Sverre Fjæra

Optimal and Adaptive Path Planning and Following for Permanent Resident Cleaning Robot Operating in Fish Farms

Master's thesis in Cybernetics and Robotics
Supervisor: Associate Professor Martin Føre
Co-supervisor: Dr. Sveinung Johan Ohrem, Dr. Eleni Kelasidi

May 2021

Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Sverre Fjæra

Optimal and Adaptive Path Planning and Following for Permanent Resident Cleaning Robot Operating in Fish Farms

Master's thesis in Cybernetics and Robotics
Supervisor: Associate Professor Martin Føre
Co-supervisor: Dr. Sveinung Johan Ohrem, Dr. Eleni Kelasidi
May 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Abstract

This thesis presents the results obtained from mathematical simulations of an autonomous underwater net cleaning robot to be used in an aquaculture environment to prevent one of the main challenges in the fish farming industry; biofouling on the fish cage structure. Furthermore, the report focuses on solving the motion planning problem with the use of the Elastic Band Method (EBM). The EBM was developed in MATLAB and implemented in FhSim for simulations with Mithal's state of the art biofouling prevention robotic concept Remora, and with a conventional free-swimming Remotely Operated Vehicle (ROV) for comparison with more traditional cleaning solutions to demonstrate the applicability of the EBM in different underwater robotic systems. A series of stepwise simulation experiments for testing the method are presented, starting with simulations to verify basic elements before more complex operations resembling actual autonomous biofouling prevention and cleaning operation was conducted:

- The robot's initial path is chosen in advance of the operation before the EBM adapts the path in each time-step to avoid the added static and dynamic regions of avoidance and finds a new feasible alternative path.
- Ocean current was introduced, and thus deforming the net structure of the fish cage. The planner's ability to adapt the path around the regions of avoidance while compensating for the environmental disturbances was evaluated.
- Lastly, the disturbances were combined with a realistic net-cleaning procedure with added complexity in the vehicle needing to return to a docking station (provided as a fixed point) to recharge during the operation before continuing where it left off.

Through the various experiments, the time complexity of the method showed promising results, implying use in a real-time application. The tuning possibilities of the EBM made obstacle avoidance possible with both static and dynamic regions of avoidance, with ocean current deforming the net. The same was obtained during the simulations of the ROV, indicating that the method can be used in several underwater aquaculture applications on multiple vehicle types. As the results showed, the method might be suitable for use on a specialised vehicle designed to crawl on the net, as well as a conventional ROV, in a dynamic aquaculture environment. The method should be and are planned to be tested in a real-life situation in a simulated environment aimed to resemble a fish cage in a lab trial after this thesis to verify the results obtained from the numerical simulations.

The results of this thesis have been submitted as an abstract to OCEANS 2021 and are attached in Appendix B. Experimental validation of the developed concepts in lab and field trials are planned and will target journal publication.

Samandrag

Denne oppgåva presenterar resultata frå matematiske simuleringar av ein autonom undervanns notvaskarrobot som skal brukast i eit havbruksmiljø for å forhindra ei av hovedutfordringane i oppdrettsnæringa; biobegroing på merdstrukturen. Vidare, fokuserar rapporten på å løyse bevegelse planleggingsproblemet ved å bruke Elastic Band Method (EBM). EBM blei utvikla i MATLAB og implementert i FhSim for simulering av Mithal sitt moderne robot konsept for forhindring av biobegroing, Remora, og med ein meir konvensjonell Remotely Operated Vehicle (ROV) for samanlikning av meir tradisjonelle reingjeringsmetodar, samt demonstrere anvendelegheita til EBM i forskjellige robotsystem under vann. Ein serie trinnvise simuleringseksperiment for testing av metoden er presentert, og startar med simuleringar for å verifisere grunnleggjande element før meir komplekse operasjonar som liknar faktisk autonome operasjonar for forhindring og reingjering av biobegroing har blitt gjennomført:

- Roboten sin opphavlege bane blir valt av ein operatør i forkant av operasjonen, deretter tilpassar EBM banen i kvart tidssteg for å unngå dei tilførte statiske og dynamiske hindringane og finner ein gjennomførleg alternativ bane.
- Havstraum blei introdusert og deformererte dermed merda sin nettstruktur. Planleggarens evne til å tilpasse banen rundt hindringane og kompensere for miljøforstyrringane blei evaluert.
- Til slutt blei forstyrringane kombinert i ein realistisk not-reingjeringsprosedyre med tilført kompleksitet i form av at roboten må returnere til dokkingstasjonen (gitt som eit fast punkt) for å lade batteriet under operasjonen, før den fortset der den slapp.

Gjennom dei forskjellige eksperimenta viste tidskompleksiteten til metoden lovande resultat, som tyder på at metoden kan brukast i sanntidsapplikasjonar. Moglegheita for å stille inn EBM til ønska oppførsel gjorde at roboten unngikk både statiske og dynamiske hindringar medan systemet var utsett for havstraum. Dei same resultata var observert etter simulering av ROV'en, noko som indikerer at metoden kan bli brukt i fleire applikasjonar under vann i eit havbruksmiljø. Som resultata viste, kan metoden være egna for bruk på eit spesiallaga køyretøy som er designa

for å krabbe på nettet, i tillegg til konvensjonelle ROV'ar, i eit dynamisk miljø. Metoden bør bli, og er planlagt, testa i ein verkeleg situasjon i eit simulert miljø som har til hensikt å likne ei oppdrettsmerd i eit laboratorieforsøk etter denne oppgåva for å verifisere resultata frå dei numeriske simuleringane.

Resultata frå denne oppgåva har blitt sendt inn som eit samandrag til OCEANS 2021 og er lagt ved i Appendix B. Eksperimentell validering av det utvikla konseptet i laboratorie- og feltforsøk er planlagd og er sikta mot ein tidskriftpublikasjon.

Preface

This thesis concludes my master's degree in Cybernetics and Robotics at the Norwegian University of Science and Technology (NTNU), under the Department of Engineering Cybernetics. The thesis has been carried out in collaboration with SINTEF Ocean as a part of the NetClean 24/7 project. Some of the work presented is based on my specialisation project conducted in the fall of 2020.

SINTEF Ocean has provided a personal computer with their framework for mathematical simulations in an aquaculture environment, FhSim, which has been used to obtain the results presented in this thesis.

First and foremost, I would like to thank Martin Føre for his extensive help and valuable meetings, ensuring progress and motivation throughout the semester. Many thanks to Eleni Kelasidi for the shared knowledge and for guiding my research in the right direction. Many thanks to Sveinung Johan Ohrem, who has done a fantastic job introducing FhSim and advanced control systems used in the simulations. An additional thanks to Herman B. Amundsen for help and guidance regarding the net pen following equations and some playtime with the BlueROV2. I would also like to express my gratitude to Bent Haugaløkken and Nina Blöcher for helpful advice on the report and for clarifying current biofouling control solutions and the consequences regarding biofouling.

Trondheim, 31.05.2021

Sverre Fjæra

Contents

Abstract . . . . **i**

Samandrag . . . . **iii**

Preface . . . . **v**

Contents . . . . **vii**

Figures . . . . **xi**

Tables . . . . **xv**

Acronyms . . . . **xvii**

1 Introduction **1**

1.1 Background 2

1.2 Motivation 3

1.3 Scope 4

1.4 Structure 4

2 Theory **5**

2.1 Mathematical Model of the Remora 5

2.2 Motion Planning and Control of Unmanned Underwater Vehicles . . 7

2.2.1 Path Planning vs Trajectory Planning 8

2.2.2 Path Following 8

2.2.3 Motion Planning Concepts 13

2.3 Elastic Band Method 18

2.3.1 The Phases of the EBM 18

2.3.2 The Steps of the EBM 24

2.3.3 Important Parameters and Pseudocode 25
2.3.4 Adaptation of the EBM to Cylindrical Coordinates 28
3 Implementation 31
3.1 Tools of Implementation 31
3.2 Algorithm Development 32
3.3 ROV Model 34
3.4 Simulation Setup 39
3.4.1 Main Program 41
3.4.2 Control System, Heading Guidance and Reference Models . 42
3.4.3 BlueROV2 43
3.5 Simulation Experiments 46
3.5.1 Parameterisation of the EBM 46
3.5.2 Case 1: Five-round Cleaning Routine under Ideal Conditions 47
3.5.3 Case 2: Static Regions of Avoidance 47
3.5.4 Case 3: Dynamic Region of Avoidance 48
3.5.5 Case 4: Ocean Current 48
3.5.6 Case 5: Ocean Current with Static and Dynamic Region of Avoidance 50
3.5.7 Case 6: Five-round Cleaning Routine under Non-Ideal Conditions 50
3.5.8 Time Complexity 51
4 Results 53
4.1 Simulations of the Remora 53
4.1.1 Parameterization of the EBM for the Remora 54
4.1.2 Case 1: Five-round Cleaning Routine under Ideal Conditions 56
4.1.3 Case 2: Static Regions of Avoidance 56
4.1.4 Case 3: Dynamic Region of Avoidance 58
4.1.5 Case 4: Ocean Current 58

4.1.6 Case 5: Ocean Current with Static and Dynamic Region of Avoidance. 61

4.1.7 Case 6: Five-round Cleaning Routine under Non-Ideal Conditions 63

4.1.8 Time Complexity 64

4.2 Simulations of the BlueROV2 66

4.2.1 Verifying the Tuning Parameters of the EBM 66

4.2.2 Case 1: Five-round Cleaning Routine under Ideal Conditions 66

4.2.3 Case 2: Static Region of Avoidance 67

4.2.4 Case 3: Dynamic Region of Avoidance 68

4.2.5 Case 4: Ocean Current 68

4.2.6 Case 6: Five-round Cleaning Routine under Non-Ideal Conditions 69

5 Discussion 71

5.1 Time Complexity 71

5.2 Elastic Band Method 72

5.3 Control System 73

5.4 Lab Trial 73

6 Conclusion and Future Work 75

6.1 Conclusion 75

6.2 Future Work 76

Bibliography 77

A Static Figures of Dynamic Illustrations 83

A.1 Implementation 84

A.1.1 Algorithm Development - 2D 84

A.1.2 Algorithm Development - 3D 85

A.2 Results 86

A.2.1 Remora - Case 3 86

A.2.2 Remora - Case 5 87

A.2.3 Remora - Case 6 88

A.2.4 BlueROV2 - Case 3 89

A.2.5 BlueROV2 - Case 5 90

B Abstract submitted to OCEANS 2021 91

Figures

1.1 Image of the Remora vehicle. 3

2.1 Cylindrical coordinates. 6

2.2 High-level system description. 8

2.3 LOS guidance. 10

2.4 System description with a detailed path following structure. 11

2.5 Possible solution of a hybrid motion planning concept. 14

2.6 Dubins path. 15

2.7 Elastic band bubble concept. 16

2.8 EBM - Internal force calculations. 19

2.9 EBM - External force calculations. 20

2.10 EBM - Net force calculations results in a new position. 21

2.11 EBM - Redundancy check. 21

2.12 EBM - Check for gap between bubbles. 22

2.13 EBM - Redundant and "missing" at the same time. 22

2.14 Spline principle. 23

2.15 Cylindrical representation to 2D workspace. 29

3.1 Connection between MATLAB and FhSim. 31

3.2 Animation of the EBM in 2D. 33

3.3 Animation of the EBM in 3D. 34

3.4 Image of the BlueROV2. 35

3.5 Physical layout of the thruster configuration on the BlueROV2. . . . 37

3.6 Screenshot of the Remora vehicle model in FhSim. 39

3.7 Two initial paths used for simulation. 40

3.8 Simplified state machine. 41

3.9 Screenshot of the BlueROV2 model in FhSim. 45

4.1 Screenshot from a simulation in FhSim. 53

4.2 The response when tuning the elastic band method. 55

4.3 The cross track error with different lookahead distance. 55

4.4 A five-round cleaning routine under ideal conditions. 56

4.5 The response from the final configuation. 57

4.6 The response in thruster commands. 57

4.7 Animation with static and dynamic region of avoidance. 58

4.8 Resulting path with ocean current without net deformation. 59

4.9 Graph of the control output from the FL- and PI-controller together with the actual surge speed. 59

4.10 Screenshot of a simulation in FhSim with net deformation. 60

4.11 Different paths with different configuration of ocean currents, causing varying cage radius. 60

4.12 Actual path with static region of avoidance and ocean current. . . . 61

4.13 Animation of the Remora with static regions of avoidance. 62

4.14 Animation of the Remora with dynamic region of avoidance. 62

4.15 A five-round cleaning routine with corresponding battery level and trigger point. 63

4.16 Animation of a five-round cleaning routine with battery level. 64

4.17 Time complexity of the EBM. 64

4.18 Time complexity of a realistic sized elastic band. 65

4.19 ROV tuning verification. 66

4.20 A five-round cleaning routine with the ROV. 67

4.21 The difference of deformation in axis. 67

4.22 Animation of the ROV with static and dynamic region of avoidance. 68

4.23 The response in surge and sway speed, due to the controllers adapting to the ocean forces. 69

4.24 A five-round cleaning routine under non-ideal conditions. 70

4.25 Animation of a five-round cleaning routine under non-ideal conditions with dynamic regions of avoidance. 70

A.1 Series of still figures of the animation in Figure 3.2. 84

A.2 Series of still figures of the animation in Figure 3.3. 85

A.3 Series of still figures of the animation in Figure 4.7. 86

A.4 Series of still figures of the animation in Figure 4.13. 87

A.5 Series of still figures of the animation in Figure 4.14. 87

A.6 Series of still figures of the animation in Figure 4.16. 88

A.7 Series of still figures of the animation in Figure 4.22. 89

A.8 Series of still figures of the animation in Figure 4.25. 90

Tables

2.1 Model parameters. 7

2.2 EBM parameters. 25

3.1 Position and orientation of thrusters on BlueROV2. 37

3.2 Controller parameters for Remora. 42

3.3 Controller parameters for BlueROV2. 44

3.4 Parameter configurations for the EBM. 47

3.5 Configuration parameters for ocean current. 49

Acronyms

AUV Autonomous Underwater Vehicle.

DOF Degrees Of Freedom.

DVL Doppler Velocity Log.

DVZ Deformable Virtual Zone.

EBM Elastic Band Method.

EOM Equations Of Motion.

LOS Line Of Sight.

LQR Liner Quadratic Regulator.

NLGL Nonlinear Guidance Law.

PLOS Pure Pursuit and Line Of Sight.

PP Pure Pursuit.

PRM Probabilistic Road Map.

PWM Pulsewidth Modulation.

RL Reinforcement Learning.

ROV Remotely Operated Vehicle.

RRT* Rapidy-Exploring Random Tree Star.

UGES Uniformly Globally Exponentially Stable.

VF Vector-Field.

Chapter 1

Introduction

This master's thesis is a part of a research project called NetClean 24/7, where SINTEF Ocean and NTNU are research partners. Mithal AS is the project owner, and together with Nordlaks Oppdrett AS and Xylem Aanderaa AS, NetClean 24/7 aims to make fish farming more sustainable and future-oriented by developing a state of the art net cleaning robot [1]. The amount of time and the costs associated with net cleaning today are significant as the current solutions require a considerable amount of manual labour. The net cleaning operations are carried out at different intervals, depending on the season, and it may be difficult for the fish farmers to acquire the net cleaning services during peak seasons. This may lead to extended periods of sub-optimal conditions for the fish and the structures. The NetClean 24/7 project aims to develop a permanently resident, autonomous robot that will continuously groom the nets and ensure that the biofouling colonies cannot establish.

This thesis presents solutions for autonomous motion planning during biofouling mitigation operations. The results presented are of high importance to developing autonomous functions in the robot, as no such functions exist in literature today for a net cleaning robot. Through a literature study and extensive simulations, a method that fulfils specific requirements related to aquaculture applications is proposed to solve the motion planning problem.

The specialisation project conducted in the autumn of 2020 completed a literature review with corresponding simulations of several motion planners for the same underwater net cleaning robot [2]. Therefore, some of the work is reused in this thesis and will be stated where necessary. This chapter is motivated by the specialisation project but elaborated further where needed.

1.1 Background

In 1971 the first farmed fish was produced in Norway, but the industry did not break through until the mid-1990s. Due to the nutritious fjords providing the right conditions, aquaculture has had an astonishing growth, leading to Norway becoming one of the primary producers of Atlantic salmon with over 3500 fish cages along the coast [3]. However, this growth has stagnated due to strict governmental regulations introduced as a measure to prevent the spread of salmon lice. In particular, a three-tiered system known as the traffic light system was introduced in 2020 [4]. Based on the number of salmon lice in the local area, the production capacity can either increase, decrease, or remain the same. Therefore, it is crucial to work towards a sustainable industry to increase further the amounts of food produced and improve fish welfare.

A typical fish cage used in Norway is a simple, robust and flexible construction ideal for rough conditions. They consist of a floating element at the top, a net in the middle, and a sinking element at the bottom to stretch the net, achieving maximum volume inside the cage. These structures are susceptible to the accumulation of biofouling which is a major problem and cost factor in the production [5]. Biofouling can be defined as the unwanted growth of organisms (e.g. hydroids, mussels, algae) at artificial substrates. Some of the challenges caused by biofouling are:

- Modified hydrodynamics in and around the cage affecting the water quality and the cage volume and stability.
- Increased disease risk due to biofoulers and associated pathogens.
- Reservoirs for non-indigenous species.

It was previously believed that biofouling had negative behavioural impacts on cleaner fish, which are used as a biological control agent against sea lice, as it was believed that the cleaner fish was grazing on the biofouling. It is, however, not scientifically proven that biofouling leads them to eat less salmon lice [6]. In fact, a study has found that biofouling has a positive effect on cleaner fish [7].

Biofouling control represents 5-10% of the production cost in the aquaculture industry [8]. The current biofouling mitigation solutions often make use of high-pressure cleaning robots that are partially automated. They still require substantial human involvement and are carried out at regular intervals, typically every 5-14 days depending on the season. This is both time-consuming and expensive, and the periodic routine leads to sub-optimal conditions both in advance, during, and after the cleaning process. In addition, the nets in a fish cage are often coated in special, antifouling coatings that contain copper oxide, cadmium and zinc [9]. High-pressure washers reduce the lifetime of this antifouling coating and release harmful substances into the environment. This, together with a large amount of biofouling released simultaneously, can cause a poor and stressful environment for the fish inside the cage and in the nearby cages [5].

Due to the increase in fish farming locations along the coast of Norway, the area of suitable and available space is limited, which has led to a gradual move towards exposed coastal locations [10]. Exposed locations are subject to more extreme environmental forces, such as wind and ocean currents, and in an already dangerous profession, health, safety, and environment become extra important. Findings in [10] points towards autonomous systems and technologies for remote operation as a part of the solution to increasing fish farming safety.

Keeping the level of biofouling to a minimum is essential when creating a stable and predictable living environment for the salmon. Increasing fish welfare, production efficiency and reducing the risk of wear, causing unwanted incidents at the site. Therefore it is expedient to investigate a new management solution for biofouling prevention where the objective is to have an automated robotic system operating in fish farms to solve the challenges discussed above.

1.2 Motivation

The NetClean 24/7 aims to address this challenge using a tetherless net cleaning robot called Remora and introduce autonomous continuous net cleaning. This method would clean less intense more often, reducing personnel needs and hence HSE risks during cleaning operations. Furthermore, the method will save costs and create a less variant environment for the fish. Besides maintaining the marine environment by reducing the abrasion of the antifouling coating on the net, it also prevents biofouling from accumulating over time, reducing the danger of incidents.

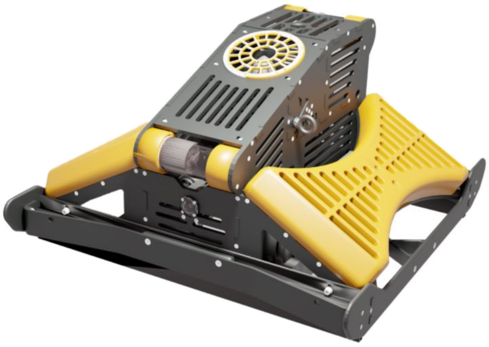


Figure 1.1: Image of the Remora vehicle. Image courtesy of Mithal AS, [11].

Figure 1.1 shows an image of the Remora. The vehicle has two belts and one thruster on the top to force it onto the net, such that it can traverse the net in any direction. A brush underneath the vehicle will irritate the biofouling organisms recurrently such that the biofouling levels are limited [1]. Autonomous continuous cleaning would not make as significant an impact with regards to disturbing the fish when cleaning, ensuring an invariable environment. By implementing a docking station, the robot can charge and transfer data autonomously at any time.

The Remora vehicle aims to be a fully automated robot working without human interaction while performing the cleaning operation. It has a limited power source, making motion planning highly critical for a successful net cleaning operation. The requirements needed for this application is that the robot needs to plan, adapt and follow a path to ensure the underneath brush covers the whole net.

1.3 Scope

The focus of this project is applying a path planning method on the Remora vehicle in an aquaculture environment. Through realistic simulations, the path planner is evaluated and later discussed in detail.

Previous work implemented a simple path planning algorithm and controllers under the assumption of ideal conditions [12]. This project aims to improve the performance of the path planner and extend the area of use. A literature study regarding path planners was conducted during the preceding specialisation project, where real-time possibilities and obstacle avoidance were highly valued criteria. The three most promising methods for this application were studied in greater detail with simple simulations. Together with the cellular decomposition method, the Elastic Band Method (EBM) showed promising results to cope with dynamic non-optimal conditions.

This thesis will focus on the implementation and simulation of the EBM used on a mathematical model of the Remora vehicle in a fish cage with environmental disturbances.

1.4 Structure

The report is structured as follows: Chapter 2 presents theory about motion planning, focusing on the EBM. Chapter 3 concerns the implementation and describes the chosen path planner and the simulation cases. Chapter 4 presents the results from the simulation cases, before Chapter 5 discuss the findings, while Chapter 6 concludes the thesis.

Throughout the report, animations of the motion planning algorithm illustrate the concepts of the method and the results obtained from simulation. These can be viewed in an applicable PDF-viewer, e.g. the Adobe Acrobat Reader. A corresponding series of static figures are also provided in Appendix A for those who cannot see the animations.

Chapter 2

Theory

Parts of this chapter is based on the preceding specialisation project [2]. More specifically, Section 2.2 are adapted from the specialisation project with some modifications to elaborate further. Section 2.1 and Section 2.3 supply new theory needed in this thesis. More details from the specialisation project can be found in [2].

This chapter focus on the motion planning part of the system. However, to understand what requirements needed in a motion planner, the following section introduces the mathematical model of the Remora robot to be used in a fish cage and its workspace.

2.1 Mathematical Model of the Remora

A mathematical model of the vehicle is crucial for realistic simulations and control of the vehicle. For the most realistic results from the numerical simulations, a precise model is desirable. Variables marked in **bold** is vectors and matrices, and this notation is used throughout the report.

A 6-DOF marine craft Equations Of Motion (EOM) can be defined in matrix-vector form as [13]

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{J}_\Theta(\boldsymbol{\eta})\boldsymbol{\nu} \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) + \boldsymbol{g}_0 &= \boldsymbol{\tau} + \boldsymbol{\tau}_{\text{wind}} + \boldsymbol{\tau}_{\text{wave}} \ , \end{aligned}$$

where $\boldsymbol{J}_\Theta(\boldsymbol{\eta})$ is the transformation matrix, $\boldsymbol{g}(\boldsymbol{\eta})$ is the restoring forces and moments, and $\boldsymbol{\tau}$, $\boldsymbol{\tau}_{\text{wind}}$ and $\boldsymbol{\tau}_{\text{wave}}$ is the propulsion, wind and wave forces respectively. $\boldsymbol{M}$, $\boldsymbol{C}(\boldsymbol{\nu})$ and $\boldsymbol{D}(\boldsymbol{\nu})$ is the system intertia, coriolis-centripital and damping matrix respectively. Furthermore are $\boldsymbol{\eta}$, $\dot{\boldsymbol{\eta}}$ and $\boldsymbol{\nu}$ defined from the notation of

SNAME as [14]

$$
\begin{aligned}
\boldsymbol{\eta} &= [x^n, y^n, z^n, \phi, \theta, \psi]^\top && (2.1)\\
\dot{\boldsymbol{\eta}} &= [\dot{x}^n, \dot{y}^n, \dot{z}^n, \dot{\phi}, \dot{\theta}, \dot{\psi}]^\top && (2.2)\\
\boldsymbol{\nu} &= [u, v, w, p, q, r]^\top . && (2.3)
\end{aligned}
$$

Note that (2.3) and ϕ, θ, ψ is defined in BODY frame, as it is advantegous when deriving the EOM [13]. The superscript ^{n} denotes variables represented in NED frame.

As the Remora has belts that hold it onto the net, it only has two degrees of freedom on the net and one attitude degree of freedom. The workspace of the vehicle is a cylindrical-shaped net. It is therefore intuitive and advantageous to convert the representation from cartesian to cylindrical coordinates, reducing the states to three states: $\boldsymbol{\eta} = [\alpha_v, D, \psi]^\top$, and $\boldsymbol{\nu} = [u, r]^\top$. α_v is the azimuth angle describing the position along the circumference of the cage, while D is the depth along the z^n-axis. ψ represents the heading of the vehicle on the net, showed in Figure 2.1. Furthermore, u is the speed in the x^b axis, and r is the angular velocity around the vehicles z-axis, z^b.

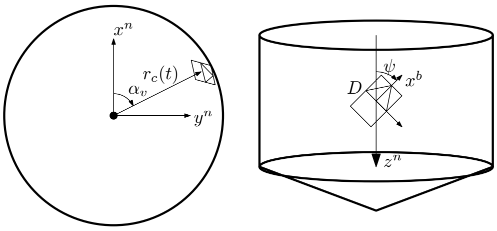


Figure 2.1: Cylindrical coordinates. Image based on Figure 2.7 in [12].

This coordinate change has been exploited in [15] and used to develop a precise mathematical model of the Remora vehicle with reasonable assumptions. The subscript $_r$ denotes the relative velocity in surge due to ocean currents, and $\dot{\boldsymbol{\nu}}_r$ is defined as $[u_r, r]^\top$. When solving for $\dot{\boldsymbol{\nu}}_r$, the resulting model becomes

$$
\begin{aligned}
\dot{\boldsymbol{\eta}} &= \boldsymbol{J}_c(\psi)\boldsymbol{\nu}\\
\dot{\boldsymbol{\nu}}_r &= \boldsymbol{M}_c^{-1}\left(-\boldsymbol{D}_{l,c}\,\boldsymbol{\nu}_r - \boldsymbol{D}_{n,c}\left(\boldsymbol{\nu}_r\right)\boldsymbol{\nu}_r + \boldsymbol{\tau}_c\right) ,
\end{aligned}
$$

where the transformation matrix between the body and the cylindrical frame is given as

$$
\boldsymbol{J}_c(\psi) = \begin{bmatrix} \sin(\psi)/r_c(t) & 0 \\ -\cos(\psi) & 0 \\ 0 & 1 \end{bmatrix} .
$$

The system intertia matrix $\boldsymbol{M}_c = \boldsymbol{M}_c^\top = \boldsymbol{M}_{RB,c} + \boldsymbol{M}_{A,c} \geq 0$ is defined as

$$
\boldsymbol{M}_c = \begin{bmatrix} m - X_{\dot{u}} & 0 \\ 0 & I - N_{\dot{r}} \end{bmatrix} .
$$

The linear ($\boldsymbol{D}_{l,c}$) and nonlinear ($\boldsymbol{D}_{n,c}(\boldsymbol{\nu}_r)$) damping matrices are

$$\boldsymbol{D}_{l,c} = \begin{bmatrix} X_u + X_{|u|u}|u| & 0 \\ 0 & N_r + N_{|r|r}|r| \end{bmatrix}$$

$$\boldsymbol{D}_{n,c}(\boldsymbol{\nu}_r) = \begin{bmatrix} X_{|u|u}|u_r| & 0 \\ 0 & N_{|r|r}|r| \end{bmatrix}.$$

Furthermore, the force and moment vector ($\boldsymbol{\tau}_c$) is given by

$$\boldsymbol{\tau}_c = \boldsymbol{T} * \boldsymbol{\tau} = \begin{bmatrix} 1 & 1 \\ l & -l \end{bmatrix} \begin{bmatrix} \tau_{\text{right}} \\ \tau_{\text{left}} \end{bmatrix} = \begin{bmatrix} \tau_u \\ \tau_r \end{bmatrix},$$

where $\boldsymbol{T}$ is the configuration matrix that maps the torque from the belts to the body-fixed forces and moments τ_u and τ_r. All parameters are described in Table 2.1.

