesis, Instituto Superior Técnico, 2015.

[4] J. Ribeiro. Motion control of single and multiple autonomous marine vehicles. Master's thesis, Instituto Superior Técnico, 2011.

[5] B. Anderson and J. Moore. *Optimal control: linear quadratic methods*. Prentice-Hall information and system sciences series. Prentice Hall, 1989.

[6] R. Horn and C. Johnson. *Matrix Analysis*. Matrix Analysis. Cambridge University Press, 2013.

[7] I. Kaminer, A. M. Pascoal, P. P. Khargonekar, and E. E. Coleman. A velocity algorithm for the implementation of gain-scheduled controllers. *Automatica*, 31(8):1185–1191, 1995.

[8] L. Lapierre, D. Soetanto, and A. Pascoal. Nonsingular path following control of a unicycle in the presence of parametric modelling uncertainties. *International Journal of Robust and Nonlinear Control*, 16(10):485–503, 2006.

[9] J. Stewart. *Calculus: Early Transcendentals*. Textbooks Available with Cengage Youbook. Cengage Learning, 2010. ISBN 9780538497909.

[10] A. P. Aguiar and A. M. Pascoal. Dynamic positioning and way-point tracking of underactuated AUVs in the presence of ocean currents. *International Journal of Control*, 80(7):1092–1108, July 2007.

[11] A. Pascoal, I. Kaminer, and P. Oliveira. Navigation system design using time-varying complementary filters. *IEEE Transactions on Aerospace and Electronic Systems*, 36(4):1099–1114, 2000.

[12] N. T. Hung, A. M. Pascoal, and T. A. Johansen. Cooperative path following of constrained autonomous vehicles with model predictive control and event-triggered communications. *International Journal of Robust and Nonlinear Control*, 30(7):2644–2670, 2020.

[13] N. T. Hung, F. C. Rego, and A. M. Pascoal. Event-triggered communications for the synchronization of nonlinear multi agent systems on weight-balanced digraphs. In *2019 18th European Control Conference (ECC)*, pages 2713–2718, 2019.

[14] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[15] R. Diestel. *Graph Theory*. Electronic library of mathematics. Springer, 2006. ISBN 9783540261834.

[16] C. D. Godsil. *Algebraic graph theory*. Springer, New York, 2001. ISBN 978-1-4613-0163-9.

[17] R. Ghabcheloo, A. P. Aguiar, A. Pascoal, C. Silvestre, I. Kaminer, and J. Hespanha. Coordinated path-following control of multiple underactuated autonomous vehicles in the presence of communication failures. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 4345–4350, 2006.

[18] E. A. Herji. Desenho e estudo de um sistema de comunicação ótica subaquático. Master's thesis, Instituto Superior Técnico, 2018.