Table 2.1: Model parameters.

Name	Description
$r_c(t)$	Cage radius, can vary due to environmental forces
m	Mass of vehicle
$X_{\dot{u}}$	Added mass force along the x-axis due to an acceleration $\dot{u}$
I	Vehicle inertia
$N_{\dot{r}}$	Added mass force around the z-axis due to an acceleration $\dot{r}$
X_u	Linear damping in surge
$X_{\|u\|u}$	Nonlinear damping in surge
$\|u\|$	Absolute velocity in surge direction
$\|u_r\|$	Absolute relative velocity in surge direction
N_r	Linear damping in yaw rate
$N_{\|r\|r}$	Nonlinear damping
$\|r\|$	Absolute angular velocity

In this section, we have presented the mathematical model briefly since it will be used for the simulations conducted in Chapter 4. The complete derivation of the model of the cleaning robot can be found in [15].

2.2 Motion Planning and Control of Unmanned Underwater Vehicles

Motion planning intends to produce a feasible desired path and orientation that the path following system can use to control the Remora vehicle towards the desired path and attitude. Figure 2.2 shows the high-level system description and how the different parts of the system connect. Feedback connection from the Remora to the path following system makes it possible to control the vehicle along the path. With feedback from the Remora vehicle, the motion planning system also gets input from the robot's environment. This feature separates the previous research [12] from current research. Environmental forces is an unpredictable element in the system, as the ocean current and waves may change the net structure of the fish cage.

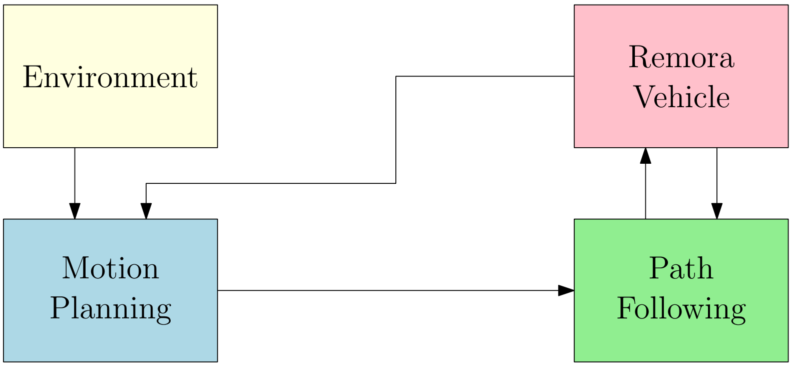


Figure 2.2: High-level system description of the autonomous net cleaning robot gives an overview of the different parts that work together. Motion planning is the focus of this project.

2.2.1 Path Planning vs Trajectory Planning

Path planning and trajectory planning have many similarities but can be separated by time. While path planning is time-independent, trajectory planning use time to represent the trajectory and where the vehicle should be at a specific time. In combination with trajectory planning, it is common to implement trajectory tracking. Dynamic environments involve disturbances, such as wind for an aeroplane, and are hard to model precisely due to the highly variant behaviour [16]. In the case of a strong headwind, it might not be possible to achieve the pre-computed trajectory. Given that the disturbances can vary and are hard to predict, trajectory tracking and trajectory planning can be challenging. In these situations, it would be beneficial to introduce path planning and path following, which are better suited for unmodelled disturbances. Due to the fact that ocean current can be compared to wind, path planning and path following are advantageous for this application, and the following will therefore focus on path planning and path following.

2.2.2 Path Following

Path following intends to make the robot converge to and follow a predefined path, where there are no constraints related to the time or velocity of the vehicle. Path following differs from trajectory tracking by its time independence. The path is often a combination of straight and curved lines that form the path from start to finish. Usually, path following consists of an attitude control system and a path following control system. The attitude control system ensures that the vehicle's attitude matches the desired attitude, often by a heading or course autopilot. The path following control system keeps the vehicle on the predefined path with predefined dynamics, ë.g., speed control. These two controllers need to communicate to ensure the wanted results [13].

Heading Guidance

Guidance, in general, can be defined as stated in [13]: "The process for guiding the path of an object towards a given point, which in general may be moving," i.e., the objective is to find the optimal reference heading to navigate the vehicle along the desired path.

Widely used and popular methods are Line Of Sight (LOS) guidance and Pure Pursuit (PP) guidance. These are simple and intuitive to implement and understand. LOS uses a three-point guidance scheme forming a triangle between the vehicle, path, and target. The guidance focus on minimising the cross-track error (y_e^p) by calculating the correct heading angle, denoted as χ_d in (2.4) and is provided by [13] as

$$\chi_d = \pi_p - \arctan\left(\frac{y_e^p}{\Delta}\right) \tag{2.4}$$

$$\pi_p = atan2(y_t^n - y_{ref}^n, x_t^n - x_{ref}^n) \tag{2.5}$$

$$\underbrace{\begin{bmatrix} \cos\left(\pi_p\right) & \sin\left(\pi_p\right) & 0 \\ -\sin\left(\pi_p\right) & \cos\left(\pi_p\right) & 1 \\ \tan\left(\pi_p\right) & -1 & 0 \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} x_p^n \\ y_p^n \\ y_e^p \end{bmatrix}}_{\boldsymbol{x}} = \underbrace{\begin{bmatrix} \cos\left(\pi_p\right) x^n + \sin\left(\pi_p\right) y^n \\ -\sin\left(\pi_p\right) x^n + \cos\left(\pi_p\right) y^n \\ \tan\left(\pi_p\right) x_t^n - y_t^n \end{bmatrix}}_{\boldsymbol{b}} \tag{2.6}$$

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}\ , \tag{2.7}$$

where Δ is the lookahead distance used to change the response of the desired heading. The cross-track error is obtained by extracting the third element in $\boldsymbol{x}$. The preceding equations corresponds to the illustration in Figure 2.3, where (x_t^n, y_t^n) is the target point of the LOS vector determined by the lookahead distance. (x_{ref}^n, y_{ref}^n), (x_i^n, y_i^n) and (x_{i+1}^n, y_{i+1}^n) represents the points along the path, and will be shifted when the vehicle reaches the point, or within a region, of (x_i^n, y_i^n).

PP can be seen as a simplified version of LOS, where the objective is to align the vehicle's velocity with the straight line between target and vehicle [17]. Many compare this algorithm to nature, where a predator is hunting prey by chasing the prey's tail. A combination of these gives Pure Pursuit and Line Of Sight (PLOS), which takes the best of two worlds, resulting in an easy and intuitive method that can handle disturbances to a certain degree.

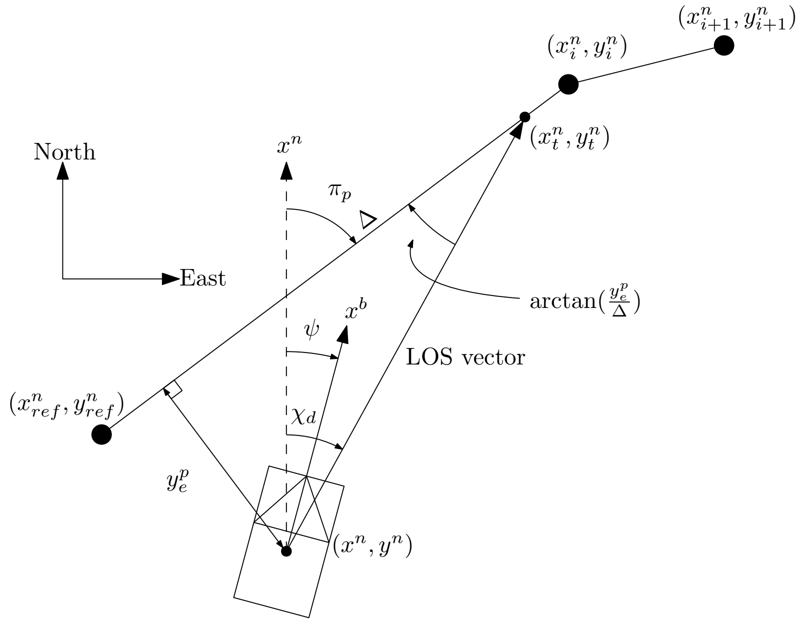


Figure 2.3: LOS guidance. Adapted from Figure 10.9 in [13].

Other more advanced method like Vector-Field (VF), Nonlinear Guidance Law (NLGL), and Liner Quadratic Regulator (LQR)-based methods exists and could be beneficial to investigate in detail [18]. However, the simple heading guidance method LOS will be sufficient for this application, and the mentioned methods will not be further reviewed in this thesis.

Velocity Guidance

Velocity guidance is used to adjust the vehicle's velocity to cope with different changes in the environment, such as; battery level, docking procedure, and danger zones. Task priority is an intuitive way of deciding where different velocities are assigned to different situations. For instance, if the vehicle is less than 1 meter from the docking station, the velocity should be decreased to avoid damages.

Another method is to use multi-objective optimisation, which is the process of optimising systematically and simultaneously a collection of objective functions [19]. Further, it utilises the Pareto optimality, which is the concept of defining an optimal point. Unlike a single-objective optimisation, typically, there is no single global solution, and it is often necessary to define a set of points to determine the optimum between the solutions.

As the focus in this thesis is the motion planner, simple velocity guidance is advantageous to implement. Therefore the task priority is chosen and gained from the motion planner, described later.

Control System

Previously, a PID and PD controller was used to regulate the heading and velocity respectively [12], illustrated in Figure 2.4. This showed promising results and maintained the desired states during simulations. However, as environmental disturbances are added to the system, a more advanced approach might be beneficial to examine to increase the performance of the controllers, and consequently, the system as a whole.

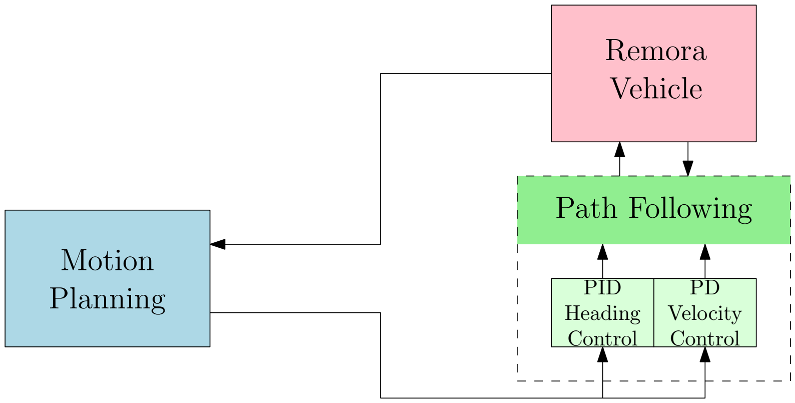


Figure 2.4: System description with the path following structure and control system with PID & PD controller as used in [12].

In this thesis, we have used a feedback linearisation controller following the same approach as presented in [15], where the stability of the system is also proven to be Uniformly Globally Exponentially Stable (UGES).

In general, one can say that a feedback linearisation controller intends to transform a nonlinear system into a controllable linear system by choosing the input u to cancel the nonlinear terms. For a nonlinear system on the form [20]

$$\dot{x} = A^{n \times n} x + B^{n \times p} \gamma(x)[u - \alpha(x)] ,$$

where A and B are controllable, $\alpha : R^n \rightarrow R^p$ and $\gamma : R^n \rightarrow R^{p \times p}$ are defined in a domain $D \subset R^n$ that contains the origin and $\gamma(x)$ is nonsingular for every $x \in D$. A controller on the form

$$u = \alpha(x) + \beta(x) v ,$$

where $\beta(x) = \frac{1}{\gamma(x)}$, yields a linear system

$$\dot{x} = Ax + Bv .$$

One can choose $v = -Kx$ to make $A - BK$ Hurwitz and ensure that the origin of the system $\dot{x} = (A - BK)x$ is globally asymptotically stable (GAS). This method is very powerful but requires perfect knowledge of the parameters of the mass and damping matrices, which may be hard to obtain [15].

In advance of any control system, it is common to implement a reference model. A reference model can smooth the desired states to prevent significant changes in a small period of time, reducing the jerky behaviour of the state. This is beneficial because the actuators need some time to reach the desired set point (depending on the actuator), which will increase the lifetime of the actuators by reducing wear and tear. However, tuning the model very slow will compromise the vehicle's performance, and it may not follow the desired path. The model can also generate the desired speed and acceleration to be used in a controller.

The reference model is often inspired by a physical system, and for marine crafts, it is convenient to use a mass-damper-spring system to generate the desired state trajectories [13]. A second-order system can be used to generate the desired velocities

$$\frac{\nu_{d_i}}{r_i^b}(s) = \frac{\omega_{n_i}^2}{s^2 + 2\zeta_i \omega_{n_i} s + \omega_{n_i}^2}\,, \tag{2.8}$$

where γ_i and ω_{n_i} are the relative damping ratio and natural frequency, respectively. (2.8) can be rewritten as a state-space representation

$$\begin{aligned} \dot{\boldsymbol{x}}_d &= \boldsymbol{A}_d \boldsymbol{x}_d + \boldsymbol{B}_d \boldsymbol{r}^b \\ \boldsymbol{A}_d &= \begin{bmatrix} \boldsymbol{0}_{n\times n} & \boldsymbol{I}_n \\ -\boldsymbol{\Omega}^2 & -2\boldsymbol{\Delta}\boldsymbol{\Omega} \end{bmatrix}, \boldsymbol{B}_d = \begin{bmatrix} \boldsymbol{0}_{n\times n} \\ \boldsymbol{\Omega}^2 \end{bmatrix}, \end{aligned}$$

where $\boldsymbol{x}_d$ is the desired state and is defined as $[\boldsymbol{\nu}_d^\top, \dot{\boldsymbol{\nu}}_d^\top]^\top \in \mathbb{R}^{2n}$. A step in the command $\boldsymbol{r}^b$ will give a smooth response in the acceleration and velocity.

Similarly, a third-order system can be defined as

$$\frac{\eta_{d_i}}{r_i^n}(s) = \frac{\omega_{n_i}^2}{(1 + T_i s)\left(s^2 + 2\zeta_i \omega_{n_i} s + \omega_{n_i}^2\right)}\,, \tag{2.9}$$

and is often used as a reference model for position and attitude [13].

The feedback linearisation controller was mainly used in the simulations, but a comparison study with a PID and PI controller was conducted to evaluate the difference in the performance. The reference models explained above was used to smooth out the wanted behaviour from the motion planning method.

2.2.3 Motion Planning Concepts

Motion planning can be compared to a hiker that navigates with a map and compass, planning a route from A to B that is best suited based on the hiker's skills and abilities. Another hiker may choose another route based on different prerequisites. This planner also needs to handle the event of an obstacle, i.e., an avalanche that blocks the original path. The hiker analogy can be extended to planning a path for a robot. Each robot has different capabilities and different working environments. An example of motion planning can be the highly popular robot lawnmower. They work on a static, non-changing workspace with a simple algorithm that makes sure that it does not get stuck and reaches the charging station before the battery runs out.

In contrast, the net cleaning robot operates on a net structure affected by ocean currents and waves, leading it to deform. In addition, other obstacles may interfere, creating regions of avoidance, e.g., fish, sensors, camera, ropes, or other foreign objects. Consequently, creating a dynamic environment that requires an advanced motion planner robust to environmental changes. Due to a large workspace and limited power source, an energy-efficient solution covering the whole net is also a criterion.

Motion planning is a highly researched area within robotics, and there are many applications where this has been implemented with success. A consequence of this is an extensive amount of terms and methods that may be confusing. Therefore a structured approach is advantageous when evaluating this topic. There are three types of motion planning concepts: global, local, and reactive motion planning [21].

Global Motion Planning

Global path planning involves methods that rely on predefined workspace using a topological map [22]. These methods will find a path prior to the operation based on known information. However, they are not very well suited to take unknown dynamic obstacles into account due to the computational cost to produce the global solution repeatedly when encountering an obstacle. The most used global motion planners are A* Family, RRT Family, and Voronoi. More information can be found in [21, 23], and in the specialisation thesis [2].

Local Motion Planning

Local planning methods intend to calculate a collision-free path to add to the graph [24]. Unlike the global method, local methods do not solve the entire planning problem, but small segments in a usually simple way, e.g., shortest path.

These methods are used if the start and goal configurations are close to each other or to avoid obstacles [25]. A local motion planning algorithm has the shortcoming where the algorithm can become "stuck" in a local minimum, unable to find the best path or even any feasible path. As some local planners try to connect two points with a feasible path, they are sometimes dependent on a global planner that produces the two points. This configuration can be seen in Figure 2.5 and is referred to as a hybrid system.

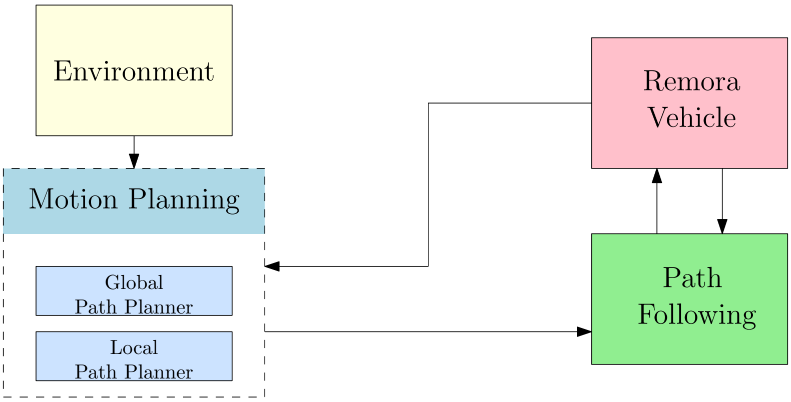


Figure 2.5: One possible solution where motion planning is divided into global and local planning to create a hybrid solution.

The work in [26] utilised a version of a cellular decomposition method with Probabilistic Road Map (PRM) to make a hybrid system and an efficient motion planning algorithm on 3- and 4-DOF robots. The cellular decomposition is often used for complete coverage problems and can work as both a global and local path planner. The PRM algorithm samples the workspace and adds the sample to a graph as a node, and then use a shortest path algorithm to find the shortest feasible path from start to the end node, working as a global and local path planner. In [26] the cellular decomposition was used as a global planner, while the PRM calculated the local path inside the cell structure provided by the cellular decomposition method.

Some local planners are Fermat's spiral [27], clothoids, state lattices (also global), RRT Family [21], and Dubins path [24]. Dubins path is based on the findings in [28], and is summarized in [13] as: *The shortest path (minimum time) between two poses* (x_i^n, y_i^n, ψ_i^n) *and* $(x_{i+1}^n, y_{i+1}^n, \psi_{i+1}^n)$ *of a craft moving at constant speed* U *is a path formed by straight lines and circular arc segments.* This means that an optimal path will be a combination of right and left turns and straight lines, as Figure 2.6 illustrates. Dubins path is well suited for use in a sampling-based planning algorithm [24]. A sampling-based algorithm samples the workspace with, e.g. random points, before connecting some or all of the points to create a graph, which may create a jerky path. The Dubins path is then used to ensure curvature continuity on the path.

Dubins path has also been modified to include obstacle avoidance and turn rate, and acceleration limits. Note that the Dubins path only accounts for forward-moving motions. The work done in [29] proposed a path that utilises the properties of the Dubins path but handles both forward and reverse direction and is known by Reeds Shepp path.

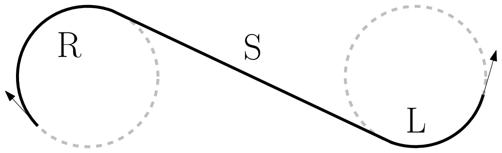


Figure 2.6: The Dubins path illustrated with a right-hand turn, followed by a straight line and a left-hand turn. Image inspired by Figure 2 in [30].

Reactive Motion Planning

Reactive motion planning is referred to as a class of algorithms that only use the local knowledge of the obstacle field to plan [31]. This class is useful when dealing with uncertainty and obstacles explored at the last minute due to its fast response time. In addition, they can ensure local feasibility. As these planners do not solve the global motion planning problem, they are often used in combination with another planner. Lately, bio-inspired reactive planning algorithms have been investigated. A 3D bio-inspired neural network model was used to solve the path planning problem from one point to another with complete coverage of an area with obstacles [32].

The work in [33] introduces a planner that considers reflex actions and planning with lazy techniques to handle obstacle changes by using the lazy PRM method to compute a collision-free feasible path over the global area. When the robot moves and a dynamic obstacle is detected, a reactive method, Deformable Virtual Zone (DVZ), takes control of the motion planning and generates commands to force the robot away from the obstacles before the lazy PRM handle the planning again. The reaction, triggered by, e.g. a sensor that detects an obstacle, which causes a change in the behaviour of the vehicle, is often defined as reactive planning and separates this type from conventional planners [31]. The lazy version of the PRM tries to minimize the number of collision checks to improve time efficiency [34]. Detailed description regarding the DVZ can be found in [33].

Braitenberg vehicles are considered as a part of reactive navigation [35]. This is widely used in commercial robot lawnmowers and robot vacuum cleaners. It will move straight forward until a sensor senses an obstacle, where it will turn by a predefined angle and repeat the process. This is a robust application that neglects the environment making it suitable for dynamic workspaces. However, since the resulting path will overlap, it has a low coverage rate making it suboptimal for an application with a limited power supply. A regular-sized fish cage has a large surface compared to the vehicle's cleaning area, so it is essential to move efficiently on the net, finding the shortest path while covering the whole net. Therefore, this method is not eligible for use in an aquaculture environment.

Another reactive motion planning method is the EBM. Introduced by Quinlan and Khatib in 1993 for robotic path planning, and later used widely in automotive problems, within lane changing and vehicle following [36, 37]. This method optimises a global plan locally by minimising the path's length while taking moving obstacles into account. One advantage is that it optimises the path incrementally, meaning that the longer the robot moves, the more time the algorithm improves the path, resulting in a better output. However, if the environment changes are too large, the algorithm could fail to deform to a collision-free path even if one exists. An analogy to a robot can be made, where a planned path through a door gets unavailable if someone closes the door. It might be another path through another open door, but this may require global replanning [36]. The work in [37] used a solution that implemented several elastic bands reacting to the obstacles. The best one was chosen from specific criteria and thus improved the algorithm's ability to plan in complex environments.

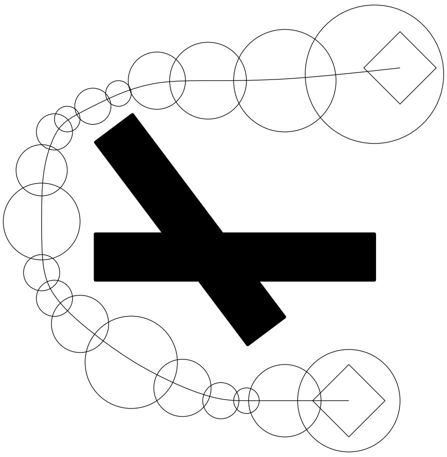


Figure 2.7: Bubble concept to maintain a collision-free path. Image courtesy of [36].

The elastic band algorithm use bubbles to maintain a collision-free path, as shown in Figure 2.7. The series of bubbles represent an elastic band, where the bubbles vary in numbers and size. In a complex environment, the number of bubbles increases, and the radius decreases, and vice versa in the opposite situation. A path follower may use the size of the bubble as an input on the desired velocity. The low number of bubbles keeps the computational cost low, allowing the algorithm to run in real-time. In Figure 2.7 the bubbles are represented by circles in 2D. However, [38] used the elastic band technique in 3D for an autonomous helicopter and is used as a base for this application. This method will be explained in detail later.

Reinforcement Learning for Motion Planning

Reinforcement Learning (RL) is the ability of a system to learn what to do in a different situation, based on previous results. It must learn what gives the highest reward by trying different actions in different situations. An analogy can be made to a mobile robot that decides whether to enter a new room and continue its operation or return to the charging station. This decision is made from the battery level and from previous attempts that give information about how easy it was to reach the charging station in previous situations [39].

RL is described as a machine-learning paradigm that addresses the task of how an agent (animal, human, or machine) interacts with its environments [40]. Based on the behaviour of different obstacles that act differently from time to time, this method can act accordingly. The method was used to solve the motion planning problem on a 4-wheeled robot that navigated among moving humans in a crowded environment [41]. Here the humans represent an agent and have unpredictable behaviour. Instead of predicting or modelling complex interaction and cooperation, collision avoidance was achieved without assuming that other agents followed any particular behaviour model.

This method has also been implemented in underwater environments, where a deep reinforcement learning method was used to solve the motion planning problem of an underactuated Autonomous Underwater Vehicle (AUV) in a mapless environment [42]. It has been used in a wide variety of other applications, too; reaching a target in a high current environment, target and trajectory tracking, obstacle avoidance, and making a helicopter hover [42].

Using RL to solve the motion planning problem is relatively complicated and is not considered suitable for this project. However, it is a promising solution implemented in several applications with satisfying results, making it possible to investigate in the future.

Choosing Motion Planner

The specialisation project [2] reviewed several motion planners of each of the three types of motion planning. Furthermore, was the Rapidy-Exploring Random Tree Star (RRT*), EBM and cellular decomposition simulated in a 2D environment resembling the size and shape of a fish cage to look closer at each algorithms strengths and weaknesses. The cellular decomposition and the elastic band method outperformed RRT*. They were suggested to work in a hybrid solution, where the cellular decomposition provides the initial path needed in the EBM, and improve the global planning performance, while the EBM would find a feasible path in real-time and avoiding obstacles. However, for simplicity, this thesis will provide the initial path in advance and will only implement the EBM.

2.3 Elastic Band Method

This section covers the details of the EBM, how it is modified from [38], how it is used, and what parameters one can use to tune the behaviour to match the specific application.

2.3.1 The Phases of the EBM

The EBM has been used for 3D path planning for a helicopter to handle both static and dynamic obstacles. They achieved real-time performance and satisfactory performance in flight manoeuvres [38]. By partitioning the algorithm into 16 steps and four phases, they presented the algorithm precisely and clearly. The 16 steps together with the four phases are the basis for this thesis, but some changes are made to adapt the solution to this specific application. The same notation is used, except vectors, where $\boldsymbol{v}$ is used instead of $\vec{v}$. Illustrations used are presented in 2D for simplicity.

Phase 1: Initial Path

For the algorithm to work, it is necessary to have an initial path that can be adjusted in order to avoid obstacles. A path consists of a series of bubbles, $\boldsymbol{b}$, with a position and a radius: $\boldsymbol{b} = [x, y, z, R^{bub}]$. For optimal performance, the initial path should be the shortest path from start to finish in the case of obsolete obstacles, with bubbles evenly spaced along the path. From now, we denote the initial path with $\boldsymbol{\Gamma}_{init}$ and the elastic band path with $\boldsymbol{\Gamma}_{EB}$.

To choose the bubble radius, some limitations need to be set, and the following equation for R^{bub} is suggested [38]

$$2D^{vehicle} = R^{bub}_{min} \leq R^{bub} \leq R^{bub}_{max} = V^{vehicle}_{max} \Delta T \ , \tag{2.10}$$

where $D^{vehicle}$ is the diameter/width of the vehicle, ΔT is the time interval of local path planning, and $V^{vehicle}_{max}$ is the maximum speed of the vehicle. Reasonably values can be set independently of (2.10), as long as $R^{bub}_{min} \leq R^{bub} \leq R^{bub}_{max}$ is satisfied. The size of the bubbles is of significant importance as this will be a part of the obstacle avoidance and will also be used as guidance for the speed of the robot.

Phase 2: Elastic Band Deformation

In order to adapt the path continuously concerning static and dynamic obstacles, two virtual forces are implemented. The first force is called the internal force, which works between the internal bubbles in the elastic band and intends to contract the bubbles leading to a shorter and smoother path. Each node is attracted by its preceding and following node by the following equation

$$\begin{aligned}\boldsymbol{F}^{bi}_{int} = k_{int} \Bigg(& \frac{\boldsymbol{b}_{i+1} - \boldsymbol{b}_i}{\|\boldsymbol{b}_{i+1} - \boldsymbol{b}_i\|} \left(\|\boldsymbol{b}_{i+1} - \boldsymbol{b}_i\| - R^{bub}_{\min}\right) \\ & + \frac{\boldsymbol{b}_{i-1} - \boldsymbol{b}_i}{\|\boldsymbol{b}_{i-1} - \boldsymbol{b}_i\|} \left(\|\boldsymbol{b}_{i-1} - \boldsymbol{b}_i\| - R^{bub}_{\min}\right) \Bigg) \ , \end{aligned} \tag{2.11}$$

where $\|\cdot\|$ denotes the Euclidean norm and calculates the distance between two bubbles, while k_{int} decides the contraction gain. $\boldsymbol{F}^{bi}_{int} \in \mathbb{R}^n$ where n is the number of states of the position in the bubbles $\boldsymbol{b}$. Figure 2.8 shows how the two parts of (2.11) calculate the distance and direction from one bubble to another, subtracting the radius of the smallest possible bubble. The greater the distance, the greater the internal force, trying to pull the bubbles closer. All parameters are thoroughly described in Section 2.3.3 to understand how each value affects the behaviour of the algorithm.

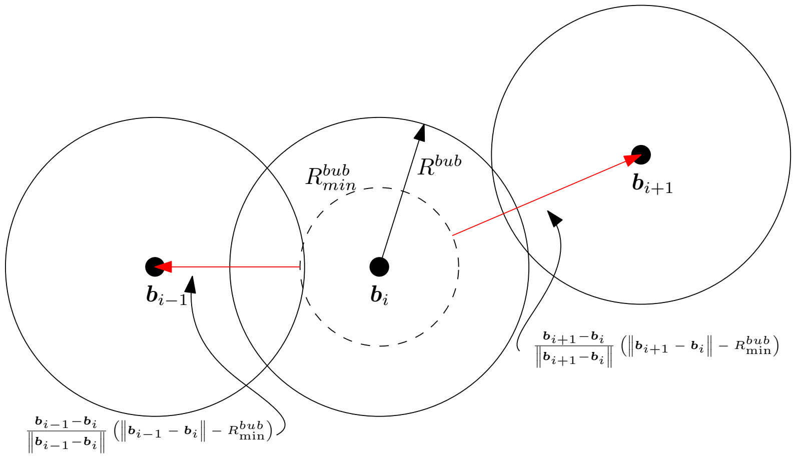

R^{bub}_{min}
R^{bub}
$\boldsymbol{b}_{i+1}$
$\boldsymbol{b}_{i-1}$
$\boldsymbol{b}_i$
$\frac{\boldsymbol{b}_{i+1}-\boldsymbol{b}_i}{\|\boldsymbol{b}_{i+1}-\boldsymbol{b}_i\|}\left(\|\boldsymbol{b}_{i+1}-\boldsymbol{b}_i\| - R^{bub}_{\min}\right)$
$\frac{\boldsymbol{b}_{i-1}-\boldsymbol{b}_i}{\|\boldsymbol{b}_{i-1}-\boldsymbol{b}_i\|}\left(\|\boldsymbol{b}_{i-1}-\boldsymbol{b}_i\| - R^{bub}_{\min}\right)$

Figure 2.8: Internal force calculations.

The second virtual force is called the external force and is decided from how the obstacle ($\boldsymbol{O}_j$) interact with the path. $\boldsymbol{O}_j = [x, y, z, R^{Obst}]$ is defined equal to the bubbles for simplicity and is stored in an array. The subscript $_j$ denotes the obstacle's index number in the array. Each node is affected by all nearby obstacles if they are near enough $\left(\left\|\boldsymbol{b}_i - \boldsymbol{O}_j\right\| < R^{bub}_i + R^{Obst}_j\right)$, and is repelled by a force from each obstacle defined as

$$f^j_{ext}(b_i) = k_{ext} e^{-(D_{aff})} \left(\frac{\boldsymbol{b}_i - \boldsymbol{O}_j}{\left\|\boldsymbol{b}_i - \boldsymbol{O}_j\right\|} \right) , \tag{2.12}$$

where k_{ext} is the repulsive gain, and the fading function $e^{-(D_{aff})}$ is dependent on the affected distance given by

$$D_{aff} = \|\boldsymbol{b}_i - \boldsymbol{O}_i\| - D_{safe}, \text{ if } \left\|\boldsymbol{b}_i - \boldsymbol{O}_j\right\| > D_{safe}; \text{ zero otherwise.} \tag{2.13}$$

The safe distance is defined as: $D_{safe} = R^{bub}_{min} + V^{obs}_{bub} T_s$, where V^{obs}_{bub} is the relative velocity from the obstacle to the specific bubble, and T_s is the sampling interval of obstacle detection. If $D_{aff} = 0$, the fading function equals one, and the repulsive force will be greater than if the obstacle is some distance away from the specific bubble. This is desired as the reaction needs to be larger if the obstacle is closer.

As Figure 2.9 illustrates, the unit vector is calculated (shown in solid red arrow), as well as the distance (shown as dashed red line) between $\boldsymbol{b}_i$ and $\boldsymbol{O}_j$. In the case where the obstacle is static, $V^{obs}_{bub} = 0 \implies D_{safe} = R^{bub}_{min}$, from the figure one can see that $\left\|\boldsymbol{b}_i - \boldsymbol{O}_j\right\| > D_{safe}$ holds, and $D_{aff} = \|\boldsymbol{b}_i - \boldsymbol{O}_i\| - D_{safe}$ (also shown in Figure 2.9). This situation would result in a fading function with a value of less then 1, consequently reducing the external repulsive force compared to no fading function. This property lets the external force vary with the distance from the safe distance to the obstacle, resulting in a higher resolution of external force.

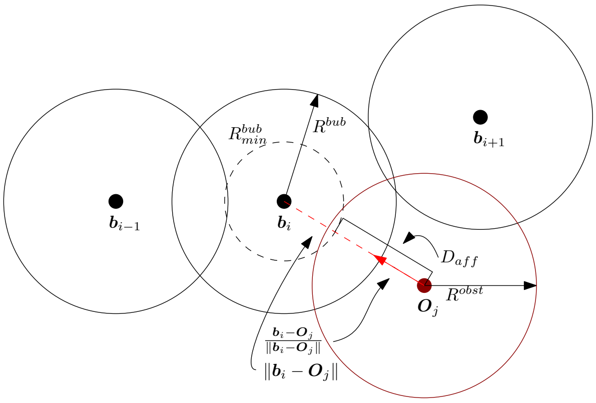

R^{bub}_{min}
R^{bub}
$\boldsymbol{b}_{i+1}$
$\boldsymbol{b}_{i-1}$
$\boldsymbol{b}_i$
D_{aff}
R^{obst}
$\boldsymbol{O}_j$
$\frac{\boldsymbol{b}_i - \boldsymbol{O}_j}{\|\boldsymbol{b}_i - \boldsymbol{O}_j\|}$
$\|\boldsymbol{b}_i - \boldsymbol{O}_j\|$

Figure 2.9: External force calculations.

The radius of the obstacle, R^{obst}, visualised in Figure 2.9, is used when calculating proximity and determine when an obstacle is near enough to react. Further, the external force is summed over each obstacle nearby

$$\boldsymbol{F}^{bi}_{ext} = \sum_{j=1}^{N} \boldsymbol{f}^{j}_{ext}(b_i) \ . \tag{2.14}$$

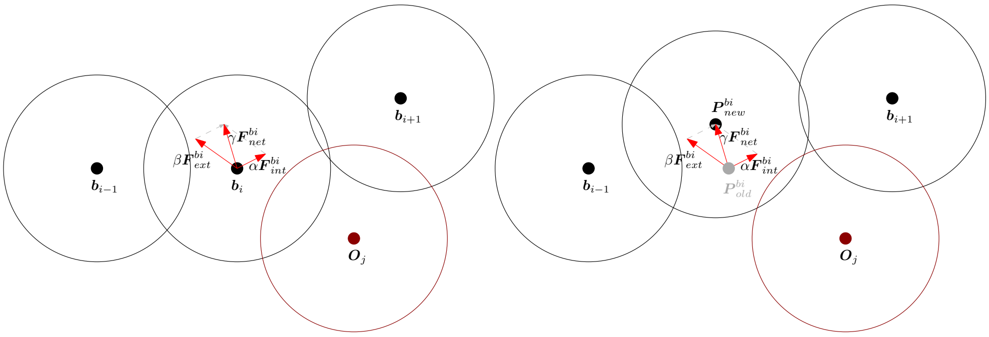
Eventually, the internal and external force is combined to calculate the net force applied to the bubble b_i, with weighting factors for internal (α) and external forces (β)

$$\boldsymbol{F}^{bi}_{net} = \alpha \boldsymbol{F}^{bi}_{int} + \beta \boldsymbol{F}^{bi}_{ext} \ . \tag{2.15}$$

At last, the resulting net force multiplied by the step size for updating the bubble deformation (γ) is added to the old position of the bubble and updated by the following equation

$$\boldsymbol{P}^{bi}_{new} = \boldsymbol{P}^{bi}_{old} + \gamma \boldsymbol{F}^{bi}_{net} \ , \tag{2.16}$$

where $\boldsymbol{P}^{bi}_{new}$ represent the new position of the centre point of the bubble i in the path $\boldsymbol{\Gamma}_{EB}$. The result is visualised in Figure 2.10 where the net force is calculated (Figure 2.10a) and applied to the old position of the bubble, adapting the path according to the net force (Figure 2.10b).



(a) Net force calculations.

(b) New position of the bubble.

Figure 2.10: Net force calculations applied on the old position, creating a new position of the bubble.

Phase 3: Bubble Reorganisation

Phase 3 is crucial to maintain feasibility and improve efficiency. Two operations define bubble reorganisation. The first operation checks if a bubble is redundant and removes it. A bubble is considered redundant if (2.17) holds. Figure 2.11 illustrates a situation where (2.17) holds (the first condition to the left and the last condition to the right).

$$|R_k - R_{k-1}| \geq \|\boldsymbol{b}_k - \boldsymbol{b}_{k-1}\| \text{ or } R_{k+1} + R_{k-1} > \|\boldsymbol{b}_k - \boldsymbol{b}_{k-1}\| + \|\boldsymbol{b}_{k+1} - \boldsymbol{b}_k\| \quad (2.17)$$

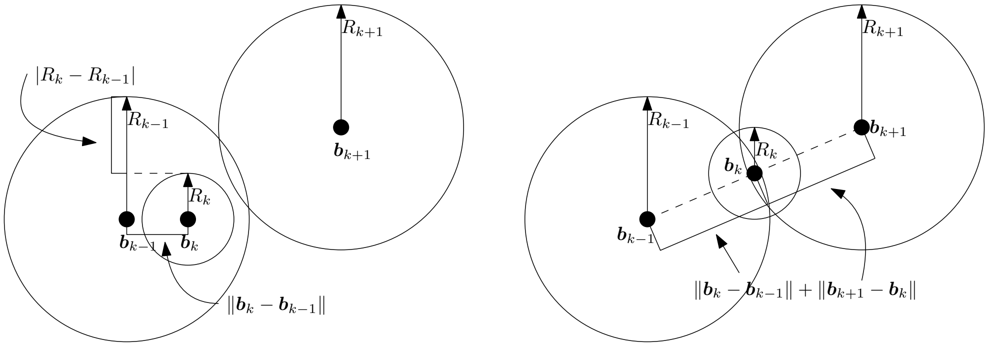


Figure 2.11: Redundancy check by evaluating (2.17).

The second operation checks if there is a connection between two bubbles by evaluating (2.18), and if not, a new bubble is inserted between the two bubbles,

$\boldsymbol{b}_k$ and $\boldsymbol{b}_{k-1}$. Figure 2.12 illustrates the situation where (2.18) holds. There is a gap between bubble k and $k-1$, and a new bubble should be inserted.

$$R_k + R_{k-1} - d_{ol} < \|\boldsymbol{b}_k - \boldsymbol{b}_{k-1}\| \ , \tag{2.18}$$

where d_{ol} is the desired overlap distance between two bubbles. d_{ol} is set by preference and can be viewed as a safety parameter that increases or decrease the overlap between bubbles, and consequently, the number of bubbles along the path. If increased, the accuracy and resolution will be increased but at the expense of time complexity.

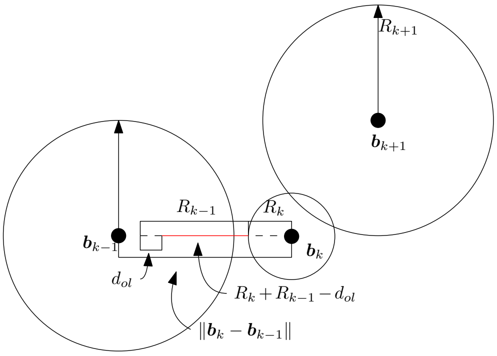


Figure 2.12: Check for gap between bubbles by evaluating (2.18).

However, in the edge case, a bubble might appear redundant and "missing" at the same time. This can happen if the desired overlap, d_{ol}, is greater than the overlap between $\boldsymbol{b}_{k+1}$ and $\boldsymbol{b}_{k-1}$. As visualised in Figure 2.13, where the leftmost figure illustrates (2.17) which in this case holds, and consequently the k-th bubble is removed from the elastic band, as shown at the right part of the figure. Here one can see that (2.18) is used to determine if the band has a broken connection. Due to the desired overlap d_{ol}, the equation holds since the red line is smaller than the two radiuses combined minus the desired overlap. The algorithm will then insert a new bubble between the two, and the same situation will appear again.

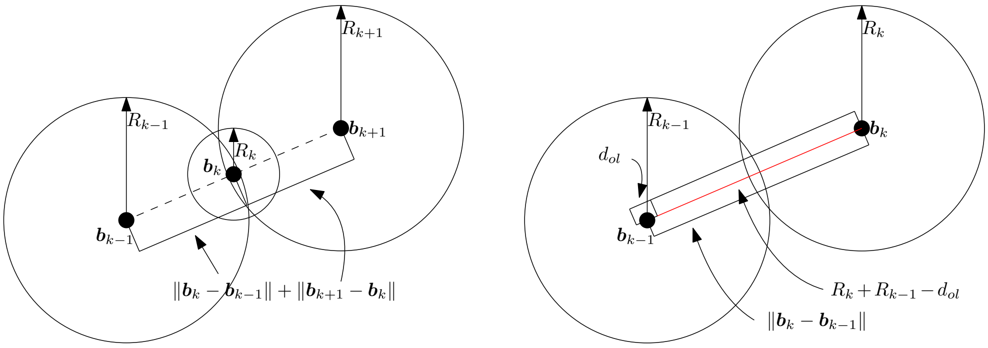


Figure 2.13: Redundant and "missing" at the same time.

This thesis presents a solution to the problem. By adding the same desired overlap in (2.18) to (2.17) the updated equation yields

$$
\begin{aligned}
&|R_k - R_{k-1}| \geq \|\boldsymbol{b}_k - \boldsymbol{b}_{k-1}\| \text{ or} \\
&R_{k+1} + R_{k-1} > \|\boldsymbol{b}_k - \boldsymbol{b}_{k-1}\| + \|\boldsymbol{b}_{k+1} - \boldsymbol{b}_k\| + d_{ol} \,.
\end{aligned}
\tag{2.19}
$$

This modification makes sure the desired overlap is accounted for when checking for redundancy as well.

Phase 4: Trajectory Transformation

This phase has two objectives. The first intends to make the resulting path more feasible for a vehicle to follow by smoothing the path based on the elastic band bubbles. One solution is to use a cubic B-spline technique, suggested by [38]. The details regarding the cubic B-spline will not be explained in this thesis, but the general idea is to improve the feasibility of the solution by "filtering" out unnecessary points and, in addition, creating a shorter path. The downside is a less accurate path, and in a situation where complete coverage and collision avoidance is of importance, this should be handled with care. This is illustrated in Figure 2.14, where the resulting path (solid line) is much smoother than the original path (dashed line), but at the expense of accuracy.

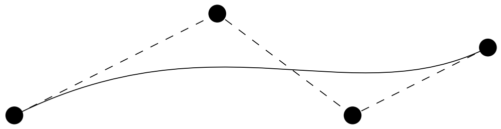


Figure 2.14: Spline principle.

If the downside of using this method outweigh the benefits, smoothing could be left to the control system through the use of reference models discussed in Section 2.2.2. It can also be handled in combination with the path following system, e.g. the LOS heading guidance which can be adjusted to output specific behaviour by tuning the lookahead distance, discussed in Section 2.2.2.

The second objective is to decide the vehicles desired velocity. This can be defined as

$$
\boldsymbol{V}_i = \frac{\boldsymbol{b}_{i+1} - \boldsymbol{b}_i}{T_c} \,, \tag{2.20}
$$

where $\boldsymbol{b}_{i+1}$ and $\boldsymbol{b}_i$ are the next and current bubble, and T_c is a predefined time interval. In doing so, the distance between two bubbles is used to calculate the desired velocity for each bubble [38]. Another method, proposed in [36], involves using the bubble size as a reference for the vehicle's speed since the size varies with the environments. This method only calculates the desired speed; therefore, separate heading guidance is required.

2.3.2 The Steps of the EBM

Described by [38] the following algorithm ensures a real-time trajectory path planner. The phases explained in Section 2.3.1 is divided into smaller steps to make the algorithm more concrete.

1. Construct the initial path $\mathbf{\Gamma}_{init}$ consisting of $N_{bubbles}$ bubbles allocated evenly between the start waypoint and the goal waypoint. Referring to Phase-1 in Section 2.3.1
2. Perform the elastic band deformation process from the current bubble node through all the uncompleted bubble nodes. Referring to Phase-2 in Section 2.3.1
    a. Compute the internal forces acting on the i-th bubble node due to its neighbor nodes of index $i-1$ and $i+1$ respectively. (2.11)
    b. Calculate the external forces acting on the i-th bubble node due to all the static and dynamic obstacles. (2.12)-(2.14)
    c. Execute the path deformation for the i-th bubble node applying the sum of the virtual forces given in steps 2a and 2b. (2.15)-(2.16)
    d. Repeat steps 2a-2c until the deformation amount is less than the tolerance.
    e. Decide bubble's radius on the smallest clearance distance to all the obstacles.
    f. Shift the bubble index to the next and repeat steps 2a-2e until the end of the elastic band, $i = N_{\text{bubbles}}$.
3. Do the bubble reorganisation from the current bubble node through all the uncompleted bubble nodes. Referring to Phase-3 in Section 2.3.1
    a. Check if the i-th bubble is redundant and delete it if (2.19) holds.
    b. Check if the i-th bubble has a broken connection with its previous one and insert an extra bubble at the middle of two bubbles if (2.18) holds.
    c. Do the path deformation if an extra bubble is generated. Referring to Steps 2a-2d.
    d. Decide the radius of the bubble as the smallest clearance distance to all obstacles.
    e. Shift the bubble index to the next and repeat steps 3a-3d until the end of the elastic band, $i = N_{\text{bubbles}} - 1$.
4. Generate the collision-free trajectory from the current position to the goal. Referring to Phase-3 in Section 2.3.1
    a. Smooth the deformed elastic band path $\mathbf{\Gamma}_{EB}$ by using a cubic spline function.
    b. Generate a real-time trajectory by correlating the path $\mathbf{\Gamma}_{EB}$ with time using (2.20).

Repeat steps 2-4 until the goal is achieved ($i = N_{bubbles}$).

Step 3d is added due to the tolerance of deformation. In the case where an extra bubble is generated, virtual force is applied until the deformation reaches the preset tolerance of deformation, and the bubble may still be in a non-feasible place. A proposed solution is then to resize the bubble such that the overlap with any obstacles minimises or disappears altogether.

Note that step 4 is not used in this thesis, as a reference model together with an appropriate lookahead distance in the LOS heading guidance is used to achieve the same goal.

2.3.3 Important Parameters and Pseudocode

The EBM is highly adjustable and has several parameters that can be adjusted in order to adapt and optimize the algorithm for specific applications. This section explains the parameters listed in Table 2.2 and their impact.

Table 2.2: EBM parameters.

Parameter name	Description
R_{min},R_{max},R^{bub}	Minimum, maximum and the actual radius of the bubbles
k_{int}	Contraction gain
k_{ext}	Repulsive gain
α	Weighting factor for internal forces
β	Weighting factor for external forces
γ	Step size for updating the elastic band deformation
d_{ol}	Desired overlap distance between two bubbles
$deformTol$	Tolerance for deformation
$spinConst$	Prevent the algorithm from "spinning" at one place

Several of these parameters were proposed by [38]:

- R_{min} and R_{max} sets the limits of R^{bub} if change of radius is needed. Their initial values are explained in Section 2.3.1. In a system where the size is used for speed guidance, these parameters play a vital role as they decrease near obstacles and increase if the vehicle moves away from the obstacles.
- k_{int} and k_{ext} are the contraction and repulsive gain, respectively, and is used to amplify or reduce the force calculated from the internal or external force. Increasing k_{int} would result in a higher contraction force that would reduce the length of the path and contract the bubbles towards each other. Increasing k_{ext} would increase the repulsive force, and thus amplifying the reaction to an obstacle.
- α, and β works together with k_{int} and k_{ext} in the sense that they are weighting factors that decide how much of the calculated force should be applied to

the bubble when deforming the path. As before, an increase in value would increase the impact on the path. If the workspace is complexed and obstacle avoidance is crucial, one should consider increasing β. Alternatively, if there are few and the region of avoidance can be detected a long time ahead, one could get a smoother path by increasing α.

- γ is the step size used for updating the new position. This value is determined by how often one calls the EBM, and in simulations, it is limited by the integrator step size.
- d_{ol} describes the desired overlap between bubbles and intends to make the algorithm more robust by having some margin of deformation without losing connection with neighbouring bubble instantaneously.
- $deformTol$ is used to limit how much one allows the algorithm to deform at each timestep. This is a safety feature to avoid the elastic band deforming beyond sensible limits. $deformTol$ can be set pretty low, as long as k_{int}, k_{ext}, α and β is properly tuned.

In addition to these, the practical implementation of the method for the present purpose required the introduction of an additional auxiliary parameters:

- $spinConst$ is another safety feature designed to avoid the algorithm to adapt the path if the deformation is very small. This prevents the algorithm sometimes to get stuck and improves time efficiency.

The way these parameters are used and how the algorithm is structured is better illustrated through the pseudocode in Algorithm 1, 2 and 3 below.

Algorithm 1: Elastic band.

```
input  : Initial path $\mathbf{\Gamma}_{init}$ from A to B, or last $\mathbf{\Gamma}_{EB}$
output: $\mathbf{\Gamma}_{EB}$
for $i \leftarrow 1$to length of $\mathbf{\Gamma}_{EB}$ do
    $\mathbf{\Gamma}_{EB} = ElasticBandDeformation(\mathbf{\Gamma}_{EB}, \mathbf{O}, i)$
    $\mathbf{\Gamma}_{EB} = NewRadiusBubble(\mathbf{\Gamma}_{EB}, \mathbf{O}, i)$
while $k \leftarrow 2 \leq$ length of $\mathbf{\Gamma}_{EB} - 1$ do
    if (2.17) then
        Delete $\mathbf{\Gamma}_{EB_k}$
    else if (2.18) then
        $\boldsymbol{newBubble} = (\mathbf{\Gamma}_{EB_k} - \mathbf{\Gamma}_{EB_{k-1}})/2 + \mathbf{\Gamma}_{EB_{k-1}}$
        Insert $\boldsymbol{newBubble}$ at $\mathbf{\Gamma}_{EB_k}$
        $\mathbf{\Gamma}_{EB} = ElasticBandDeformation(\mathbf{\Gamma}_{EB}, \mathbf{O}, i)$
        $\mathbf{\Gamma}_{EB} = NewRadiusBubble(\mathbf{\Gamma}_{EB}, \mathbf{O}, i)$
    else
        $k = k + 1$
```

Algorithm 1 is the algorithm called from the main program and returns the elastic band updated according to the environments. The first iteration inputs the initial path, $\boldsymbol{\Gamma}_{init}$, before it is adapted, and the updated path is returned as $\boldsymbol{\Gamma}_{EB}$. The next iteration inputs $\boldsymbol{\Gamma}_{EB}$, adapts and updates the path before returning the new version of $\boldsymbol{\Gamma}_{EB}$. By constantly adapting the path, one can allow small values for the contraction and repulsive gains to avoid overshoot but still manage to deform the path enough to avoid any obstacles.

Algorithm 2 is called from within Algorithm 1 and performs the second phase of the EBM described in Section 2.3.1. This is where the internal and external forces are calculated, and the deformation process is conducted and verified that the new position is within the physical regions of the workspace. A check if the deformation should continue is made before the updated path, $\boldsymbol{\Gamma}_{EB}$ is returned.

Algorithm 2: ElasticBandDeformation.

```
input  : $\boldsymbol{\Gamma}_{EB}, \boldsymbol{O}, i$
output: $\boldsymbol{\Gamma}_{EB}$
$\boldsymbol{b}_{base} = \boldsymbol{\Gamma}_{EB_i}$
contDeform = true
while deformation<deformTol and contDeform do
    if i is first or last element then
        $\boldsymbol{F}^{bi}_{int} = 0$
    else
        $\boldsymbol{F}^{bi}_{int} = (2.11)$
    while $j \leftarrow 1 <$ length of $\boldsymbol{O}$ do
        if $\left\|\boldsymbol{\Gamma}_{EB_i} - \boldsymbol{O}_j\right\| < \boldsymbol{\Gamma}_{EB_i}.R^{bub} + \boldsymbol{O}_j.R^{bub}$ then
            $D_{aff} = (2.13)$
            $\boldsymbol{f}^{j}_{ext} = (2.12)$
        $j++$
    $\boldsymbol{F}^{bi}_{ext} = (2.14)$
    $\boldsymbol{F}^{bi}_{net} = \alpha\boldsymbol{F}^{bi}_{int} + \beta\boldsymbol{F}^{bi}_{ext}$ (2.15)
    $\boldsymbol{P}^{bi}_{new} = \boldsymbol{P}^{bi}_{old} + \gamma\boldsymbol{F}^{bi}_{net}$ (2.16)
    $deformation = \left\|\boldsymbol{b}_{base} - \boldsymbol{P}^{bi}_{new}\right\|$
    if $\left\|\boldsymbol{\Gamma}_{EB_i} - \boldsymbol{P}^{bi}_{new}\right\| < spinConst$ then
        $contDeform = false$
    Check if the position is within the limits of the workspace, and replace
      $\boldsymbol{P}^{bi}_{old}$ with $\boldsymbol{P}^{bi}_{new}$
```

Algorithm 3 calculates a new bubble radius if needed, with limits between the predefined constants R_{min} and R_{max}. This is called from Algorithm 1 after each deformation and after a new bubble is added and deformed.

Algorithm 3: NewRadiusBubble.

```
input  : $\boldsymbol{\Gamma}_{EB}, \boldsymbol{O}, i$
output: $\boldsymbol{\Gamma}_{EB}$
for $j \leftarrow 1 <$ length of $\boldsymbol{O}$ do
    $\boldsymbol{L}_j = \left\| \boldsymbol{\Gamma}_{EB_i} - \boldsymbol{O}_j \right\|$
find the minimum value in $\boldsymbol{L}$ ($L_{min}$) and the corresponding index ($i_L$)
if $L_{min} < \boldsymbol{\Gamma}_{EB_i}.R^{bub} + \boldsymbol{O}_{i_L}.R^{bub}$ then
    $\boldsymbol{\Gamma}_{EB_i}.R^{bub} = L_{min} - \boldsymbol{O}_{i_L}.R^{bub} - safetyMargin$
    Check if radius is within the limits of $R_{min}$ and $R_{max}$
```

2.3.4 Adaptation of the EBM to Cylindrical Coordinates

As explained in Section 2.1, it is beneficial to represent the Remora in cylindrical coordinates, and consequently, it is necessary to adapt the 3D elastic band planner algorithm accordingly.

As the coordinate system is changed, one also need to change the path variables in $\boldsymbol{\Gamma}_{init} = \boldsymbol{\Gamma}_{EB} = [\alpha_v, D, \psi, R^{bub}]^\top$, where α_v, D and ψ are described in Section 2.1. The obstacles are also redefined to fit the cylindrical coordinate system as $\boldsymbol{O}_j = [\alpha_o, D_o, R^{Obst}]$.

The relation between cartesian coordinates and cylindrical coordinate is presented as [15]

$$\begin{aligned} N &= r_c(t)\cos(\alpha_v) \\ E &= r_c(t)\sin(\alpha_v) \\ D &= D\ , \end{aligned}$$

where α_v is within the range from $[-\pi, \pi]$ as shown in the left illustration in Figure 2.15. To avoid discontinuity at the edge cases, one should wrap the range to $[0, 2\pi]$ when $|\alpha_v| > \frac{\pi}{2}$. $r_c(t)$ is the radius of the cage at the position of the vehicle and is time-varying due to environmental forces deforming the structure of the net.

An essential part of the algorithm involves calculating the distance from either one bubble to another or one bubble to an obstacle. The length is evaluated against the radius of the bubbles, and therefore the distance should be calculated in equal units. Therefore one has to change the position of a bubble to the 2D plane by multiplying α_v with the cage radius $r_c(t)$. This is shown in Figure 2.15, where the position "x" on the net is projected onto a 2D plane.

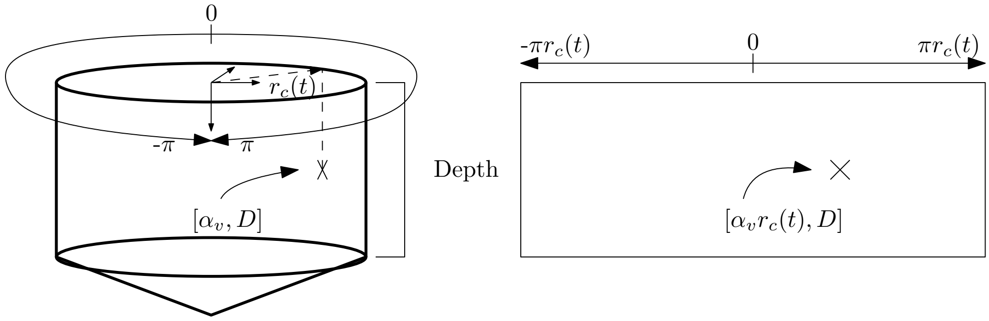


Figure 2.15: Cylindrical representation to 2D workspace.

Chapter 3

Implementation

This chapter describes how the algorithm was developed in MATLAB and later implemented in FhSim. It also describes the simulation setup, as well as the different case studies in detail.

3.1 Tools of Implementation

MATLAB was used as a development tool due to easy debugging and was a known programming environment. MATLAB was linked to FhSim through a TCP/IP connection to communicate with the Remora vehicle model and simulate while debugging the algorithm. The connection is visualised in Figure 3.1, where the motion planning box includes the EBM together with heading guidance to calculate the desired heading.

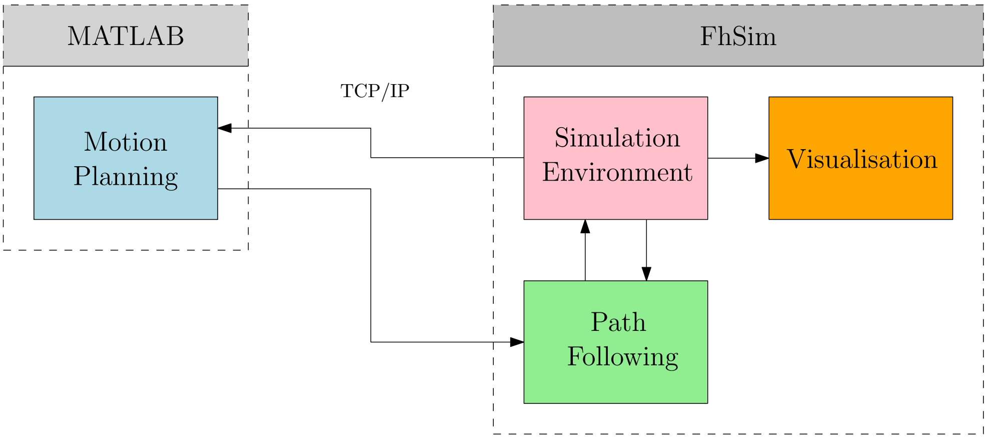


Figure 3.1: System description showing the connection between MATLAB and FhSim used in the devolopment phase. Image inspiration from [12].

Simulations conducted was made possible by the environment provided by SINTEF Ocean, called FhSim. *FhSim is a software platform and framework for mathematical modelling and numerical simulation, with a focus on marine applications* [43]. With its extensive model library, the specific mathematical models needed in this thesis, like a fish cage, are already developed and ready to use. Together with premade control systems and a vehicle model, a complete system can be implemented within reasonable time limits. The 3D visualisation makes it easy to analyse the simulation and improves one's understanding of the system and its behaviour. It can be connected with MATLAB and Simulink, but is implemented in C++ for efficiency [44]. Detailed information about FhSim's architecture and core functionalities can be found in [45]. A massive advantage of a realistic simulation environment is to avoid unnecessary problems in an expensive lab or field test by removing bugs and errors during extensive simulations.

Since FhSim is written in C++, it is beneficial to implement the algorithm in the same language. Besides, one source of error is eliminated as the TCP/IP communication is no longer needed.

3.2 Algorithm Development

First, the algorithm was implemented in a 2D environment solely in MATLAB to validate the functionalities of the algorithm. A differential drive robot [46] was simulated to prove that the robot was able to follow the predefined path. Together with a controller based on the pure pursuit guidance [47], a simple simulation inside a while-loop was conducted (see Figure 3.2) and verified in order to go another step.

The animation displays the method's properties quite precise, even when it is limited to two degrees of freedom. An initial path is made as a straight line from A to B without considering obstacles to avoid. When the simulation starts, the algorithm instantaneously calculates contraction and repulsive force and adapts the path away from the red circles. As the dynamic obstacle is closing, the path deformation becomes quite dramatic. The EBM finds a feasible path at last, but it was close to colliding with the dynamic obstacle. Under different circumstances, the dynamic obstacle could block the entire path between the two bigger obstacles. This would exploit the shortcoming of the algorithm discussed in the previous chapter, where a situation about a closing door was explained. In that case, the algorithm would not be able to recalculate a new feasible path on the right side of the medium red bubble.

Further, a 3D version of the EBM was implemented by adding the z-axis to the states. A simple animation illustrates one iteration of the EBM on startup in Figure 3.3, where one can see the different phases of the algorithm explained in Section 2.3.1. The first phase refers to the initial path, which is made before the

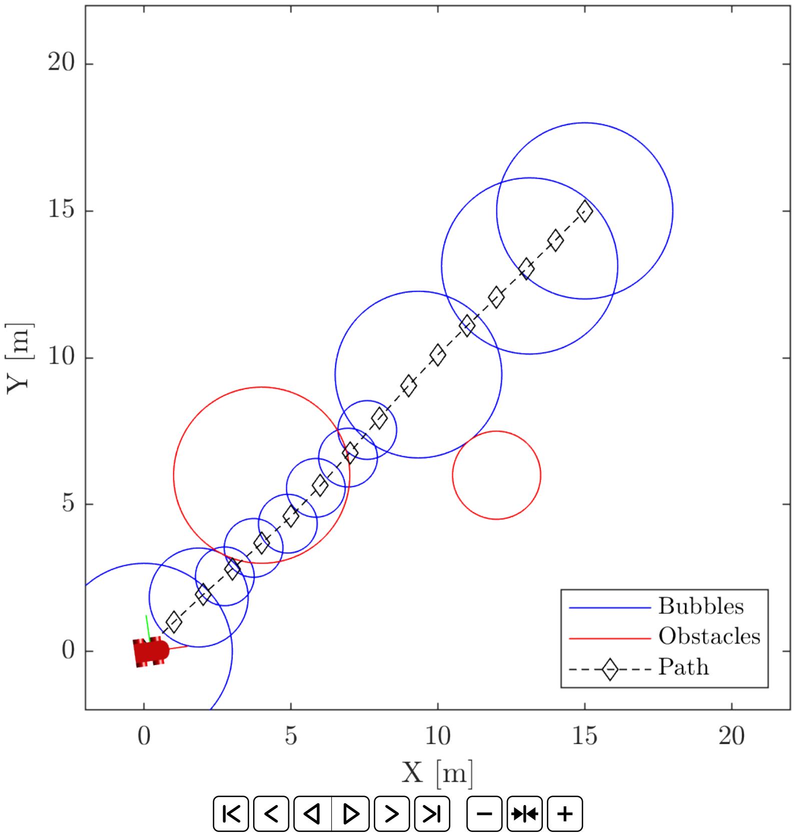


Figure 3.2: Animation of the EBM in 2D with static and dynamic obstacles. For static figures, see Figure A.1.

simulation starts. The second phase contains the deformation due to internal and external forces, and one can see the blue bubbles adapt. As the bubbles are moved around, they overlap or do not connect. This is where the third phase regarding bubble reorganisation makes a difference. The animation shows that some bubbles are deleted, while some are added and adapted to fit the environments. As the algorithm would have been called multiple times when a vehicle moves, the path would increasingly improve, as the case were in Figure 3.2.

Lastly, a 2D cylindrical version of the EBM was developed to match the frame of the Remora, as explained in Section 2.1. The states in the EBM were changed, verified in MATLAB before connection with FhSim was made for a more realistic simulation environment.

After verification in MATLAB, the algorithm was modified to C++ code by carefully changing the indexing and declaring correct variable types. A separate SimObject was made such that the algorithm is easy to use in other applications as well.

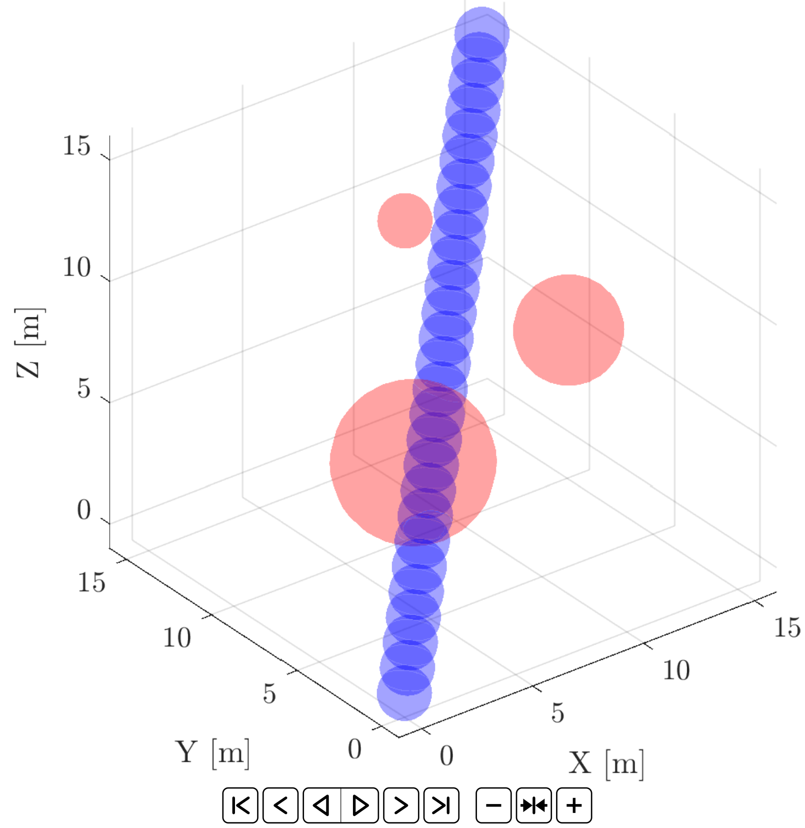


Figure 3.3: Animation of the EBM in 3D with static obstacles. For static figures, see Figure A.2.

3.3 ROV Model

The simulation experiments were designed initially for the Remora vehicle. However, since the tetherless design of the Remora is not yet fully tested and available, a similar set of simulations using an Remotely Operated Vehicle (ROV) was conducted. This was motivated by the possibilities of doing physical lab experiments to further verify the methods in the extension of the master project, where an ROV would be used. Furthermore, some of the current antifouling methods use an ROV in their cleaning procedures, like the AKVA FNC8 from AKVA group and the Stealth Cleaner from Ocein. More information regarding existing solutions is described in [12]. It is therefore highly relevant to evaluate the method for this vehicle as well.

The ROV used in a potential lab experiment is the BlueROV2 produced by BlueRobotics is displayed in Figure 3.4, and its dynamics are explained in the following section.

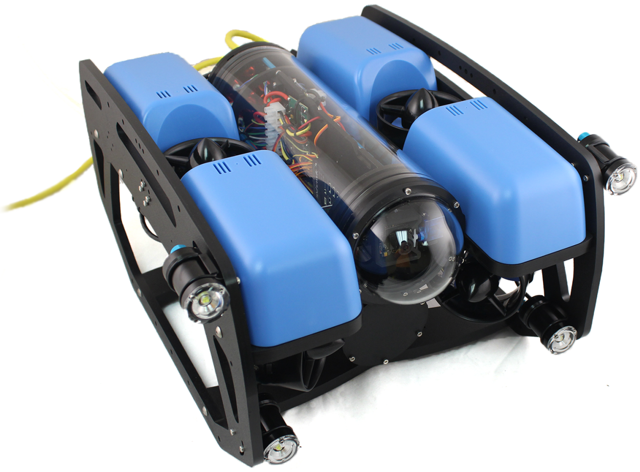


Figure 3.4: Image of the BlueROV2. Image courtesy of BlueRobotics [48].

This is a 4-DOF $[x, y, z, \psi]$ ROV with 6 thrusters in a vector configuration [48]. The model parameters are obtained from a similar vehicle, the BlueROV2 Heavy, produced by the same manufacturer. BlueROV2 Heavy is a 6-DOF ROV with eight thrusters but is considered sufficiently similar for the use of the model parameters for the BlueROV2.

The modelling identification is made in [49], while [50] provides the physical dimension on the thruster position as well as the weight of the BlueROV2. The derivation of the mathematical model has similarities to the derivation in Section 2.1 and can be viewed in detail in [49]. This chapter only displays the resulting model adapted to the BlueROV2 as

$$\begin{aligned}\dot{\boldsymbol{\eta}} &= \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} &= \boldsymbol{\tau}\ ,\end{aligned}$$

where $\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A$ and $\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_l + \boldsymbol{D}_n(\nu)$, and their values are

$$\boldsymbol{M}_{RB} = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & m & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & 0 & 0 & 0.16 \end{bmatrix}, \boldsymbol{M}_A = \begin{bmatrix} 5.5 & 0 & 0 & 0 \\ 0 & 12.7 & 0 & 0 \\ 0 & 0 & 14.5 & 0 \\ 0 & 0 & 0 & 0.12 \end{bmatrix},$$

$$\boldsymbol{D}_l = \begin{bmatrix} 4.03 & 0 & 0 & 0 \\ 0 & 6.22 & 0 & 0 \\ 0 & 0 & 5.18 & 0 \\ 0 & 0 & 0 & 0.07 \end{bmatrix}, \boldsymbol{D}_n(\nu) = \begin{bmatrix} 4.03 & 0 & 0 & 0 \\ 0 & 6.22 & 0 & 0 \\ 0 & 0 & 5.18 & 0 \\ 0 & 0 & 0 & 0.07 \end{bmatrix},$$

where $m = 11.167$.

Thruster Allocation for the ROV

Calculating the correct thruster allocation matrix is very important in a physical lab trial. In contrast to a numerical simulation, where one can input the wanted force in each degree of freedom, one must consider the thrusters' number, position, and attitude.

The thruster allocation problems are stated as [13]

$$\boldsymbol{\tau} = \boldsymbol{B}\boldsymbol{u} \; , \tag{3.1}$$

where $\boldsymbol{\tau}$ is the generalized control forces, $\boldsymbol{u}$ is the control inputs and $\boldsymbol{B}$ is the input matrix. To calculate the wanted input $\boldsymbol{u}$, one solve the equation above such that

$$\boldsymbol{u} = \boldsymbol{B}^{-1}\boldsymbol{\tau} \quad \text{or} \quad \boldsymbol{u} = \underbrace{\boldsymbol{B}^{\top}\left(\boldsymbol{B}\boldsymbol{B}^{\top}\right)^{-1}}_{\boldsymbol{B}^{\dagger}}\boldsymbol{\tau} \; . \tag{3.2}$$

The pseudoinverse $\boldsymbol{B}^{\dagger}$ is used when the number of actuators is higher than the number of DOF, which in this case is true since there are four DOF's and six thrusters.

Further the generalized forces are defined as [13]

$$\boldsymbol{\tau} = \sum_{i=1}^{r}\begin{bmatrix} \boldsymbol{F}_i^b \\ \boldsymbol{r}_{bp_i}^b \times \boldsymbol{F}_i^b \end{bmatrix} = \sum_{i=1}^{r}\begin{bmatrix} F_{x_i} \\ F_{y_i} \\ F_{z_i} \\ F_{z_i} l_{y_i} - F_{y_i} l_{z_i} \\ F_{x_i} l_{z_i} - F_{z_i} l_{x_i} \\ F_{y_i} l_{x_i} - F_{x_i} l_{y_i} \end{bmatrix} \; . \tag{3.3}$$

The control forces $\boldsymbol{f}$ can be written as a product of the control inputs $\boldsymbol{u}$ and a diagonal force coefficient matrix $\boldsymbol{K}$

$$\boldsymbol{f} = \boldsymbol{K}\boldsymbol{u} \; . \tag{3.4}$$

Furthermore $\boldsymbol{\tau}$ can also be defined as the product of a thrust configuration matrix $\boldsymbol{T}$ and the control forces $\boldsymbol{f}$ which yields

$$\boldsymbol{\tau} = \boldsymbol{T}\boldsymbol{f} \; . \tag{3.5}$$

Combining (3.1) and (3.4) gives

$$\boldsymbol{B} = \boldsymbol{T}\boldsymbol{K} \; , \tag{3.6}$$

where (3.6) is inserted in (3.2), and thus calculating correct input signals to the thrusters.

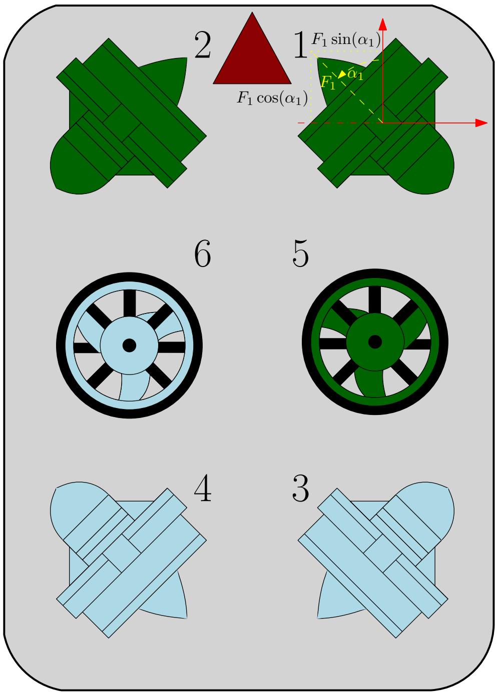


Figure 3.5: Physical layout of the thruster configuration on the BlueROV2. Since thruster 1, 2, 3 and 4 is angled with an azimuth angle α_i the force need to be decomposed in x and y direction, as illustrated for the first thruster. Image adapted from [52].

Table 3.1: Table of the physical position and orientation of the thrusters related to Figure 3.5.

Thruster (T_i)	l_{x_i}	l_{y_i}	l_{z_i}	$Rot_z(\alpha_i)$
T_1	0.12	0.11	0	$-\frac{\pi}{4}$
T_2	0.12	-0.11	0	$\frac{5\pi}{4}$
T_3	-0.12	0.11	0	$\frac{\pi}{4}$
T_4	-0.12	-0.11	0	$\frac{3\pi}{4}$
T_5	0	0.11	0	0
T_6	0	-0.11	0	0

The thrust configuration matrix $\boldsymbol{T}$ takes the physical configuration of the actuators into account. The position and attitude, displayed in Table 3.1, were obtained from the model used for simulating the ROV together with ROS in a Gazebo simulation [51]. The mounting position and orientation of the thrusters are displayed in Table 3.1 and correspond to the layout in Figure 3.5.

Applying (3.3) on each thruster T_i and filling in parameters from Table 3.1, the thrust configuration matrix $\boldsymbol{T}$ is calculated. Only the first and fifth thruster is derived to show the calculation for horizontal and vertical mounted thrusters as

$$\boldsymbol{\tau}_1 = \begin{bmatrix} F_{x_1} \\ F_{y_1} \\ F_{z_1} \\ F_{z_1} l_{y_1} - F_{y_1} l_{z_1} \\ F_{x_1} l_{z_1} - F_{z_1} l_{x_1} \\ F_{y_1} l_{x_1} - F_{x_1} l_{y_1} \end{bmatrix} = \begin{bmatrix} F_1 \cos(\alpha_1) \\ F_1 \sin(\alpha_1) \\ 0 \\ 0 \\ 0 \\ F_1 \sin(\alpha_1) l_{x_1} - F_1 \cos(\alpha_1) l_{y_1} \end{bmatrix},$$

$$\boldsymbol{\tau}_5 = \begin{bmatrix} F_{x_5} \\ F_{y_5} \\ F_{z_5} \\ F_{z_5} l_{y_5} - F_{y_5} l_{z_5} \\ F_{x_5} l_{z_5} - F_{z_5} l_{x_5} \\ F_{y_5} l_{x_5} - F_{x_5} l_{y_5} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ F_5 \\ F_5 l_{y_5} \\ -F_5 l_{x_5} \\ 0 \end{bmatrix}.$$

Solving for 6-DOF with (3.5) yields

$$\underbrace{\begin{bmatrix} X \\ Y \\ Z \\ K \\ M \\ N \end{bmatrix}}_{\tau} = \underbrace{\begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & 0 \\ -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & -0.11 & -0.11 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{\sqrt{2}}{200} & -\frac{\sqrt{2}}{200} & -\frac{\sqrt{2}}{200} & -\frac{\sqrt{2}}{200} & 0 & 0 \end{bmatrix}}_{\boldsymbol{T}(\alpha_i)} \underbrace{\begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \end{bmatrix}}_{f}. \tag{3.7}$$

Extracting the relevant DOF's from (3.7) and rewrite the result on the form $\boldsymbol{\tau} = \boldsymbol{TKu}$ from (3.4) and (3.5), and we obtain a non-square thrust configuration matrix, hence the pseudoinverse needs to be used when calculating the wanted input $\boldsymbol{u}$.

$$\underbrace{\begin{bmatrix} X \\ Y \\ Z \\ N \end{bmatrix}}_{\tau} = \underbrace{\begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & 0 \\ -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ -\frac{\sqrt{2}}{200} & -\frac{\sqrt{2}}{200} & -\frac{\sqrt{2}}{200} & -\frac{\sqrt{2}}{200} & 0 & 0 \end{bmatrix}}_{\boldsymbol{T}(\alpha_i)} \underbrace{\begin{bmatrix} K_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & K_6 \end{bmatrix}}_{\boldsymbol{K}} \underbrace{\begin{bmatrix} u_1 \\ \vdots \\ u_6 \end{bmatrix}}_{\boldsymbol{u}}$$

$$\boldsymbol{\tau} = \boldsymbol{TKu}$$

$$\text{solving for } \boldsymbol{u}\text{:} \quad \boldsymbol{u} = \boldsymbol{K}^{-1} \boldsymbol{T}^{\dagger} \boldsymbol{\tau}$$

The force coefficient matrix $\boldsymbol{K}$ is obtained from a linear interpolation of the force/Pulsewidth Modulation (PWM) graph provided by the manufacturer of the T200 thrusters used on the BlueROV2 [53]. Linear interpolation is used to create a more precise estimate based on a set of values, and is calculated by

$$y = \frac{y_0(x_1 - x) + y_1(x - x_0)}{x_1 - x_0},$$

where x_0 and x_1 is the closest values of force to the desired force x, and y_0 and y_1 is the corresponding PWM values to x_0 and x_1. y is the desired PWM signal that represents the desired force. This is calculated for each thruster and transforms the calculated force to PWM signal to be sent to the thrusters.

3.4 Simulation Setup

Through FhSim the simulation was conducted, and Figure 3.6 illustrates the graphical feedback from the simulation, here presented by the Remora vehicle model.

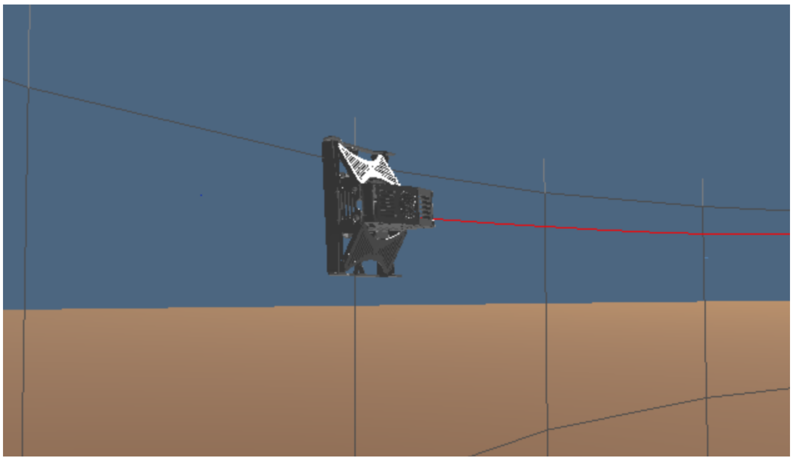


Figure 3.6: Screenshot of the Remora vehicle model in FhSim.

A net pen with a radius of 25 meters and a depth of 15 meters was used as a framework for the workspace of the vehicles. In order to adapt the EBM to the Remora simulation, some limitations have been added to avoid the path surpassing the physical limits of the workspace of the robot. It can not move above the surface nor through the bottom of the net. Therefore the depth in the new position, $\boldsymbol{P}^{bi}_{new}$, is checked with the physical size of the cage.

Note also that in aquaculture applications, all obstacles creating a region of avoidance are considered slowly moving. In the following experiments, the assumption that $V^{obs}_{bub} = 0$ will therefore be made. This affects the calculated safe distance, affecting the fading function when calculating external repulsion force in Section 2.3.1.

Two different initial paths were made that fits on the net to evaluate the performance of the EBM in different situations. One path was created solely for testing purposes, while the other was made as a realistic cleaning routine of the whole net. They are both illustrated in Figure 3.7, where the testing path traverse half the net before returning along the bottom of the net. The cleaning path uses five rounds to reach the bottom of the net. This would not be enough for a physical implementation since the vehicle is not wide enough to cover the area between the paths, but it still shows the concept of this type of path. Note that the lowest point of the cleaning path surpasses the lowest point of the fish cage. This is done to verify the EBM ability to return a feasible path.

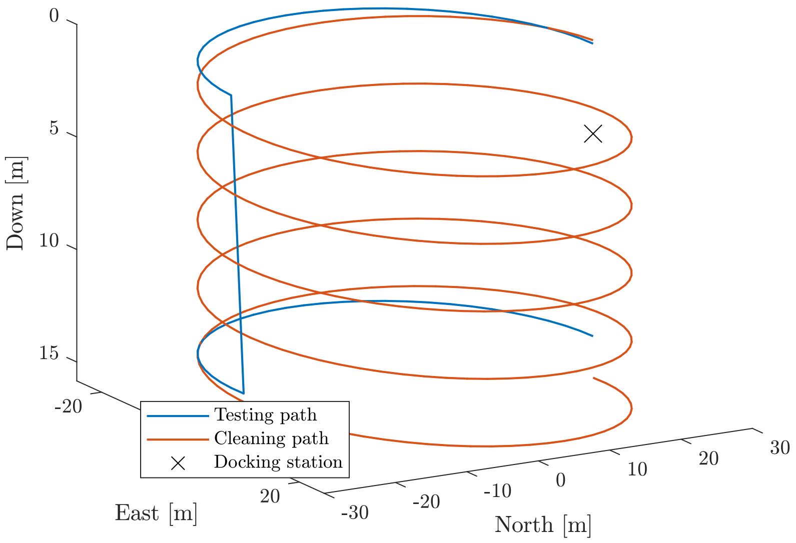


Figure 3.7: Two initial paths were used during the simulations. A docking station was implemented 5 meters below the surface.

The cleaning path is highly unusual to date due to current methods use a non-tetherless ROV, which would tangle itself in the fish cage, and the tether would be of interference with the fish. However, the tether is not accounted for in the simulations, and therefore the test is considered informative.

During the simulations, both static and dynamic regions of avoidance will be added to the environment to increase the difficulty of the operation. As will ocean current, both features are explained in detail in Section 3.5

3.4.1 Main Program

The main program for simulating the elastic band planner is based on a simple state machine with four states, where the user chooses the initial path in advance. The different states are defined as:

1. Start: Adds the starting position of the vehicle at the beginning of the predefined path $\mathbf{\Gamma}_{init}$.
2. Move: Applies the EBM to the path $\mathbf{\Gamma}_{EB}$, and outputs information to the heading guidance and the desired surge speed.
3. Dock: Inserts the predefined docking procedure to the elastic band.
4. Stop: Sets the desired speed to zero and recharges.

Figure 3.8 visualise the simplified state machine used for the simulations with the Remora vehicle. The state machine used for the BlueROV2 has many similarities and use the same four states but have some minor deviations. These are considered neglectable, and Figure 3.8 is sufficient for illustrating how the simulation was conducted for the ROV as well.

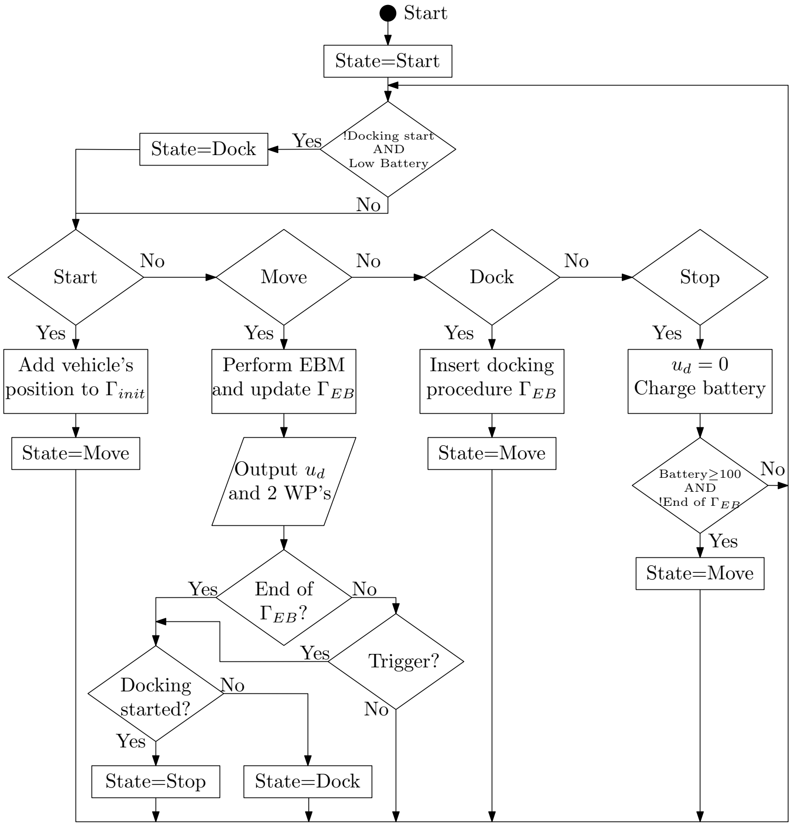


Figure 3.8: Simplified state machine for the simulations of the Remora.

After the position of the vehicle is added to $\mathbf{\Gamma}_{init}$ in the *Start* state, the state is changed to *Move*. Here the EBM adapts the initial path and updates $\mathbf{\Gamma}_{EB}$, to be used as a guidance for speed and outputs two points to be used in the heading guidance. This is repeated until an interrupt is triggered. A trigger might be that the vehicle has completed the initial path, low battery or a hole in the net is detected that requires the vehicle to return to the docking station. *Dock* state inserts the docking procedure in the current path such that it can return to the trigger point after the battery is recharged before it returns to Move state. When the vehicle arrives at the docking station, the state is changed to *Stop*, and information can be sent to a topside computer and the battery charged.

Docking Procedure

A separate docking procedure was implemented such that the vehicle approaches the docking station from the correct position and attitude. A virtual docking station is placed 5 meters below the surface. There are added some waypoints below and above the docking station to ensure that the robot approaches the underside and departs the docking station by moving upwards. As the physical docking design and constraints are unknown, this is considered a realistic and sufficient solution. The docking procedure is used either at the end of a cleaning routine or by the trigger of an event, e.g. low battery. In addition, the path is initialised with a smaller bubble size, inducing a slower speed of the vehicle during docking and undocking, achieving high precision, hence reducing the danger of colliding.

3.4.2 Control System, Heading Guidance and Reference Models

During the simulations, the control system will also be evaluated. A feedback linearisation controller is mainly used to control the surge speed and heading angle, but a simpler PID and PI controller will also be implemented and evaluated. These are already a part of the FhSim library and was ready to use after tuning. By trial and error, the gains in Table 3.2 were found to be sufficient.

Table 3.2: Controller parameters for Remora.

Feedback linearisation					PI		PID		
Surge		Yaw			Surge		Yaw		
K_p	K_i	K_p	K_i	K_d	K_p	K_i	K_p	K_i	K_d
4	4	5	1	5	20	5	10	0.1	5

To complete the system, a LOS heading guidance was used. The response created by the lookahead distance, Δ, will be evaluated, and a value that provided the wanted behaviour was found to be 3. The two first bubbles provide the points used for guidance in the elastic band, $\mathbf{\Gamma}_{EB}$, where a switching mechanism, circle of acceptance, is needed for selecting the next waypoints [13]. By checking if the robot is within the radius of the second bubble, i.e. by checking if (3.8) holds, the first bubble is deleted, and the next WP's are updated and sent to the heading guidance. For the BlueROV2, the 3D version of the EBM is used, and consequently, the circle of acceptance is updated to a sphere of acceptance (3.9).

$$\sqrt{\left(\alpha_v r_c(t) - \alpha_{\Gamma_{EB_1}} r_c(t)\right)^2 + \left(z_v - D_{\Gamma_{EB_1}}\right)^2} < R^{bub}_{\Gamma_{EB_1}} , \tag{3.8}$$

$$\sqrt{\left(x_v - x_{\Gamma_{EB_1}}\right)^2 + \left(y_v - y_{\Gamma_{EB_1}}\right)^2 + \left(z_v - z_{\Gamma_{EB_1}}\right)^2} < R^{bub}_{\Gamma_{EB_1}} , \tag{3.9}$$

where α_v is the azimuth angle of the position of the vehicle in cylindrical coordinates, $r_c(t)$ is the time varying cageradius, $\alpha_{\Gamma_{EB_1}}$, $D_{\Gamma_{EB_1}}$ and $R^{bub}_{\Gamma_{EB_1}}$ is the azimuth angle, the depth and the radius respectively of index number one in the elastic band. Further, are x_v, y_v and z_v the vehicles position in cartesian coordinates, and $x_{\Gamma_{EB_1}}$, $y_{\Gamma_{EB_1}}$ and $z_{\Gamma_{EB_1}}$ the position of index number one in the elastic band.

A reference model was implemented to smooth the desired states before. This should compensate for neglecting the fourth phase of the EBM, described in Section 2.3.1. The following values for the reference model were found to be adequate

$$\boldsymbol{A}_{d_u} = \begin{bmatrix} 0 & 1 \\ -100 & -20 \end{bmatrix}, \boldsymbol{B}_{d_u} = \begin{bmatrix} 0 \\ 100 \end{bmatrix}, \tag{3.10}$$

$$\boldsymbol{A}_{d_\psi} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -8 & -12 & -6 \end{bmatrix}, \boldsymbol{B}_{d_\psi} = \begin{bmatrix} 0 \\ 0 \\ 8 \end{bmatrix}, \tag{3.11}$$

where (3.10) is the second order reference model for the surge speed, and (3.11) is the third order reference model for the yaw angle.

3.4.3 BlueROV2

This section will briefly introduce the elements used to do simulations of the BlueROV2 in FhSim with the elastic band planner.

The ROV has four degrees of freedom in $[x, y, z, \psi]$, and four separate controllers are introduced to control each freedom separately. A feedback linearisation controller is used for the first three states, while a PD controller is used to control the heading. The BlueROV2 vehicle is relatively well known and has been used by [50, 54, 55], causing the dynamics and parameters of the vehicle to be easy to obtain, allowing using highly advanced controllers that require extensive knowledge about the vehicle. A second-order reference model is used in advance of all controllers to smooth the desired state.

By trial and error, the parameters in Table 3.3 were found for the controllers and (3.12)-(3.13) for the reference models.

Table 3.3: Controller parameters for BlueROV2.

Feedback linearisation						PD	
Surge		Sway		Heave		Yaw	
K_p	K_i	K_p	K_i	K_p	K_i	K_p	K_d
2	1	7.5	2.5	5	1	1	1

$$\boldsymbol{A}_{d_{uvw}} = \begin{bmatrix} 0 & 1 \\ -50 & -20 \end{bmatrix}, \boldsymbol{B}_{d_{uvw}} = \begin{bmatrix} 0 \\ 50 \end{bmatrix}, \tag{3.12}$$

$$\boldsymbol{A}_{d_\psi} = \begin{bmatrix} 0 & 1 \\ -1.56 & -1.88 \end{bmatrix}, \boldsymbol{B}_{d_\psi} = \begin{bmatrix} 0 \\ 1.56 \end{bmatrix}, \tag{3.13}$$

where (3.12) is the reference model used for the controller in surge, sway and heave, while (3.13) was used for the controller of the heading angle.

Traditional net cleaning methods use a vehicle that always has the front of the vehicle towards the net if the cleaning jig is mounted at the front. This behaviour has been implemented in FhSim by introducing the following equations provided by SINTEF Ocean, such that the heading of the vehicle always stays normal to the face of the net.

$$\begin{aligned} \psi_d &= \pi_p + \arctan\left(\frac{y_e^p}{\Delta}\right) \pm \frac{\pi}{2} \\ u_d &= R_{\boldsymbol{\Gamma}_{EB_1}}^{bub} \cos\left(-\psi + \pi_p - \arctan\left(\frac{y_e^p}{\Delta}\right)\right) \\ v_d &= R_{\boldsymbol{\Gamma}_{EB_1}}^{bub} \sin\left(-\psi + \pi_p - \arctan\left(\frac{y_e^p}{\Delta}\right)\right) \end{aligned}$$

Where ψ_d, u_d and v_d is the desired heading, surge speed and sway speed, respectively. π_p is the path tangential angle, y_e^p the cross-track error, and Δ is the lookahead distance, all related to the LOS guidance. $\pm\frac{\pi}{2}$ are added dependent on in which direction the vehicle moves. Moreover, $R_{\boldsymbol{\Gamma}_{EB_1}}^{bub}$ is the radius of the first bubble in the elastic band, and ψ is the actual heading of the vehicle. Figure 3.9 shows the model of the BlueROV2 in FhSim with net following activated.

SINTEF Ocean has also developed an algorithm that uses a Doppler Velocity Log (DVL) sensor to approximate a region of the net in front of an ROV and control the position and attitude of the ROV to always be normal to the face of the net. This would make the system robust, realistic and applicable for an aquaculture application. However, since it is beyond the scope of this thesis, the DVL version will be replaced by the simplified version above that does not consider any deformations of the net. Therefore the simulations for the ROV will be conducted without fish cage deformation, but ocean currents will still be implemented.

The 3D version of the EBM was used since the vehicle has enough DOF to move in the three-dimensional space. However, one might prefer that the vehicle is close to

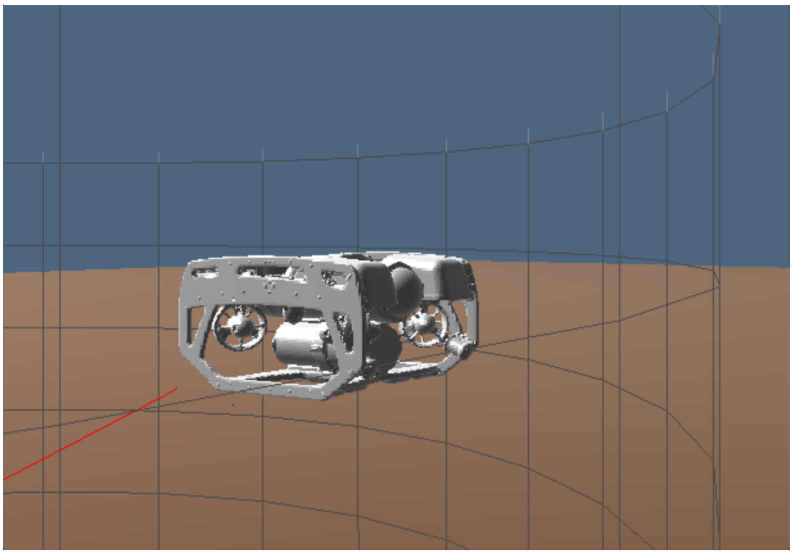


Figure 3.9: Screenshot of the BlueROV2 model in FhSim with net following activated.

the net as much as possible, and there is therefore implemented an option in the algorithm where one can choose in what direction the deformation forces should work. E.g. only in the z-direction to keep the vehicle close to the net. One can also choose to allow some deformation in x- and y-direction. For the most part of the simulation, the only deformation in the z-axis is activated. However, the difference is evaluated in Section 4.2.3.

This feature was added subsequently and only relevant for the ROV, and therefore excluded from the explanation of the EBM in Section 2.3.1, aimed at the Remora vehicle. However, the modification is pretty straightforward as (2.15) is now written as

$$\boldsymbol{F}_{net}^{bi} = \begin{bmatrix} d_x & 0 & 0 \\ 0 & d_y & 0 \\ 0 & 0 & d_z \end{bmatrix} \left[\alpha \boldsymbol{F}_{int}^{bi} + \beta \boldsymbol{F}_{ext}^{bi} \right],$$

where $\left[\alpha \boldsymbol{F}_{int}^{bi} + \beta \boldsymbol{F}_{ext}^{bi} \right]$ is a 3x1 vector, and d_x, d_y and d_z is the deformation factor in each direction. In the case of deformation only in z-direction, $d_x = 0$, $d_y = 0$ and $d_z = 1$ is set.

Note that while the net pen radius is 25 meters, the radius of the initial paths is set to 24 metres to keep the ROV with a distance from the net, displayed in Figure 3.9.

3.5 Simulation Experiments

The simulations were conducted with the idea of starting with the ideal case of no environmental disturbance before adding regions of avoidance and ocean currents to deform the structure of the fish cage, increasing the level of complexity step by step. At last, a simulation of a cleaning routine interrupted by a docking trigger, where the robot should find its way back to the docking station, charge up, send information before continuing the cleaning path.

1. Simple cleaning routine to prove performance under ideal conditions
2. Add static region of avoidance.
3. Add dynamic region of avoidance.
4. Add ocean currents.
   a. Without structural deformation of the net to evaluate the controllers.
   b. With structural deformation of the net to evaluate the motion planner.
5. Add static and dynamic region of avoidance with ocean current.
6. Perform a cleaning routine where a docking procedure is triggered (low battery/found hole), docking procedure performed, before continuing.

All simulation experiments were conducted with the Remora vehicle, while case 1-4a and 6 was conducted for the BlueROV2. The following sections describe the simulation cases in details and how the Remora and BlueROV2 simulations differ.

Before the simulation studies were conducted, the parameters in the EBM and LOS heading guidance were adjusted towards the wanted behaviour by evaluating different parameters.

3.5.1 Parameterisation of the EBM

This test will tune the algorithm to be optimised towards a net crawling vehicle in aquaculture environments. The following three requirements are considered important:

1. The resulting path must be smooth enough such that the vehicle is capable of following the path.
2. The resulting path must avoid any regions of avoidance.
3. The resulting path should strive to follow the initial path while maintaining the criteria above.

The most essential variables which have the most impact on the behaviour is the four parameters k_{int}, k_{ext}, α and β, which represent the contraction gain, repulsive gain, weighting of internal force and weighting of external force respectively.

These parameters are elaborated thoroughly in Section 2.3.1, but the effect will be demonstrated in Section 4.1.1.

Six different simulations were completed with different configuration to visualise the effect the changes has. The six configurations is defined in Table 3.4. Another important parameter to evaluate is the cross-track error from the LOS heading guidance. As explained in Section 2.2.2, the cross-track error is a measurement of how far away from the desired path the vehicle's position is, from which the desired heading is calculated, together with the lookahead distance Δ. A low value for Δ yields a quick response, while a high value yields a slower response.

Table 3.4: Parameter configurations for the EBM.

Configuration #	k_{int}	k_{ext}	α	β
1	0.5	0.5	0.5	0.5
2	0.05	0.5	0.5	0.5
3	0.05	0.5	0.1	0.5
4	0.05	5	0.1	0.5
5	0.05	5	0.1	0.01
Final	0.1	3	0.25	0.75

Four different configurations for Δ was evaluated: 1, 2, 3 and 5. One simulation was conducted for each configuration of the lookahead distance parameter.

Similar values obtained for the Remora vehicle were also used for the BlueROV2 and verified with the same setup as the values were obtained.

3.5.2 Case 1: Five-round Cleaning Routine under Ideal Conditions

The initial cleaning path was chosen, while there were no regions of avoidance in the workspace nor any ocean currents activated. This simulation should test the system's capabilities to perform a relatively long and realistic cleaning routine under ideal conditions, making sure the system is ready for a higher degree of difficulty.

The same test was conducted with the BlueROV2 without any modifications required.

3.5.3 Case 2: Static Regions of Avoidance

While using the initial testing path, two static regions of avoidance with different size and position were added to test the ability of the EBM to find a new feasible path. No ocean current was applied. During this test, the thruster commands from the controllers were evaluated to monitor the range of wanted force and see how the two belts work together.

BlueROV2 Modifications

The 3D version of the EBM has an additional tuning parameter, where one can choose the direction of adaptation. Two simulations were conducted to illustrate the effect. The first with only deformation in the z-direction, and the second with deformation in x-, y-, and z-direction. However, for the rest of the simulation, a deformation only in the z-direction will be allowed.

The output from the controllers was not evaluated in this case.

3.5.4 Case 3: Dynamic Region of Avoidance

Using the initial testing path and introducing a velocity to one of the obstacles, the algorithm was challenged with both moving and stationary avoidance regions. A velocity of 0.25 m/s directed towards the Remora was added to the obstacle. This velocity is considered high in the case that most relevant obstacles move very slowly in water.

The same test was conducted with the BlueROV2 without any modifications required.

3.5.5 Case 4: Ocean Current

The FhSim framework allows for easy manipulation of environmental forces, like ocean current. A series of variations regarding the current speed and direction was made to illustrate its effect on the vehicle and how the vehicle copes with the challenge ocean currents add to the existing problem. This was simulated with the testing path, without any regions of avoidance.

Two configurations of control systems were tested to evaluate the performance. First, the feedback linearisation controllers were implemented. This controller requires perfect knowledge about vehicle dynamics, which are challenging to find in many cases. Therefore a PID and PI controller for heading and surge respectively are also evaluated. The gains used can be found in Table 3.2.

A speed of 1 m/s is considered a large current that rarely occurs at locations in Norwegian fjords but is a way of demonstrating the effect on the vehicle and how the controllers react. On the other hand, exposed locations are more and more common, where the conditions are more demanding, the forces more significant, and the currents higher. The magnitude of the ocean current may therefore be interesting to use in simulations.

Two simulations with ocean current were conducted. During the first simulation, the ocean current *did not* deform the structure of the fish cage. Only the vehicle is under the influence of environmental forces. This was done to evaluate the controllers under known environments. The effect of the environmental force is most evident in the surge direction and will be evaluated. The direction of the current in the different simulations are described in Table 3.5. Note that the current is uniform over the whole water column for simplicity.

The second simulation will include the net deformation, complicating the workspace as the ocean current highly influences the net structure.

Table 3.5: Table of configuration parameters for simulation with ocean current with different angles.

Configuration #	u_c [m/s]	α_c [deg]
1	0	0
2	1	0
3	1	45
4	1	90

BlueROV2 Modifications

The models used for simulating the Remora and the BlueROV2 has different properties and functionalities. Among these are other ways to implement disturbance in the sense of ocean currents. While the Remora use the ocean current speed and direction, the BlueROV2 use the force produced by the ocean current. As the vehicles, dynamics and physical properties are different as well, it is hard to compare the behaviour with the exact ocean current. Nonetheless, a force is added with an angle of 45 degrees, and what approximately will be 1 m/s is implemented to evaluate the performance of the controllers. One simulation with and one without ocean forces was conducted for comparison.

Note that since the BlueROV2 is not attached to the net, the net deformation is hard to obtain, hence turned off for all simulations with the ROV.

In addition, alternative controllers evaluated for the BlueROV2 would most likely produce the same results as for the Remora. Hence only the feedback linearisation controllers will be assessed.

3.5.6 Case 5: Ocean Current with Static and Dynamic Region of Avoidance

This simulation combines the previous elements to provide a more complex simulation environment where both static and dynamic obstacles are introduced together with an ocean current. The ocean current's speed is 0.5 m/s with an angle of 45 degrees, similar to configuration number three in Table 3.5, with a difference in the magnitude of the current. Further, the same obstacles as introduced in Case 3 was used.

For a more realistic simulation case, the obstacles are assumed to be attached to the net, but since the radius of the cage varies with ocean current, the position of the obstacles needs to be adjusted. The best assumption is that the obstacle is the same length from the z-axis through origo as the robot. This might get inaccurate when the robot and obstacle are far away from each other but will be compensated for when the distance decrease and the solution is considered sufficient. The alternative would be to define obstacles as separate objects in FhSim, which would be attached to the net and pass them as input to the EBM, but for simplicity reasons, it was not implemented.

Since the BlueROV2 can not crawl on the net, this simulation case study was not conducted for this vehicle type.

3.5.7 Case 6: Five-round Cleaning Routine under Non-Ideal Conditions

For simulating a realistic scenario, the initial cleaning path was used together with the docking procedure triggered before the end of the cleaning routine. In this case, a low battery trigger occurs if the battery level drops under 20%.

To simulate the battery level, an equation that describes the energy consumption was used

$$V_{batt} = V_{batt} - \frac{\|\boldsymbol{\nu}_r\|}{a} - b \ ,$$

where a is a constant that defines the energy consumption related to moving the vehicle, and b is a constant consumption related to powering the robot independent of moving. These are set by trial and error to achieve the goal of reaching the low battery trigger before the vehicle is done with the cleaning routine, forcing the robot to initialise the docking procedure. $\|\boldsymbol{\nu}_r\|$ is the norm of the relative velocity of the vehicle.

Two simulations were conducted: One without ocean current and two static regions of avoidance, the other with an ocean current configuration equal to Case 5 and one static and dynamic region of avoidance.

BlueROV2 Modifications

One main difference from the Remora vehicle is that the BlueROV2 has four degrees of freedom. This lets the vehicle find the shortest path from one point to another without the constraint of being attached to the net. Particularly useful when the battery is low, and the vehicle needs to go to the docking station before the battery runs out. During docking, the heading of the ROV is in the direction it is going and not normal to the net. I.e. the net following is turned off during docking.

3.5.8 Time Complexity

By using the `chrono` library in C++ to access the `high_resolution_clock`, the elapsed time for one iteration of the EBM was calculated by reading the time before and after the function call. As the length of the EBM, essentially the number of points, is the main factor for time consumption, the length is evaluated against time. Another factor is regions of avoidance that affect the initial path, forcing the external forces and new radius to be calculated.

The computer used for all simulations has an Intel(R) Core(TM) i7-7600U CPU 2.80GHz with 16GB of RAM, running a 64-bit Windows 10 Enterprise.

The initial testing path was used during the first part of this test. The simulation was conducted in three parts. The first without any regions of avoidance, the second with two static, and lastly, one dynamic and one static region of avoidance was added to the environment. All simulations were conducted on the Remora without ocean currents, as this would not impact the time consumption.

As the initial path is relatively short for a practical application, a longer simulation was conducted in the second part of this test. The cleaning path was used, and the simulation contains a cleaning routine with the current, static region of avoidance, dynamic region of avoidance and a trigger action invoking the docking procedure, equal to Case 6.

Chapter 4

Results

This chapter demonstrates both the Remora and the BlueROV2 vehicle performance while being subject to various case studies. All results are obtained from simulation in FhSim. The first results presented are based on the model of the Remora vehicle.

4.1 Simulations of the Remora

Figure 4.1 shows the FhSim simulation running and visualising the Remora robot inside a net pen in a realistic aquaculture environment. The red line is the preceding path, and the two blue bubbles are the points used for heading guidance.

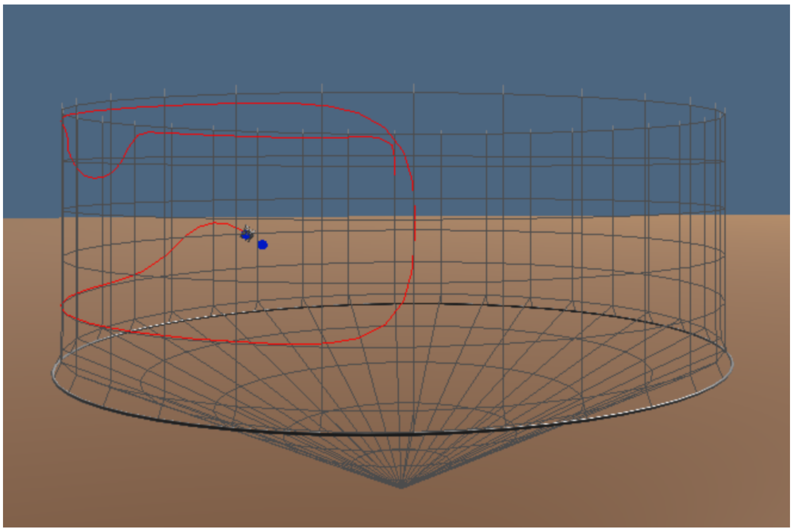


Figure 4.1: Screenshot from a simulation in FhSim.

The following section will evaluate the EBM and the LOS heading guidance with different configurations in different settings, but some parameters are set constant throughout the simulations. γ is the step size of the simulation, i.e. the step size of the integration done in simulation; in our case, this is 0.01. $deformTol$ is set to 2 and sets a maximum limit of deformation in each iteration. The desired overlap, d_{ol} is 0.25, while the $spinConst$ is set to 0.1. This prevents the algorithm from improving the solution if the change is less than 0.1. Further explanations can be found in Section 2.3.1. Other parameters need to be calibrated based on the vehicle type used and objective.

4.1.1 Parameterization of the EBM for the Remora

The configuration number relates Table 3.4 with the number on the legend in Figure 4.2. As a starting point all parameters, k_{int}, k_{ext}, α and β, are set to the same value of 0.5. From this baseline each value was altered to show the effect.

From configuration number 1 to 2, the contraction gain k_{int} is reduced by a factor of 10, resulting in a more strict behaviour such that the resulting path follows the predefined path more closely, especially in the turn when the robot heads back, but also when the vehicle avoids the region of avoidance and returns more quickly to the initial path. Also, since the repulsive gain contribution increases relative to the contraction gain, the resulting path evades the region of avoidance completely.

In configuration 3 the weighting of internal force is reduced from 0.5 to 0.1 while maintaining the contraction gain from configuration 1. This will further reduce the smoothness of the path and amplify the repulsive force from the regions of avoidance. Remember that the reference model will also contribute to smooth the resulting path.

Configuration 4 evaluates the repulsive gain by increasing k_{ext} by a factor of 10. Figure 4.2 shows that the difference between configuration 4 and 3 is minimal but can be seen around the obstacles, where the green line is slightly further away, as one should expect. The reason for the slight deviation is due to that the algorithm improves incrementally, with relatively small steps. Hence, the resulting change at each iteration becomes relatively small despite the sizeable repulsive gain.

The last configuration, number 5, intends to show the effect of the weighting of external force by decreasing the value from 0.5 to 0.01. This should reduce the effect the repulsive gain has on the system and essentially reduce the reaction to regions of avoidance. This can be seen in Figure 4.2 by the blue line, which does not avoid the obstacles but follow the predefined path quite precise.

By further trial and error the final parameters in Table 3.4 were considered sufficient for fulfilling the requirements, and the resulting path is plotted in Figure 4.2.

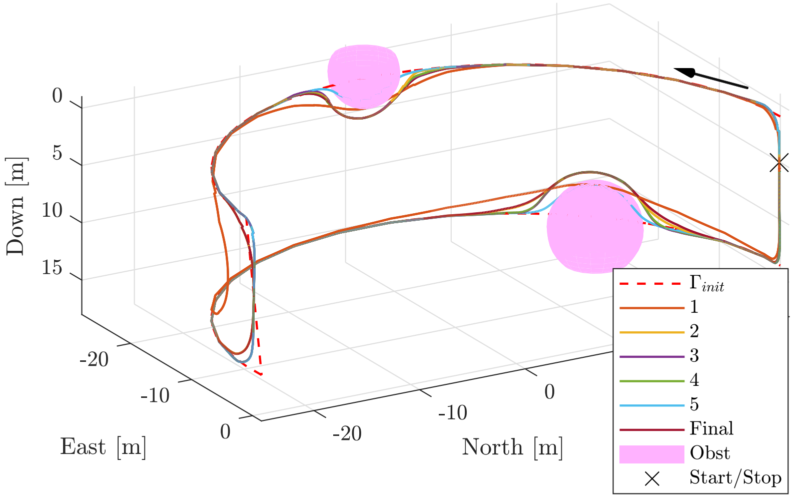
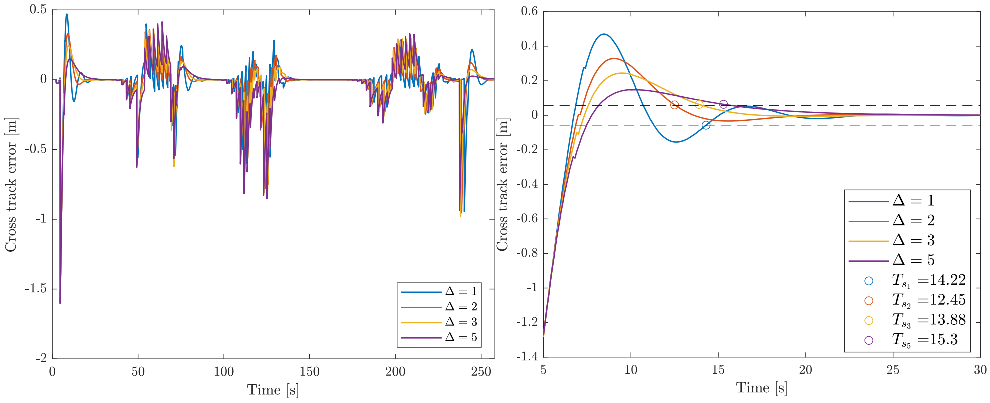


Figure 4.2: Different configurations of the algorithm yields different resulting behaviour. The dotted red line Γ_{init} represent the initial path set pre-fligth.

The response of the crosstrack error with different lookahead distances is visualised in Figure 4.3, where the response of the whole simulation (Figure 4.3a), together with an excerpt (Figure 4.3b) is showed.



(a) Overview of the cross track error.

(b) Zoomed view of the cross track error.

Figure 4.3: The cross track error with different configurations of the lookahead distance Δ.

In Figure 4.3 one can see that the largest cross-track error is approximately -1.5 and happens during the start of the simulation. Furthermore, the error is most significant during turning but converges quite nicely afterwards. The differences are best viewed in Figure 4.3b, where the dotted lines represent an error of ±5%. Furthermore are the settling time, T_s, stated in the legend for each value of Δ with corresponding coloured circles marking the settling time. One can see that the blue line has a quick response, but also some overshoot, while the purple and yellow line has a more smooth behaviour, and the overshoot is minimal. From the circles and the legend, one can see that the red line has the fastest settling time, but it also has some overshoot. A lookahead distance of 3 is considered suitable for this application due to a fast response while avoiding substantial overshoots, resulting in a sufficiently fast convergence towards the desired path.

4.1.2 Case 1: Five-round Cleaning Routine under Ideal Conditions

Figure 4.4 depicts a five-round cleaning routine with the same start and stop position. One can see that the vehicle follows the path quite precise, with a little overshoot at the first turn at the top of the net.

Furthermore, one can see a deviation towards the end of the cleaning routine before the vehicle turns and ascends towards the docking station. This is due to the imprecise initial path that exceeds the boundaries of the physical cage. Note also that the first iteration of the EBM, Γ_{EB_1}, adapts the path such that it is in the feasible region. This can be seen by the dotted line above the initial path at the turn before docking.

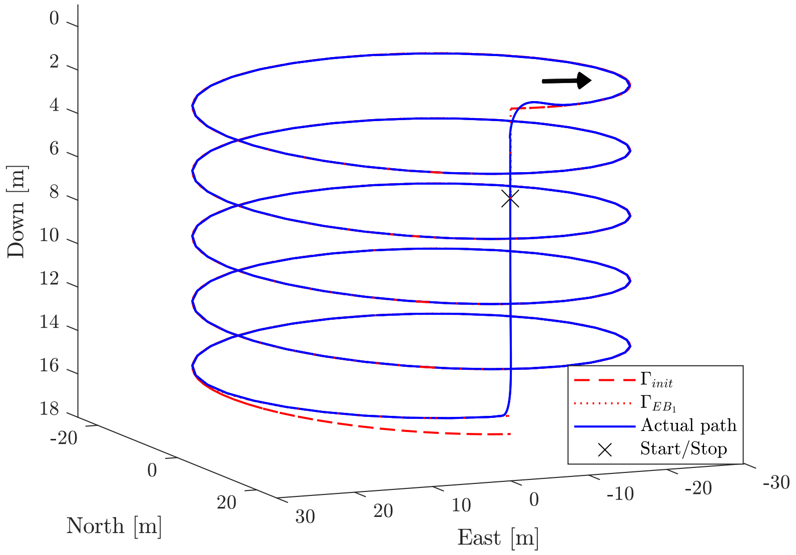


Figure 4.4: A five-round cleaning routine under ideal conditions.

4.1.3 Case 2: Static Regions of Avoidance

The resulting path, gained from the parameterization study, are plotted separately in Figure 4.5, to be used in this study. The vehicle avoids the static regions of avoidance, while maintaining a sufficient distance to the initial path, $\mathbf{\Gamma}_{init}$.

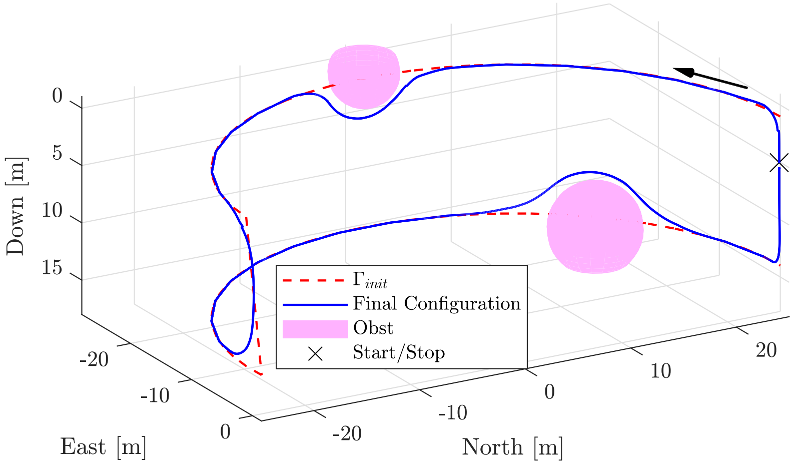
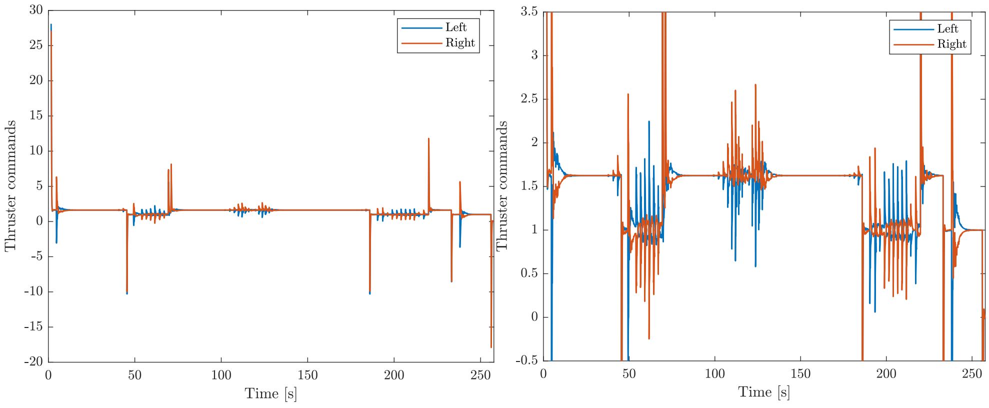


Figure 4.5: The final response of the configuration to be used for the rest of the report.

From Figure 4.6 one can see how the thruster commands evolve when the vehicle is moving in Figure 4.5. First, both belts work together to accelerate the vehicle from stationary to desired speed. Then the vehicle turns to the left to follow the path, and one can see the thruster command is positive on the right belt and negative on the left. They work in the opposite direction to maximize the turn. The difference is more evident when the vehicle avoids the first region of avoidance after approximately 50 seconds. The vehicle turns left, then right, and then left again after evading the obstacle. Figure 4.6b shows how the left and right belts work opposite of each other, one increase and the other decrease, for maximum response in turn.



(a) Overview of the thruster commands. **(b)** Zoomed view of the thruster commands.

Figure 4.6: The response of the thruster commands when the vehicle performs a test path with two static region of avoidance.

4.1.4 Case 3: Dynamic Region of Avoidance

The response can be seen in the animation in Figure 4.7 and shows how the elastic path adapts to avoid the incoming region of avoidance. The large repulsive gain found in Section 4.1.1 makes the response quick enough to evade the area. As before, it also avoids the static region of avoidance.

Note also that since the algorithm has already planned a path to avoid the obstacle at the start position, it does not return to the initial path after it passes the region of avoidance but return after it has passed the *initial* region of avoidance.

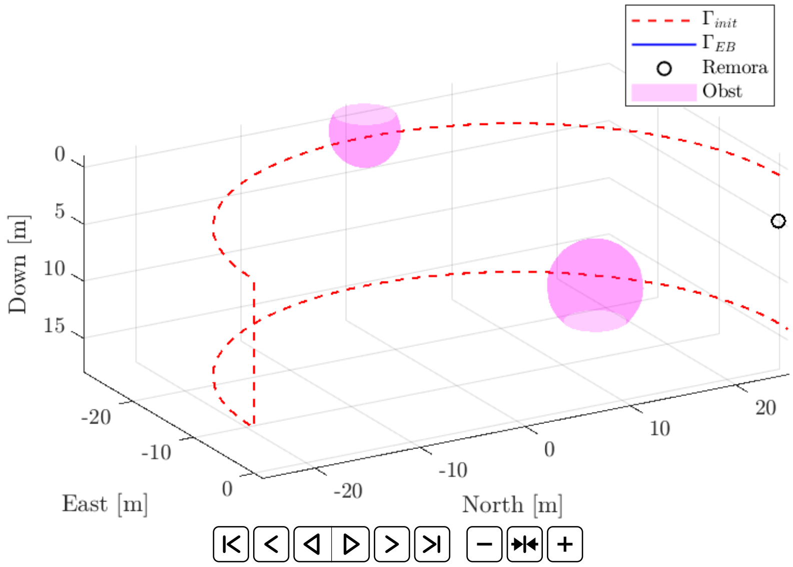


Figure 4.7: An animation on the behaviour of the Remora while facing both static and dynamics region of avoidance. For static figures, see Figure A.3.

4.1.5 Case 4: Ocean Current

The first simulation was done without net deformation, and in Figure 4.8 one can see the resulting path from the four ocean current configurations and the direction of the current. It shows that the actual path is very similar throughout the different configurations since the vehicle is attached to the net. In Figure 4.9a one can see how the output from the feedback linearisation controller increase when the current and the vehicle's velocity has opposite direction, and vice versa. The solid lines in Figure 4.9 represent the behaviour when the ocean current is absent and can be viewed as a reference for the other configurations, as this manages to follow the desired surge speed with high precision. Furthermore shows the right axis of Figure 4.9a the development in the actual surge the controllers aims to regulate, and the deviation is minimal for all simulations. The left axis shows the output from the controllers. The different configurations of ocean current is displayed in Table 3.5.

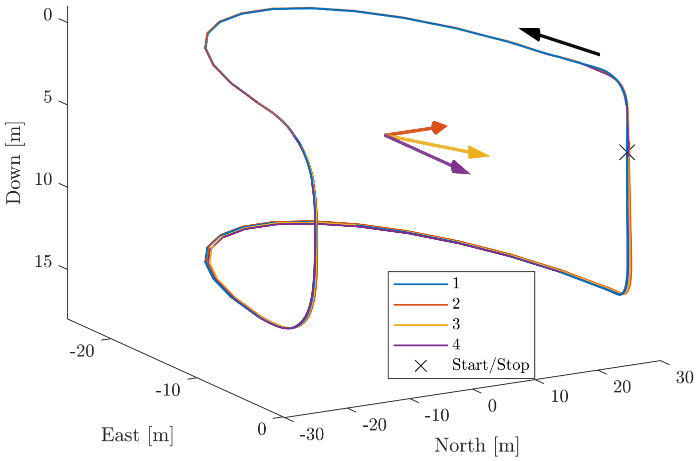
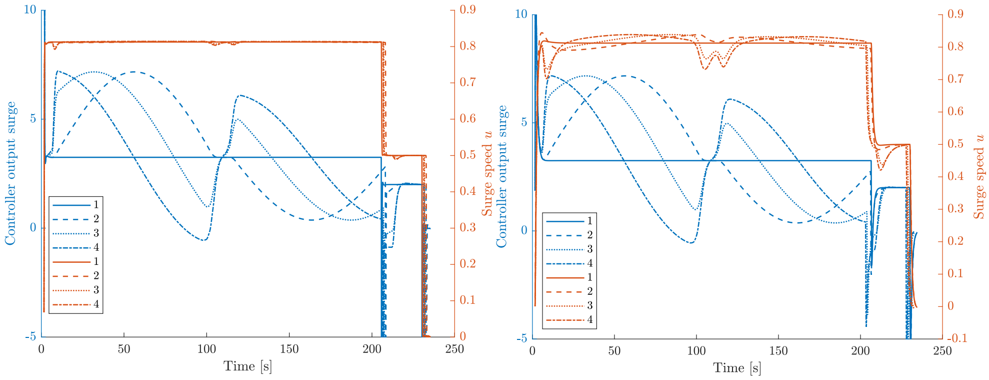


Figure 4.8: Figure of the actual path from the vehicle together with the direction of the ocean current visualised in corresponding colors.

During configuration 4, one can see that the feedback linearisation controller need to increase the output at the beginning of the simulation, as the robot turns directly into the current. During the second configuration, the robot gradually meets the current head-on, as one can see by the blue dotted line (2) in Figure 4.9a which increases gradually before it decreases, and eventually are moving with the ocean current leading the output to go towards zero.

Figure 4.9b displays how the PI controller is performing during the same simulation setup. One can see deviations in the surge speed compared to the solid red line representing the simulation without ocean current.



(a) Feedback linearisation controller.

(b) PI controller.

Figure 4.9: Graph of how the control output is changing due to different angles of ocean currents. The actual surge speed is plotted against the right y-axis.

Comparing Figure 4.9a and Figure 4.9b one can see that the controller output have many similiarities, but some deviations regarding the surge speed. While the feedback linearisation controller in Figure 4.9a maintain relatively close to the setpoint, the PI controller in Figure 4.9b deviates from the setpoint.

In the next simulation, the fish cage dynamics was turned on for a more realistic simulation, making the working environment quite complex as a large structure connected with ropes are highly dynamic. With a current of 1 m/s, the cage deformation becomes fairly dramatic, as shown in Figure 4.10.

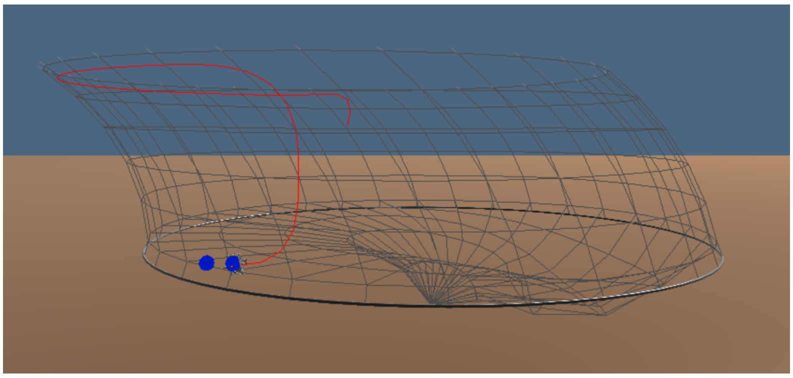
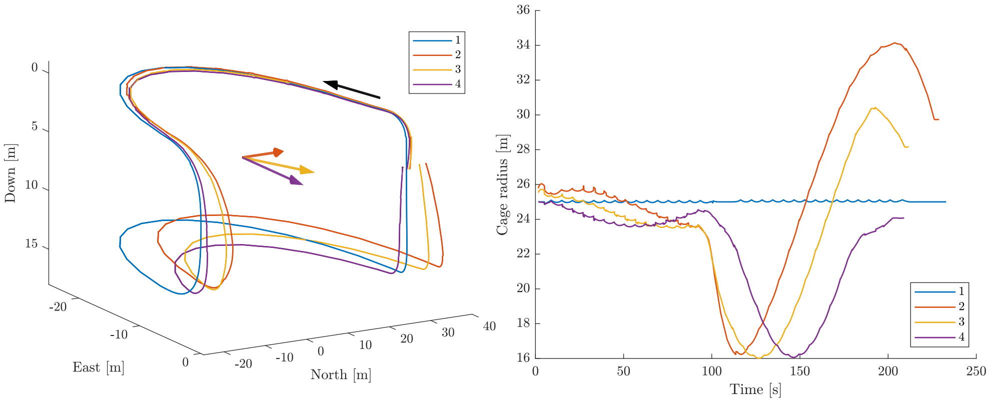


Figure 4.10: Screenshot of the simulation in FhSim during configuration 4 with net deformation.



(a) Resulting paths.

(b) The cage radius varies.

Figure 4.11: Figure 4.11a shows how the path is affected by the deformation of the fish cage caused by ocean currents , while Figure 4.11b shows the variations in the distance from the z-axis through origo to the vehicle, denoted as cage radius.

Figure 4.11a illustrate the effect the ocean current has on the path with the four different configurations. One can see that the different angles of current influence the vehicle's path differently. Note that the deformation is larger further down on the net compared to the top. This can also be seen in Figure 4.10 and is caused by the relatively stiff construction of the top ring of the cage.

In Figure 4.11b the cage radius of the position of the Remora vehicle is displayed. Since the vehicle is attached to the net, the cage radius and the distance from the z-axis in origo to the Remora are the same. This figure also illustrates how the distance is changing more when the vehicle starts to move downwards on the net after approximately 100 seconds, as explained above. One can also see small deviations along the lines. These are induced when the robot switches the net panel when traversing the net. The net is structured by many small panels attached to each other, constructing a non-continuous circle. In Figure 4.10 the panels are visible.

4.1.6 Case 5: Ocean Current with Static and Dynamic Region of Avoidance.

The resulting path with static obstacles with and without ocean current can be seen in Figure 4.12. Both simulations show that the region of avoidance is averted, but the path subjected to current is shifted in the direction of the current. The deformation is also less intense compared to the previous simulations, as the ocean current magnitude is halfed.

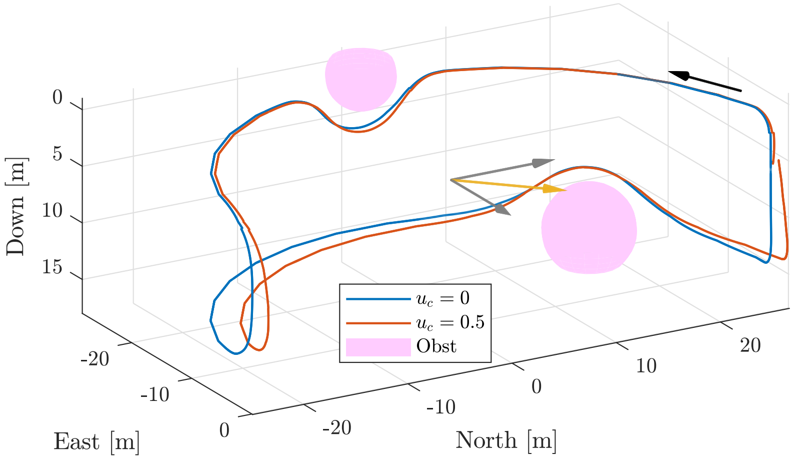


Figure 4.12: The response with static region of avoidance, with and without ocean current. The yellow arrow indicates the ocean current direction relative to the x^n and y^n axis in grey.

Even though the obstacles are initialized without velocity, they are "attached" to a dynamic net, making the obstacles move in space. This effect can be seen in the animation in Figure 4.13, where the regions of avoidance are changing the length from the z-axis through origo equally as the vehicle. It is clear that when the robot moves down on the net, changing the distance from the centre of the cage to the vehicle due to the net's structure, the obstacles move in the same direction.

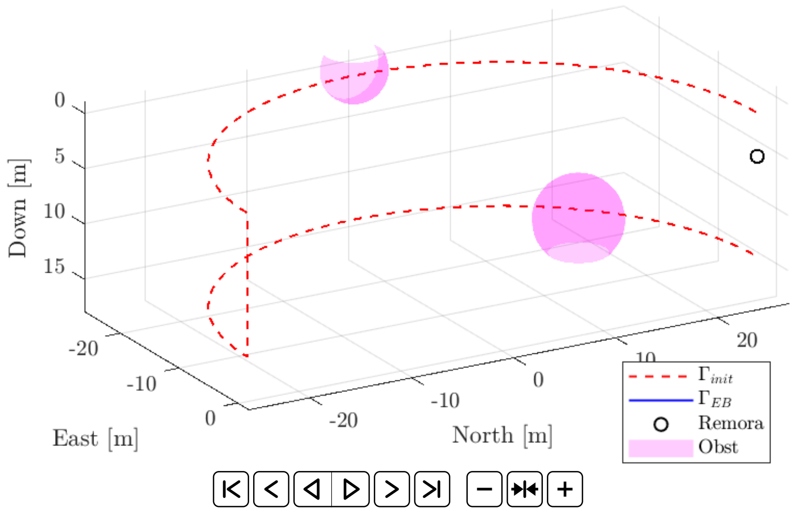


Figure 4.13: Animation of the Remora with two static regions of avoidance. For static figures, see Figure A.4.

The resulting behaviour when a dynamic obstacle was introduced is visualised in Figure 4.14, and one can see how the method calculates a feasible path, similar to the simulation without ocean currents. The distance between the dynamic region of avoidance and the vehicle is smaller than with the static region but is avoided still.

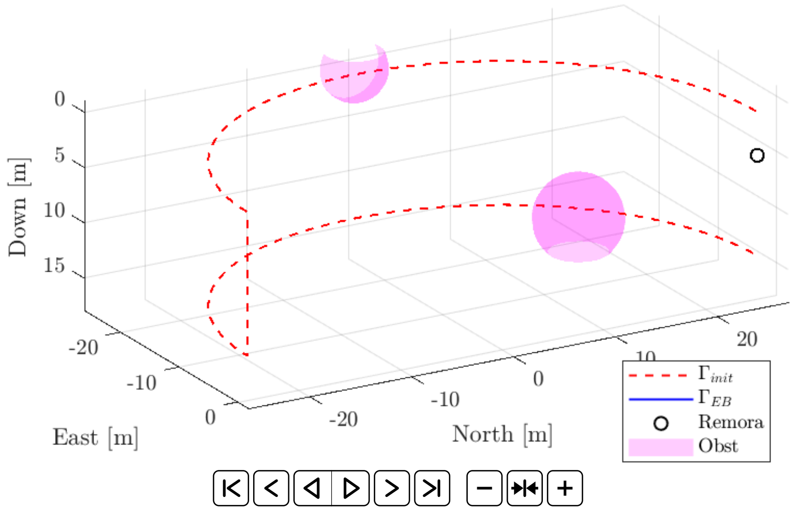


Figure 4.14: Animation of the Remora with one static and one dynamic region of avoidance. For static figures, see Figure A.5.

4.1.7 Case 6: Five-round Cleaning Routine under Non-Ideal Conditions

Two simulations were conducted in this case study. The first was conducted without ocean current and with two static regions of avoidance. The result is depicted in Figure 4.15 and shows how the battery level decreases as the vehicle moves with the colour of the line corresponding to the colour bar. The trigger point is marked with an asterisk and shows when the battery level is at the predefined limit at 20%, and the docking procedure is invoked to charge the battery. When docking, the EBM finds the shortest path to the docking station from the current position and therefore deviates slightly from the initial path.

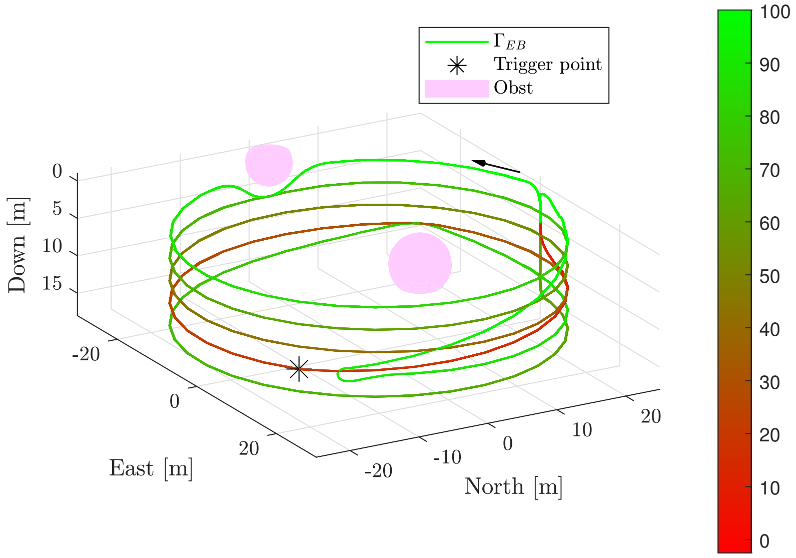


Figure 4.15: A five-round cleaning routine with corresponding battery level and trigger point.

Figure 4.16 illustrates the second simulation with an animation of how the vehicle is moving while being subject to ocean current, net deformation, static and dynamic regions of avoidance. The yellow arrow represents the direction of the ocean current. When the battery level becomes under 20%, the vehicle docks, recharge and continue the initial cleaning routine.

One thing to notice is the effect of the algorithm when planning and adapting a long time ahead, causing the path to avoid the previous position of the region of avoidance, also discussed in Section 4.1.4. On the second round in the animation, one can see that it avoids the region of avoidance but does not return to the initial path immediately.

Also, in this simulation, the deformation causes the deviation between the initial path and actual path to increase as the vehicle increase the depth.

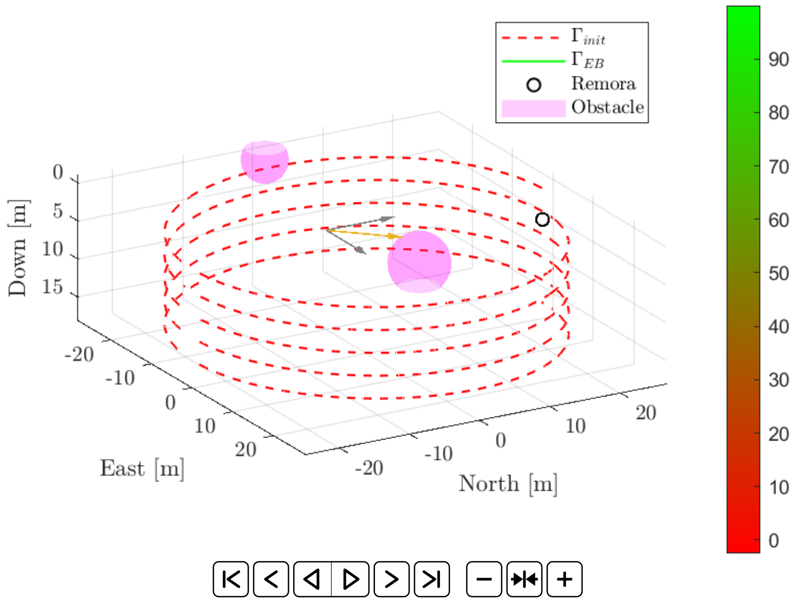


Figure 4.16: Animation of a five-round cleaning routine with docking procedure invoked by a trigger routine when the battery level gets below the limit. For static figures, see Figure A.6.

4.1.8 Time Complexity

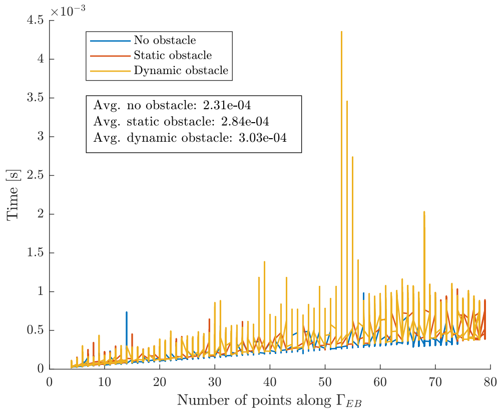
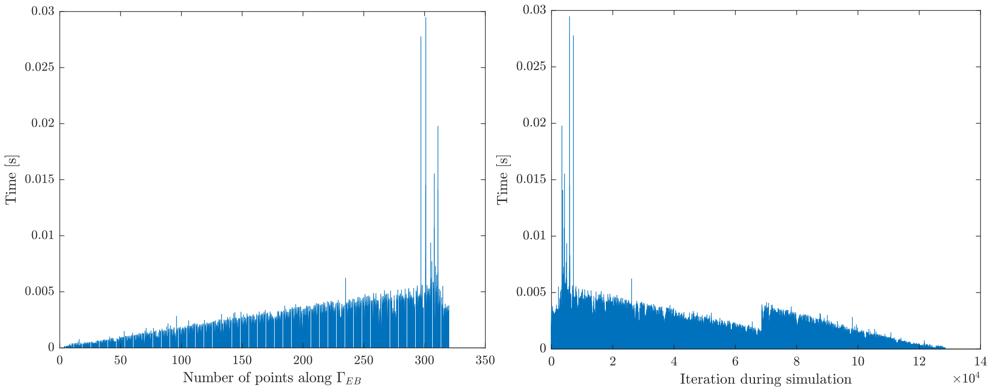


Figure 4.17: Time complexity with three different configurations, evaluated against the number of points along the Γ_{EB}.

In Figure 4.17 one can see the time consumption in three different configurations when using the testing path and how it develops with the number of points in the path, $\mathbf{\Gamma}_{EB}$. There is a trend that the time consumption, when subject to extra interference, increase with the difficulty of the interference. For instance, the time consumption is larger with a dynamic obstacle than with static obstacles. Furthermore, both are larger than the time consumption without any regions of avoidance. The time used in average is also displayed in Figure 4.17 and confirms the trend in the graph.



(a) Number of points vs time.

(b) Iterations vs time.

Figure 4.18: Time consumption of the EBM. Figure 4.18a shows how the length of the EBM relates to the consumption of time, and Figure 4.18b displays how the time consumption relates to the iterations of the simulation.

Figure 4.18 shows the time used during the simulation of Case 6 with ocean current and dynamic regions of avoidance. One can see a connection between the length of the path and the time used. Since the path gets shorter and shorter, as the algorithm deletes the preceding points as it is visited, the time consumption drops when the iteration increases. The spikes around 300 points in Figure 4.18a are due to the dynamic region of avoidance the vehicle encounters early in the simulation when the path is the longest. This can also be viewed in Figure 4.18b where one can see the same spikes among the early iterations.

Around the 7×10^4 iteration in Figure 4.18b, one can see an increase in time. This is when the docking procedure is triggered, and a new path must be added and adapted from the current position to the docking station and back to the original path, hence increasing the length of the path.

4.2 Simulations of the BlueROV2

The following results were obtained by simulating the BlueROV2 in a similar environment as the Remora vehicle.

4.2.1 Verifying the Tuning Parameters of the EBM

The resulting path with the same configuration as found for the Remora vehicle is shown in Figure 4.19.

The planner manages to find a feasible path that avoids any region of avoidance while maintaining reasonably close to the initial path and keeping the heading of the BlueROV2 normal to the net. This is visualised with the blue arrows. One can also see that the deviation from the initial path is only in the z-direction as the other direction are neglected in the planner.

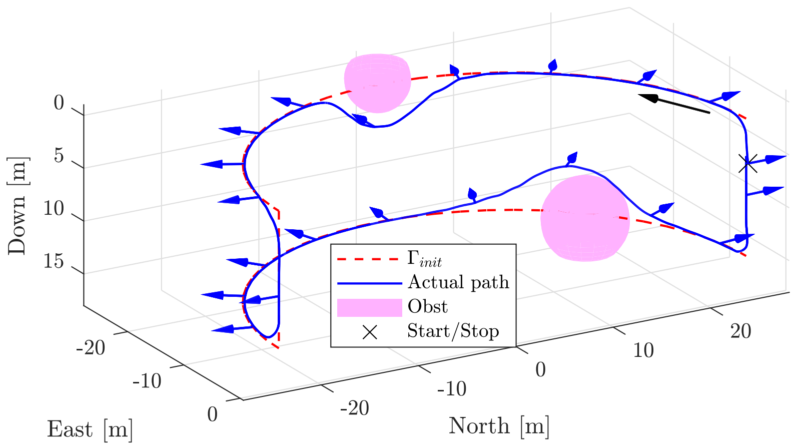


Figure 4.19: The resulting path from simulation with no ocean currents and two static obstacles.

4.2.2 Case 1: Five-round Cleaning Routine under Ideal Conditions

Figure 4.20 shows that the response is quite similar to the one obtained from the Remora in the same situation. The vehicle manages to follow the predefined path closely. As one can see, the actual path deviates more towards the end of the simulation due to the imprecise initial path. The blue arrows represent the heading of the ROV and illustrate how the ROV's heading always is normal to the net.

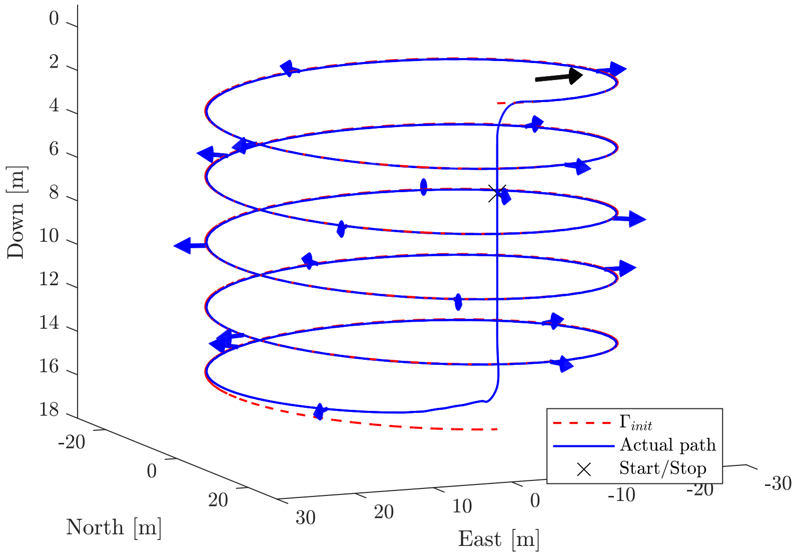
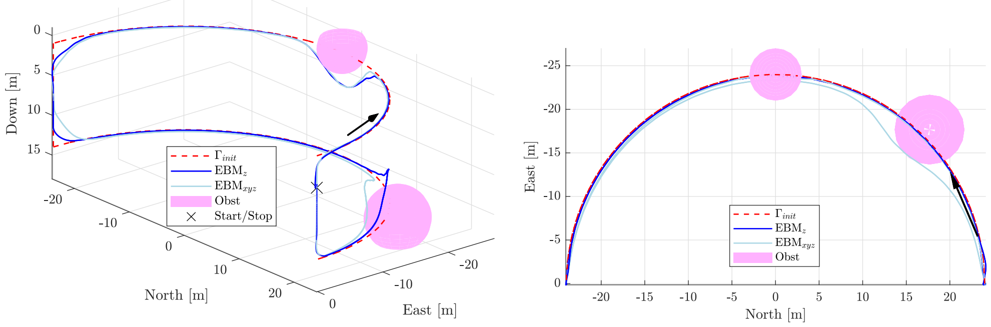


Figure 4.20: A five-round cleaning routine with the ROV. The blue arrows represent the heading of the ROV.

4.2.3 Case 2: Static Region of Avoidance

Figure 4.21 depicts two different configurations of deformation allowance of the EBM. The blue line represents the configuration with the only deformation in the z-axis, while the light blue represents the configuration with deformation in all axis.



(a) Resulting path.

(b) Top-down view of the resulting path.

Figure 4.21: Different resulting path with deformation in z-axis (blue line) and in x-, y-, and z-axis (light blue line).

From Figure 4.21a one can see that the resulting path with deformation in all directions is relatively similar to the one with deformation only in the z-direction. Since the first region of avoidance is centred in the middle of the path, the deformation is mainly in the z-direction for both cases, as one can see in Figure 4.21a. However, in the lowest region of avoidance, there is an offset between the initial path and the centre of avoidance, best viewed in Figure 4.21b. Here the planner finds it advantageous to adapt the path in all directions, and one can see in both Figure 4.21a and Figure 4.21b that the light blue path differs from the solid blue line.

4.2.4 Case 3: Dynamic Region of Avoidance

The animation in Figure 4.22 shows that the vehicle finds and follows a feasible path when being subject to static and dynamic regions of avoidance while keeping the heading normal to the net pen. The heading is visualised by the blue arrow pointing in the same direction as the heading of the BlueROV2.

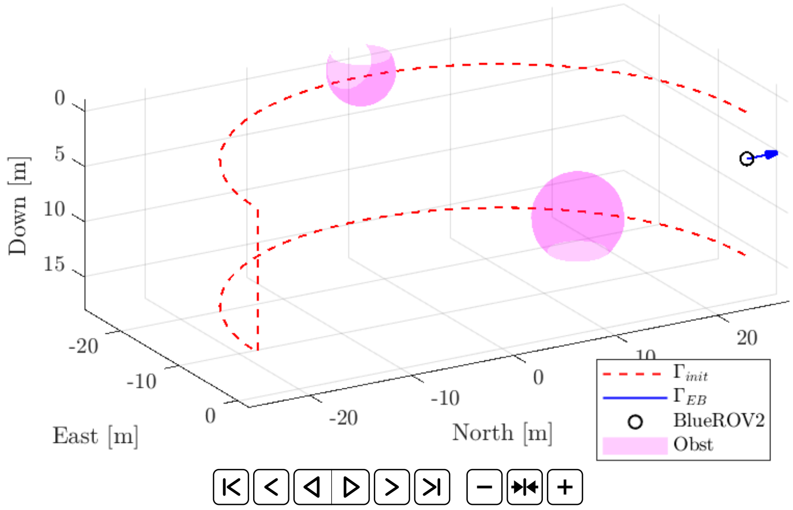
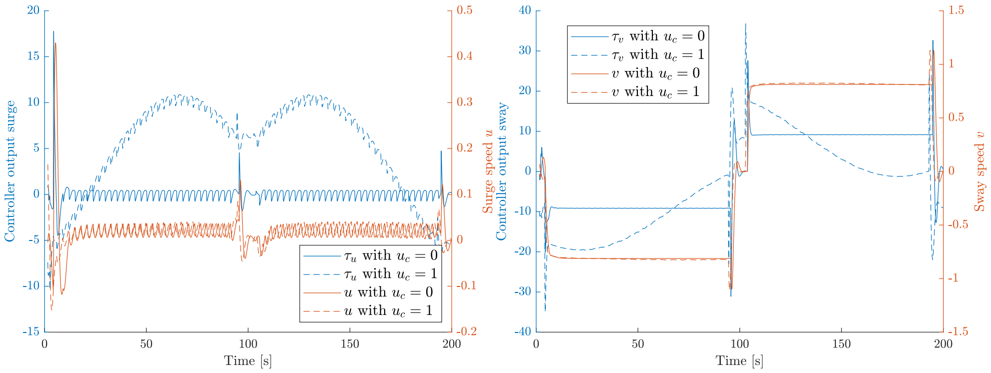


Figure 4.22: An animation on the behaviour of the ROV while facing both static and dynamics region of avoidance. For static figures, see Figure A.7.

4.2.5 Case 4: Ocean Current

In Figure 4.23 one can see the effect the ocean forces has on the vehicle as the controllers adapt the output to maintain the same speed as the vehicle did without external disturbance. The red lines represents the sway and surge speed in Figure 4.23b and Figure 4.23a and differs marginally with and without ocean current. The blue lines, however, has a larger deviation, which represents the controller output in each case. With ocean forces included, the controllers adjust the output continuously to compensate for the external forces to maintain the desired speed.

The sway speed in Figure 4.23b change the sign of the speed halfway since the path returns toward the docking station, and the ROV maintains the heading normal to the net during the whole simulation. The small oscillating variations in Figure 4.23a are due to the non-continuous circular path the heading controller uses to control the heading angle. Consequently, that affects the desired surge due to the net following equations defined in Section 3.4.3. When the net following is activated, most of the movement is done in the sway direction, thus low values in the surge direction.



(a) Surge control output and speed

(b) Sway control output and speed

Figure 4.23: The response in surge and sway speed, due to the controllers adapting to the ocean forces.

4.2.6 Case 6: Five-round Cleaning Routine under Non-Ideal Conditions

The first simulation was conducted with static obstacles and without ocean forces acting on the vehicle. The response can be seen in Figure 4.24, where the colour of the line represents the battery level. The blue arrows indicate the heading of the ROV. When the battery level is below 20%, a trigger is invoked, and the docking procedure starts. The ROV finds the shortest feasible path to the docking station, recharges, and returns fully charged to the trigger point to continue the routine. Note that during docking, the net following is turned off, and the vehicle moves in the surge direction. This can be viewed by the blue arrow on the path from the trigger point towards the docking station.

Moreover, in the second simulation, a dynamic region of avoidance and ocean current was added to the environment, and the response can be seen in the animation in Figure 4.25. The performance is very similar to the Remora vehicle in the same environment, as the path is followed closely while avoiding regions of avoidance before a trigger induces the docking procedure. Where it differs, once again, is to and from the docking station, as one can see from the blue arrow indicating the heading of the vehicle, that the vehicle's motion is mainly performed in the surge direction.

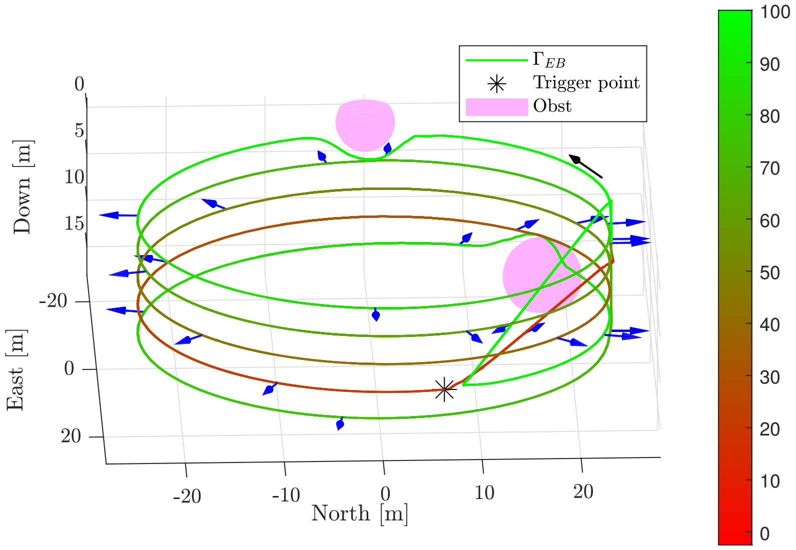


Figure 4.24: A five-round cleaning routine with two static regions of avoidance, and a low battery trigger.

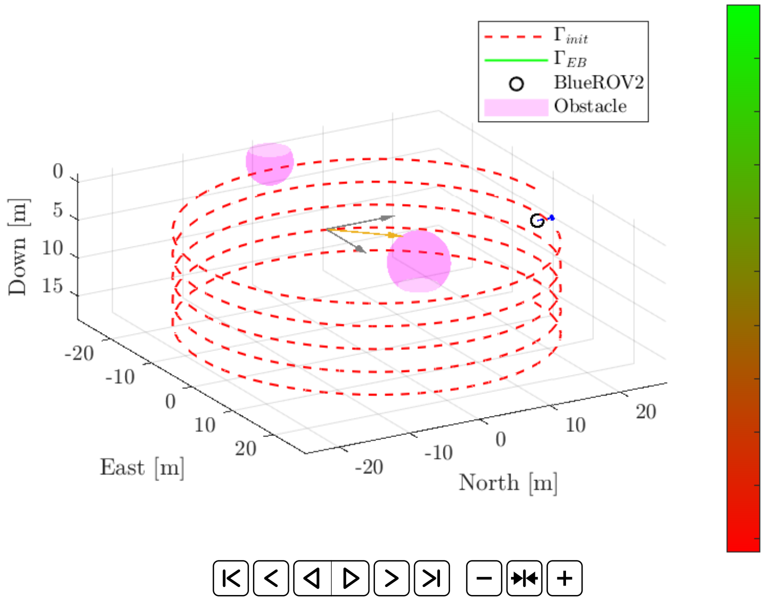


Figure 4.25: A five-round cleaning routine with static and dynamic regions of avoidance, ocean current, and a low battery trigger. For static figures, see Figure A.8.

Chapter 5

Discussion

From the extensive simulations, the strengths and weaknesses of the EBM have been exploited in an aquaculture environment with real challenges. The main findings are discussed in this chapter.

5.1 Time Complexity

The results from Section 4.1.8 showed a maximum time consumption of approximately 0.03 seconds when adapting 300 points with a dynamic region of avoidance. However, for the most part, the algorithm used below 0.005 seconds for each iteration. This supports that the algorithm seems suitable for real-time implementation in a system like the Remora. The Remora vehicle has a relatively slow working speed, and the underwater environment is more viscous than in the air, making obstacles move slowly. Combining these properties will also decrease the requirements needed for the time complexity of the motion planner.

One suboptimal response from the EBM was discovered in Section 4.1.4 and enhanced in Section 4.1.7. The suboptimality regards the planner's ability to return to the initial path when a dynamic region of avoidance is present. Since the path constantly adapts, the path avoids every place the obstacle *has* been, instead of avoiding the obstacle when nearby and return to the initial path as soon as possible, ensuring most of the initial path is covered.

A possible solution could be to decrease the number of points that are adapted by the EBM for each time step. For instance, to only evaluate 20 bubbles ahead of the vehicle. Since the number of points is heavily decreased from approximately 300 in Section 4.1.8 to, i.e. 20, the results obtained in Section 4.1.8 implies that the time complexity will decrease with the number of points, and hence could improve the performance in a real-time application.

However, the algorithm may add or delete some points due to bubble reorganisation in Phase 3, hence causing a varying length of the path, $\mathbf{\Gamma}_{EB}$, which requires vector manipulation. This might increase the time complexity of the solution.

5.2 Elastic Band Method

With static and dynamic regions of avoidance, in a changing environment, the EBM calculated a feasible path and thus avoiding every obstacle during the simulations for both vehicles, autonomously without human interference, showing promising results for the EBM to be used on the Remora vehicle.

The results obtained using the BlueROV2 model, was quite similar to the Remora vehicle, implying that the EBM is versatile and suitable on several vehicles types in an underwater aquaculture environment. Considering the fact that it has many tuning possibilities, the methods use area is wide.

The deviation towards the end of the cleaning path, seen in Section 4.1.2, is due to a lack of precision in the initial path, where the path exceeds the limit of the depth of the net pen. In a real-life situation, the initial path can be determined by an operator, and mistakes can happen. The EBM manages to adapt the path such that it does not exceed the limits of the physical cage and shows robustness. The results showed that it was accounted for already in the first iteration of the algorithm by the line marked Γ_{init}.

From the animations of the simulations with the dynamic region of avoidance, one can see that the clearance between the dynamic region and the vehicle was slightly less than between the static region and the vehicle. As a relatively fast-moving object of 0.25 m/s in the opposite direction of the vehicle was implemented, the assumption regarding underwater obstacles are slow, causing the obstacles velocity to be neglected in Section 3.4, might not be valid. If it is considered necessary, one can implement the velocity of the obstacle to the algorithm, as suggested in [38], and explained in Section 2.3.1.

A more advanced system that evaluates the possibilities to follow the predefined path towards the docking station would be advantageous to investigate. By, for instance, estimate the energy consumption required to stay on the predefined path and then decide if the vehicle should find the shortest path or not. In Figure 4.15 this most likely would be beneficial as the shortest path deviates slightly from the predefined path.

Neither of the simulation cases implemented a simulation setup where there was no feasible path and thus did not exploit the weakness demonstrated in the 2D-version during the development in Section 3.2. The ability to globally replan and find a new feasible path might be improved in the case of the "door being closed".

In that case, the work done in [37] is interesting and might be investigated further to improve the global performance of the method.

One possible improvement, which utilises an already implemented feature, could be made where regions of attractions are introduced. This should work opposite of the regions of avoidance and can be a place where the mitigation is typically higher, places where a hole in the net is common or used in conjunction with the adapted path after dynamic regions of avoidance. Then one could add a region of attraction behind the avoidance region to contract the elastic band back towards the initial path. This might be a solution instead of limiting the adaption distance ahead, proposed in Section 5.1.

Not yet discussed are the requirements needed for the EBM. The algorithm utilises the position of the vehicle and possible regions of avoidance to calculate the circle of acceptance and the external forces. In an ideal numerical simulation environment, these are easily obtained with high precision. However, in a real-life situation, an accurate underwater positioning system is of more significant challenge. WaterLinked AS has developed underwater positioning systems to be used in aquaculture as one of their area of focus and might be something to investigate further towards a fully operating system.

5.3 Control System

It was illustrated in Section 4.1.5 that the controllers were able to adapt the output when the vehicle was subject to environmental disturbances to maintain the desired set-point. Although the feedback linearisation controller outperformed the PI controller, it is hard to obtain the parameters needed in the feedback linearisation controller. In contrast, the PI controller only requires tuning of the controller to adapt it to the system. In that case, the PI controller shows relatively good results and might be the most reasonable choice despite the suboptimal response.

5.4 Lab Trial

A lab trial should be executed and are planned to be after the delivery date of this thesis to verify the results obtained in the numerical simulations. In order to test the performance of the EBM a positioning system needs to be implemented. This is accessible through a system called Qualisys at the MC-lab at Tyholt and can give the position of the vehicle and any regions of avoidance defined in the workspace. An interface through FhSim has been implemented by SINTEF Ocean such that the vehicle can be controlled with commands sent from FhSim.

Some of the work presented in the implementation chapter was preparation for the lab trials, like the thruster allocation matrix for the BlueROV2, derived in Section 3.3. This has been verified and tested in a small tank at NTNU on Gløshaugen before a small scale field trial was conducted at Brattørkaia in Trondheim. Together with SINTEF Ocean, a simple heading and depth controller was implemented and verified such that the BlueROV2 is ready for lab trials with the EBM.

The results from this thesis, combined with the results from the planned lab trials, will be submitted for journal publication.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis has presented results obtained from simulations of the EBM used on the Remora and the BlueROV2 in an aquaculture environment. An adaptive motion planning method was chosen based on the findings in the specialisation project and further reviewed in detail. It was eventually implemented in FhSim through prototyping in MATLAB before it was realised through a SimObject in C++ to work directly with FhSim.

A control system and heading guidance were implemented to complete the system's structure, making simulations feasible. Through the virtual experiments, the planner's ability to replan a feasible path away from regions of avoidance has been evaluated. Environmental forces were added, causing net deformation for a more realistic workspace, and the performance of the Remora vehicle was assessed. At last, a complete cleaning routine was conducted, including ocean currents, regions of avoidance, and a docking procedure to interrupt the initial path before finishing the cleaning routine after recharging, all autonomously.

The performance was compared against a conventional free-moving ROV. The findings open the possibilities for implementation in other underwater vehicles besides the Remora, but also for conducting lab trials with an ROV.

Through the extensive simulation study, it has been adapted and proved that the EBM might be suitable for use in an aquaculture environment, both for the Remora but also for other underwater vehicles like the BlueROV2.

Some work was done with the objective to facilitate a lab trial with an ROV, testing the EBM in a real-life situation on an underwater vehicle. The thruster allocation matrix was calculated and verified in a small scale lab and field trial, making the system ready for a large scale lab trial. The results obtained from the lab trial are aimed to be used in a journal publication.

6.2 Future Work

Even though the mathematical models are exact, the system should be verified in real life to prove the method for use in an aquaculture setting. First, a lab trial should be conducted, and if the results are satisfactory, a field test would be the next step in making the application closer to commercial use.

The system has proved many strengths, but as discussed in Chapter 5, it also has some weaknesses that might have room for improvement. Accounting for the velocity of the region of avoidance and improving the algorithm's global path planning abilities would enhance the performance of the solution. It would also be very interesting to investigate further the idea of implementing a region of attractions to expand further the possibilities and use area of the EBM.

Bibliography

[1] Sintef, *NetClean 24/7 - SINTEF*. [Online]. Available: https://www.sintef.no/en/projects/netclean-247/ (visited on 18/09/2020).

[2] S. Fjæra, *TTK4551-Optimal and adaptive trajectory planning for permanent resident cleaning robot operating in fish farms*, Trondheim, Dec. 2020.

[3] Fiskeridirektoratet, *Biomassestatistikk etter fylke*. [Online]. Available: https://www.fiskeridir.no/Akvakultur/Tall-og-analyse/Biomassestatistikk/Biomassestatistikk-etter-fylke (visited on 18/09/2020).

[4] Regjeringen, *Regjeringen skrur på trafikklyset i havbruksnæringen - regjeringen.no*. [Online]. Available: https://www.regjeringen.no/no/aktuelt/regjeringen-skrur-pa-trafikklyset-i-havbruksnaringen/id2688939/ (visited on 07/10/2020).

[5] J. Bannister, M. Sievers, F. Bush and N. Bloecher, 'Biofouling in marine aquaculture: a review of recent research and developments,' *Biofouling*, vol. 35, no. 6, pp. 631–648, Jul. 2019, ISSN: 10292454. DOI: 10.1080/08927014.2019.1640214. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/08927014.2019.1640214.

[6] E. Leclercq, B. Zerafa, A. J. Brooker, A. Davie and H. Migaud, 'Application of passive-acoustic telemetry to explore the behaviour of ballan wrasse (Labrus bergylta) and lumpfish (Cyclopterus lumpus) in commercial Scottish salmon sea-pens,' *Aquaculture*, vol. 495, no. August 2017, pp. 1–12, 2018, ISSN: 00448486. DOI: 10.1016/j.aquaculture.2018.05.024. [Online]. Available: https://doi.org/10.1016/j.aquaculture.2018.05.024.

[7] K. Eliasen, E. Danielsen, Á. Johannesen, L. L. Joensen and E. J. Patursson, 'The cleaning efficacy of lumpfish (Cyclopterus lumpus L.) in Faroese salmon (Salmo salar L.) farming pens in relation to lumpfish size and

seasonality,' *Aquaculture*, vol. 488, no. January, pp. 61–65, 2018, ISSN: 00448486. DOI: 10.1016/j.aquaculture.2018.01.026.

[8] I. Fitridge, T. Dempster, J. Guenther and R. de Nys, 'The impact and control of biofouling in marine aquaculture: a review,' *Biofouling*, vol. 28, no. 7, pp. 649–669, Aug. 2012, ISSN: 0892-7014. DOI: 10.1080/08927014.2012.700478. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/08927014.2012.700478.

[9] P. Berillis, E. Mente and K. Kormas, 'The Use of Copper Alloy in Aquaculture Fish Net Pens: Mechanical, Economic and Environmental Advantages,' *Journal of FisheriesSciences.com*, vol. 11, pp. 1–3, Oct. 2017. DOI: 10.21767/1307-234X.1000134.

[10] H. V. Bjelland, M. Fore, P. Lader, D. Kristiansen, I. M. Holmen, A. Fredheim, E. I. Grotli, D. E. Fathi, F. Oppedal, I. B. Utne and I. Schjolberg, 'Exposed Aquaculture in Norway,' in *OCEANS 2015 - MTS/IEEE Washington*, Institute of Electrical and Electronics Engineers Inc., Feb. 2016, ISBN: 978093395 7435. DOI: 10.23919/oceans.2015.7404486.

[11] Mithal, *Remora*. [Online]. Available: https://www.mithal.no/ (visited on 04/05/2021).

[12] H. Seehus, *Control Strategies for Autonomous Net Cleaning Operations using Underwater Robotic System*, 2020.

[13] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*, 1. Aufl. Hoboken: Wiley, 2011, ISBN: 1119991498.

[14] T. SNAME, 'Nomenclature for treating the motion of a submerged body through a fluid,' *The Society of Naval Architects and Marine Engineers, Technical and Research Bulletin*, no. 1-5, 1950.

[15] S. J. Ohrem, H. Amundsen and E. Kelasidi, 'Modelling and control of an underwater biofouling prevention and inspection robot,' *IROS 2021 (Submitted)*,

[16] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft: Theory and Practice*. Princeton university press, 2012, ISBN: 0691149216.

[17] M. Breivik and T. I. Fossen, 'Guidance laws for autonomous underwater vehicles,' *Underwater vehicles*, vol. 4, pp. 51–76, 2009.

[18] P. B. Sujit, S. Saripalli and J. B. Sousa, 'Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicless,' *IEEE Control Systems*, vol. 34, no. 1, pp. 42–59, Feb. 2014, ISSN: 1066033X. DOI: 10.1109/MCS.2013.2287568.

[19] R. T. Marler and J. S. Arora, 'Survey of multi-objective optimization methods for engineering Maximum objective function values,' DOI: 10.1007/s00158-003-0368-6.

[20] H. K. Khalil, *Nonlinear systems; 3rd ed.* Upper Saddle River, NJ: Prentice-Hall, 2002. [Online]. Available: https://cds.cern.ch/record/1173048.

[21] D. González, J. Pérez, V. Milanés and F. Nashashibi, 'A Review of Motion Planning Techniques for Automated Vehicles,' *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, Apr. 2016, ISSN: 15249050. DOI: 10.1109/TITS.2015.2498841.

[22] V. Kunchev, L. Jain, V. Ivancevic and A. Finn, 'Path planning and obstacle avoidance for autonomous mobile robots: A review,' in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4252 LNAI, Springer Verlag, 2006, pp. 537–544, ISBN: 3540465375. DOI: 10.1007/11893004_70.

[23] L. Yang, J. Qi, D. Song, J. Xiao, J. Han and Y. Xia, *Survey of Robot 3D Path Planning Algorithms*, 2016. DOI: 10.1155/2016/7426913.

[24] S. M. LaValle, *Planning Algorithms*. USA: Cambridge University Press, 2006, ISBN: 0521862051.

[25] Y. K. Hwang and N. Ahuja, 'Gross Motion Planning—a Survey,' *ACM Comput. Surv.*, vol. 24, no. 3, pp. 219–291, Sep. 1992, ISSN: 0360-0300. DOI: 10.1145/136035.136037. [Online]. Available: https://doi.org/10.1145/136035.136037.

[26] L. Zhang, Y. J. Kim and D. Manocha, 'A hybrid approach for complete motion planning,' in *IEEE International Conference on Intelligent Robots and Systems*, 2007, pp. 7–14, ISBN: 1424409128. DOI: 10.1109/IROS.2007.4399064.

[27] A. M. Lekkas, *Anastasios M. Lekkas Guidance and Path-Planning Systems for Autonomous Vehicles*. 2014, ISBN: 9788232601769.

[28] L. E. Dubins, 'On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents,' *American Journal of Mathematics*, vol. 79, no. 3, p. 497, Jul. 1957, ISSN: 00029327. DOI: 10.2307/2372560.

[29] J. A. Reeds and L. A. Shepp, 'Optimal paths for a car that goes both forwards and backwards,' *Pacific Journal of Mathematics*, vol. 145, no. 2, pp. 367–393, 1990, ISSN: 00308730. DOI: 10.2140/pjm.1990.145.367.

[30] I. Lugo-Cardenas, G. Flores, S. Salazar and R. Lozano, 'Dubins path generation for a fixed wing UAV,' in *2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014 - Conference Proceedings*, IEEE Computer Society, 2014, pp. 339–346, ISBN: 9781479923762. DOI: `10.1109/ICUAS.2014.6842272`.

[31] C. Goerzen, Z. Kong and B. Mettler, 'A survey of motion planning algorithms from the perspective of autonomous UAV guidance,' *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 57, no. 1-4, pp. 65–100, Jan. 2010, ISSN: 09210296. DOI: `10.1007/s10846-009-9383-1`. [Online]. Available: `https://link.springer.com/article/10.1007/s10846-009-9383-1`.

[32] M. Yan, D. Zhu and S. X. Yang, 'A Novel 3-D Bio-Inspired Neural Network Model for the Path Planning of An Auv in Underwater Environments,' *Intelligent Automation and Soft Computing*, vol. 19, no. 4, pp. 555–566, 2013, ISSN: 10798587. DOI: `10.1080/10798587.2013.869114`. [Online]. Available: `https://www.tandfonline.com/doi/abs/10.1080/10798587.2013.869114`.

[33] Abraham, R. Cuautle, M. Osorio and R. Zapata, 'Reactive Motion Planning for Mobile Robots,' in Oct. 2008, ISBN: 978-953-7619-01-5. DOI: `10.5772/6000`.

[34] R. Bohlin and L. E. Kavraki, 'Path planning using Lazy PRM,' *Proceedings-IEEE International Conference on Robotics and Automation*, vol. 1, pp. 521–528, 2000, ISSN: 10504729. DOI: `10.1109/ROBOT.2000.844107`.

[35] P. Corke, *Robotics, Vision and Control*, ser. Springer Tracts in Advanced Robotics. Cham: Springer International Publishing, 2017, vol. 118, ISBN: 978-3-319-54412-0. DOI: `10.1007/978-3-319-54413-7`. [Online]. Available: `http://link.springer.com/10.1007/978-3-319-54413-7`.

[36] S. Quinlan and O. Khatib, 'Elastic bands. Connecting path planning and control,' in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2, Publ by IEEE, 1993, pp. 802–807, ISBN: 0818634529. DOI: `10.1109/robot.1993.291936`.

[37] T. Brandt and T. Sattel, 'PATH PLANNING FOR AUTOMOTIVE COLLISION AVOIDANCE BASED ON ELASTIC BANDS,' *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 210–215, 2005, ISSN: 1474-6670. DOI: `https://doi.org/10.3182/20050703-6-CZ-1902.01245`. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S1474667016372573`.

[38] C.-T. Lee and C.-C. Tsai, '3D Collision-Free Trajectory Generation Using Elastic Band Technique for an Autonomous Helicopter,' in *Communications in Computer and Information Science*, vol. 212, Oct. 2011. DOI: 10.1007/978-3-642-23147-6_5.

[39] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018, ISBN: 0262039249.

[40] R. Meyes, H. Tercan, S. Roggendorf, T. Thiele, C. Büscher, M. Obdenbusch, C. Brecher, S. Jeschke and T. Meisen, 'Motion Planning for Industrial Robots using Reinforcement Learning,' in *Procedia CIRP*, vol. 63, Elsevier B.V., Jan. 2017, pp. 107–112. DOI: 10.1016/j.procir.2017.03.095.

[41] M. Everett, Y. F. Chen and J. How, *Motion Planning Among Dynamic, Decision-Making Agents with Deep Reinforcement Learning*, Oct. 2018.

[42] Y. Sun, J. Cheng, G. Zhang and H. Xu, 'Mapless Motion Planning System for an Autonomous Underwater Vehicle Using Policy Gradient-based Deep Reinforcement Learning,' *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 96, no. 3-4, pp. 591–601, Dec. 2019, ISSN: 15730409. DOI: 10.1007/s10846-019-01004-2. [Online]. Available: https://doi.org/10.1007/s10846-019-01004-2.

[43] FhSim, *FhSim - Simulation of Marine Systems*. [Online]. Available: https://fhsim.no/ (visited on 10/04/2021).

[44] SINTEF Fisheries Technology, *FhSim - Simulering av marine operasjoner og systemer*, 2018. [Online]. Available: https://www.sintef.no/programvare/fhsim/ (visited on 10/04/2021).

[45] K. J. Reite, M. Føre, K. G. Aarsæther, J. Jensen, P. Rundtop, L. T. Kyllingstad, P. C. Endresen, D. Kristiansen, V. Johansen and A. Fredheim, 'FhSim - Time domain simulation of marine systems,' in *Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE*, vol. 8A, American Society of Mechanical Engineers (ASME), Oct. 2014, ISBN: 9780791845509. DOI: 10.1115/OMAE2014-23165. [Online]. Available: http://asmedigitalcollection.asme.org/OMAE/proceedings-pdf/OMAE2014/45509/V08AT06A014/4431348/v08at06a014-omae2014-23165.pdf.

[46] MATLAB, *Differential-drive vehicle model - MATLAB*. [Online]. Available: https://www.mathworks.com/help/robotics/ref/differentialdrivekinematics.html (visited on 09/04/2021).

[47] MATLAB, *Create controller to follow set of waypoints - MATLAB*. [Online]. Available: `https://www.mathworks.com/help/robotics/ref/controllerpurepursuit-system-object.html` (visited on 09/04/2021).

[48] BlueRobotics, *BlueROV2 - Affordable and Capable Underwater ROV*, 2020. [Online]. Available: `https://bluerobotics.com/store/rov/bluerov2/` (visited on 10/04/2021).

[49] C.-J. Wu, *6-DoF Modelling and Control of a Remotely Operated Vehicle*, 2018.

[50] S. Pedersen, J. Liniger, F. F. Sørensen, K. Schmidt, M. Von Benzon and S. S. Klemmensen, 'Stabilization of a ROV in three-dimensional space using an underwater acoustic positioning system,' in *IFAC-PapersOnLine*, vol. 52, 2019, pp. 117–122. DOI: `10.1016/j.ifacol.2019.11.037`. [Online]. Available: `www.sciencedirect.com`.

[51] GitHub, *GitHub - patrickelectric/bluerov_ros_playground: Scripts to help BlueRov integration with ROS*. [Online]. Available: `https://github.com/patrickelectric/bluerov_ros_playground` (visited on 12/04/2021).

[52] GitBook, *Features · GitBook*. [Online]. Available: `https://www.ardusub.com/introduction/features.html` (visited on 10/04/2021).

[53] BlueRobotics, *T200 Thruster for ROVs, AUVs, and marine robotics*. [Online]. Available: `https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster-r2-rp/` (visited on 27/05/2021).

[54] Y. Cao, B. Li, Q. Li, A. A. Stokes, D. M. Ingram and A. Kiprakis, 'A nonlinear model predictive controller for remotely operated underwater vehicles with disturbance rejection,' *IEEE Access*, vol. 8, pp. 158 622–158 634, 2020, ISSN: 21693536. DOI: `10.1109/ACCESS.2020.3020530`.

[55] S. Lack, E. Rentzow and T. Jeinsch, 'Experimental Parameter Identification for an open-frame ROV: Comparison of towing tank tests and open water self-propelled tests,' in *IFAC-PapersOnLine*, vol. 52, Elsevier B.V., Jan. 2019, pp. 271–276. DOI: `10.1016/j.ifacol.2019.12.319`.

[56] Oceans2021, *Home - Oceans*. [Online]. Available: `https://global21.oceansconference.org/` (visited on 28/05/2021).

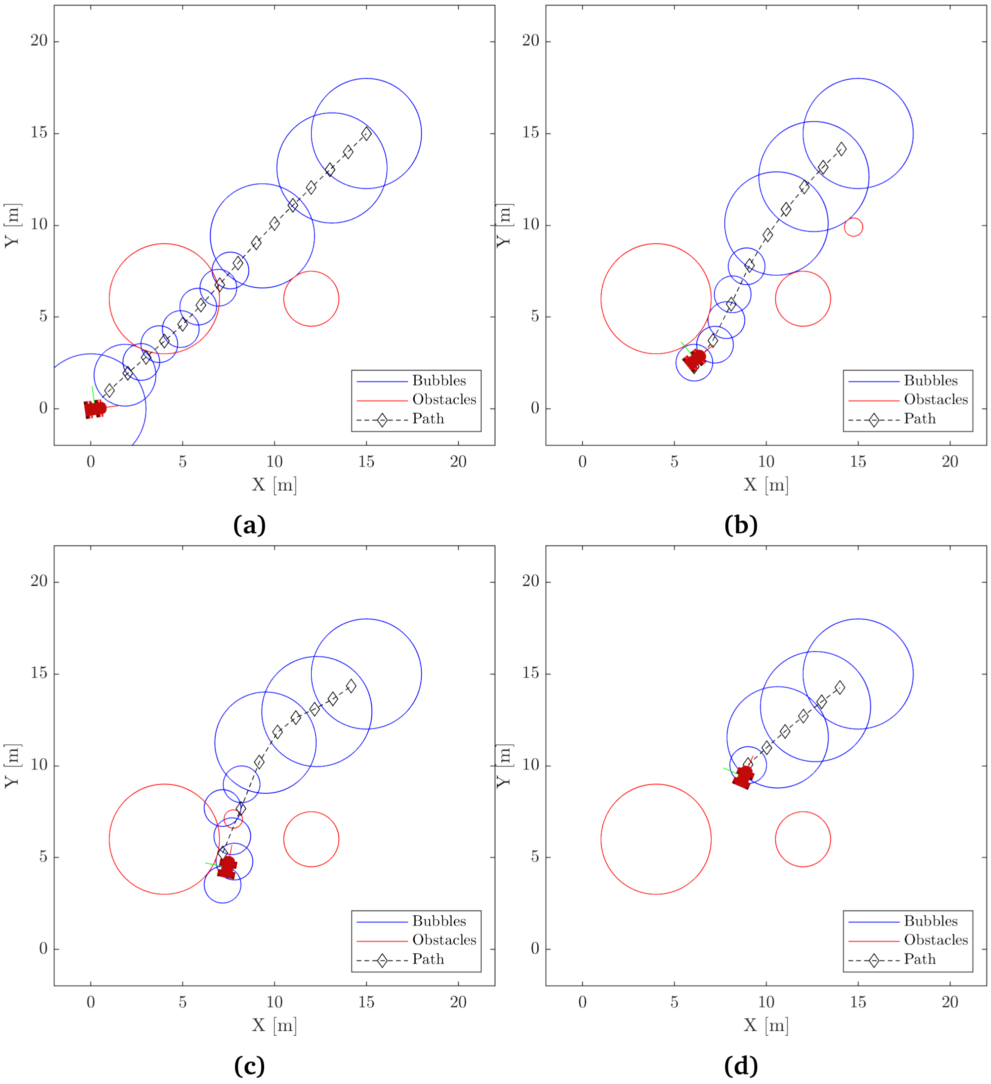
Appendix A

Static Figures of Dynamic Illustrations

If the reader can not play the animations throughout the thesis, static illustrations of the same situations are presented here. A series of figures are displayed to illustrate the dynamics in the animations, where the letter in each caption correspond to the order in the simulations. E.g. **(a)** is the first situation, and **(b)** is the next.

A.1 Implementation

A.1.1 Algorithm Development - 2D



(a)

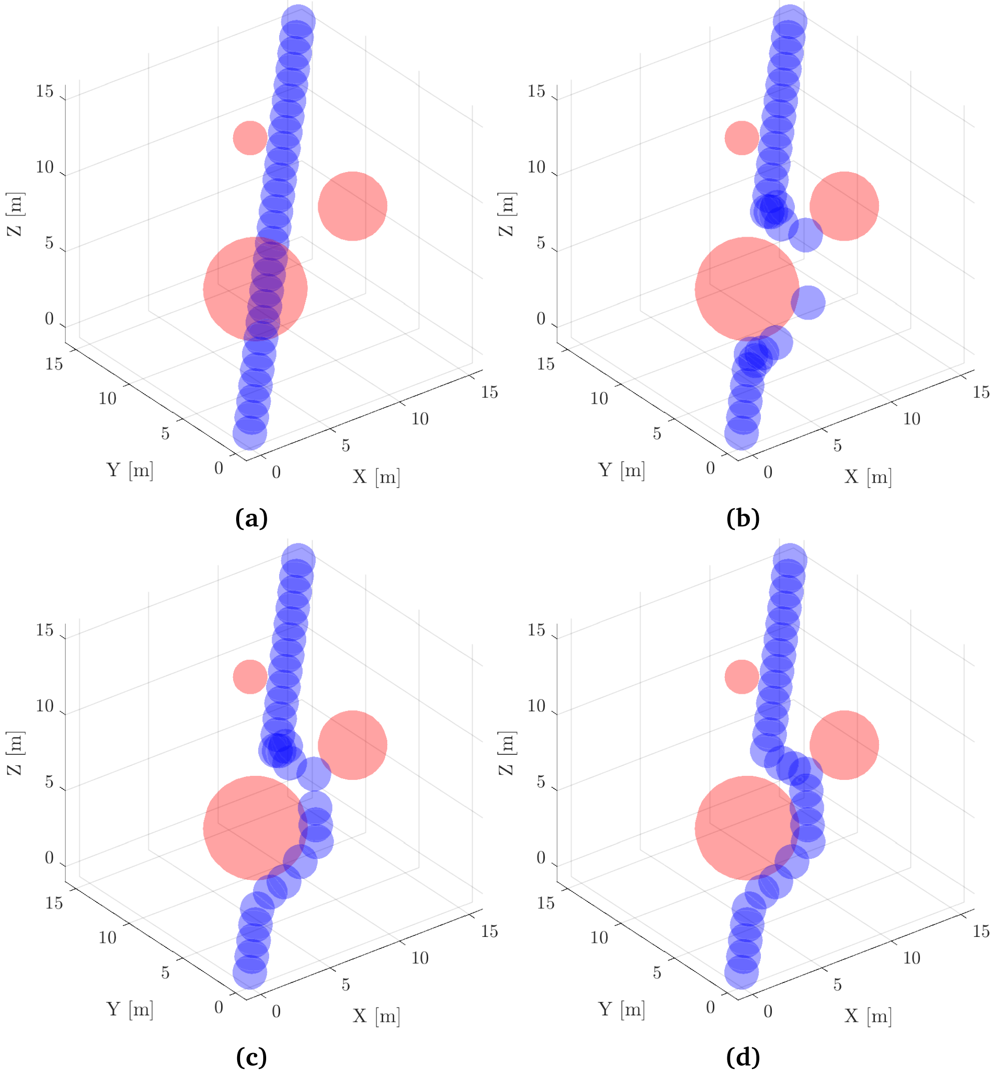
(b)

(c)

(d)

Figure A.1: Illustrating the animation in Figure 3.2.

A.1.2 Algorithm Development - 3D



(a)

(b)

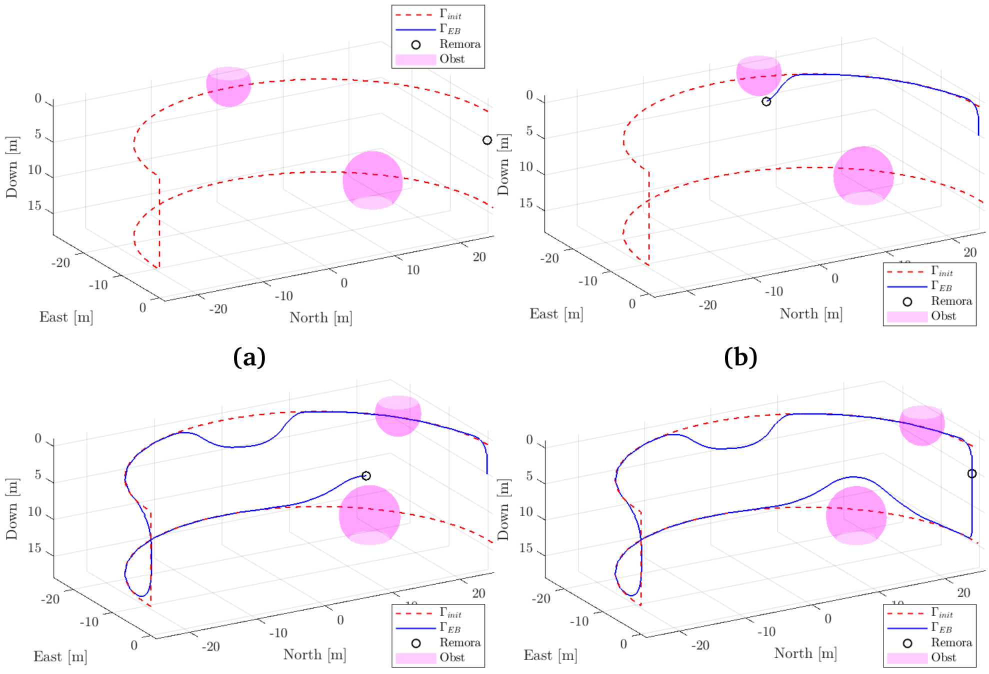
(c)

(d)

Figure A.2: Illustrating the animation in Figure 3.3.

A.2 Results

A.2.1 Remora - Case 3



(a)

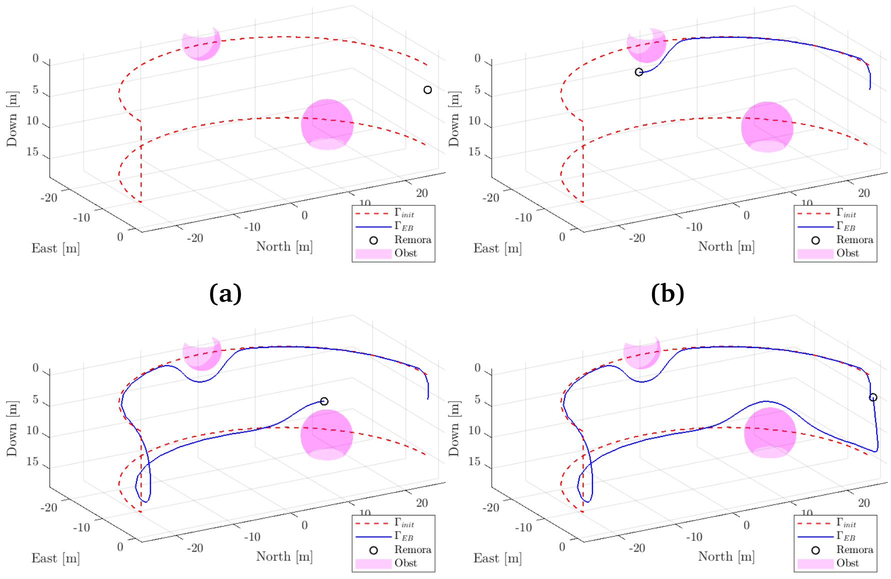
(b)

(c)

(d)

Figure A.3: Illustrating the animation in Figure 4.7.

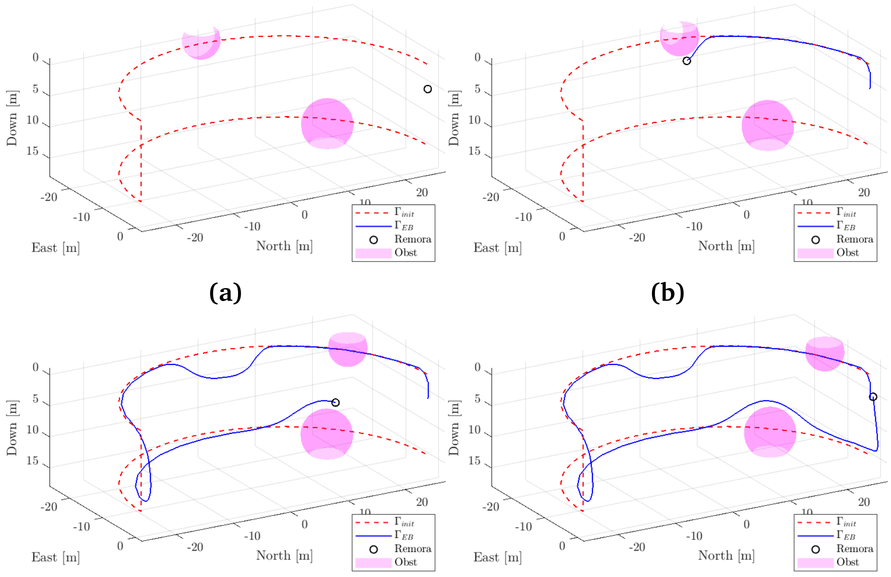
A.2.2 Remora - Case 5



(a) (b)

(c) (d)

Figure A.4: Illustrating the animation in Figure 4.13.

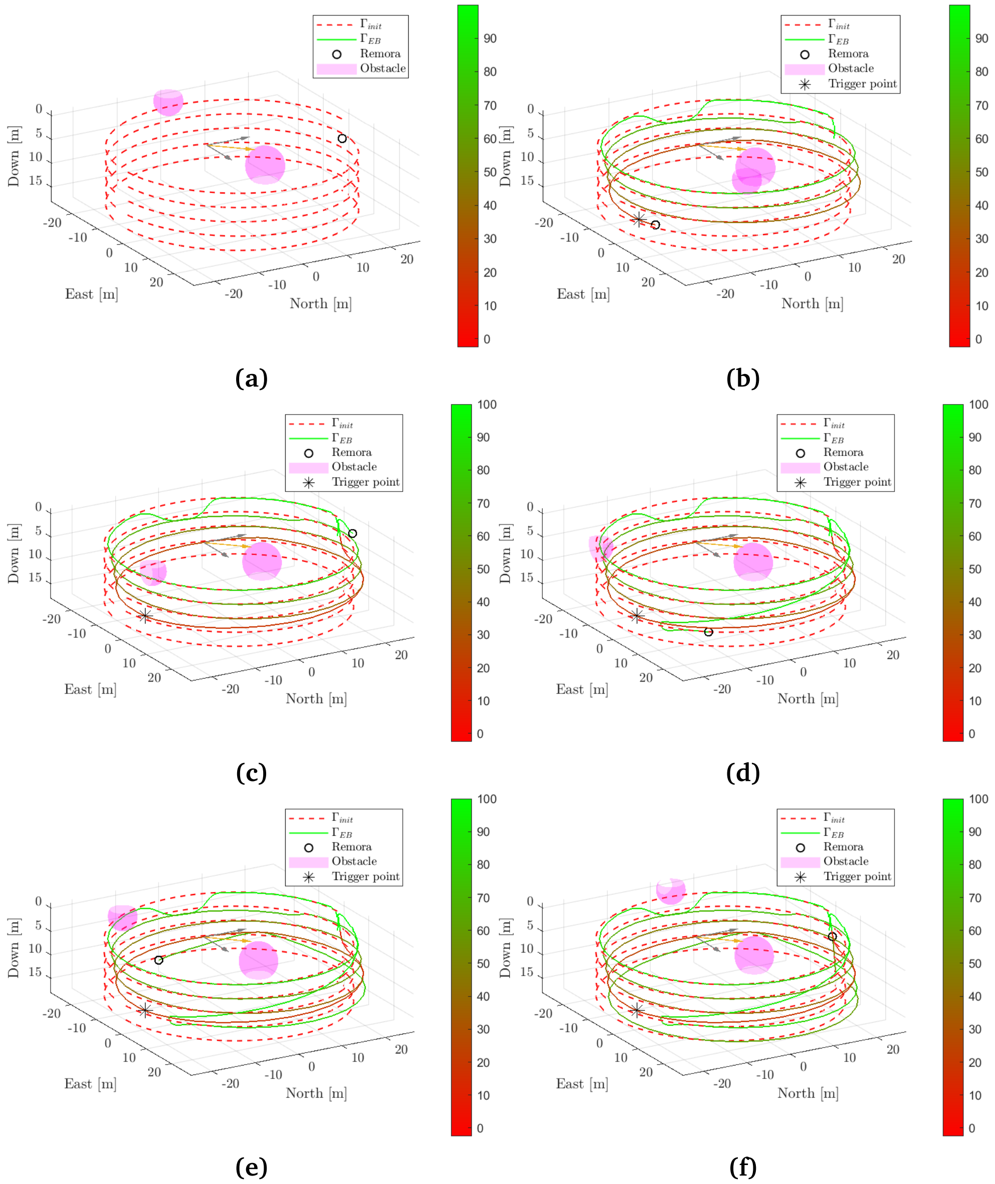


(a) (b)

(c) (d)

Figure A.5: Illustrating the animation in Figure 4.14.

A.2.3 Remora - Case 6



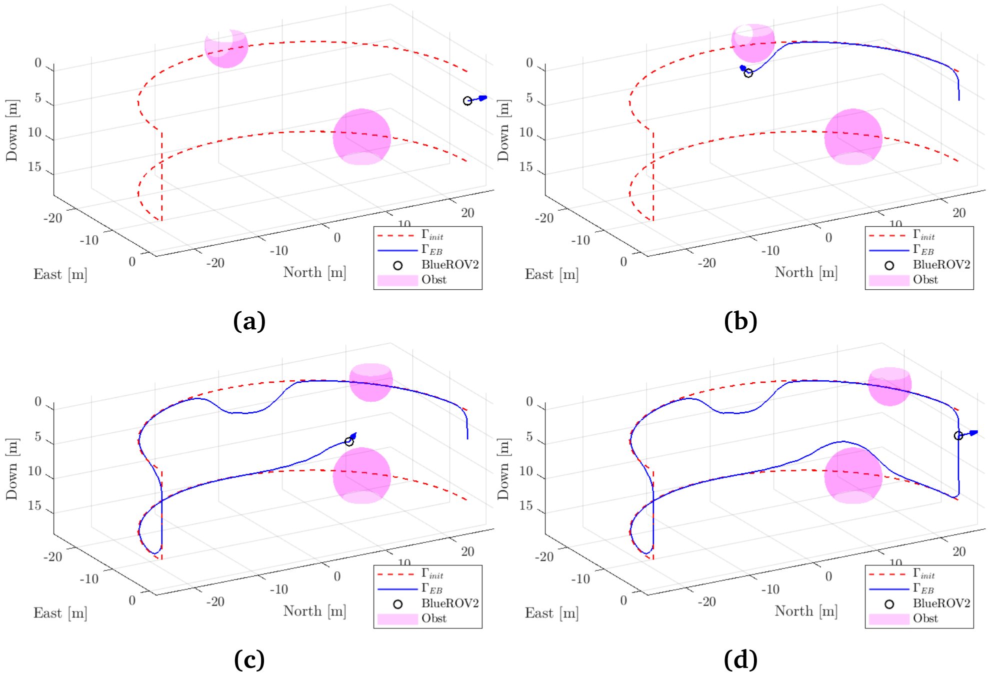
(a) **(b)**

(c) **(d)**

(e) **(f)**

Figure A.6: Illustrating the animation in Figure 4.16.

A.2.4 BlueROV2 - Case 3



(a)

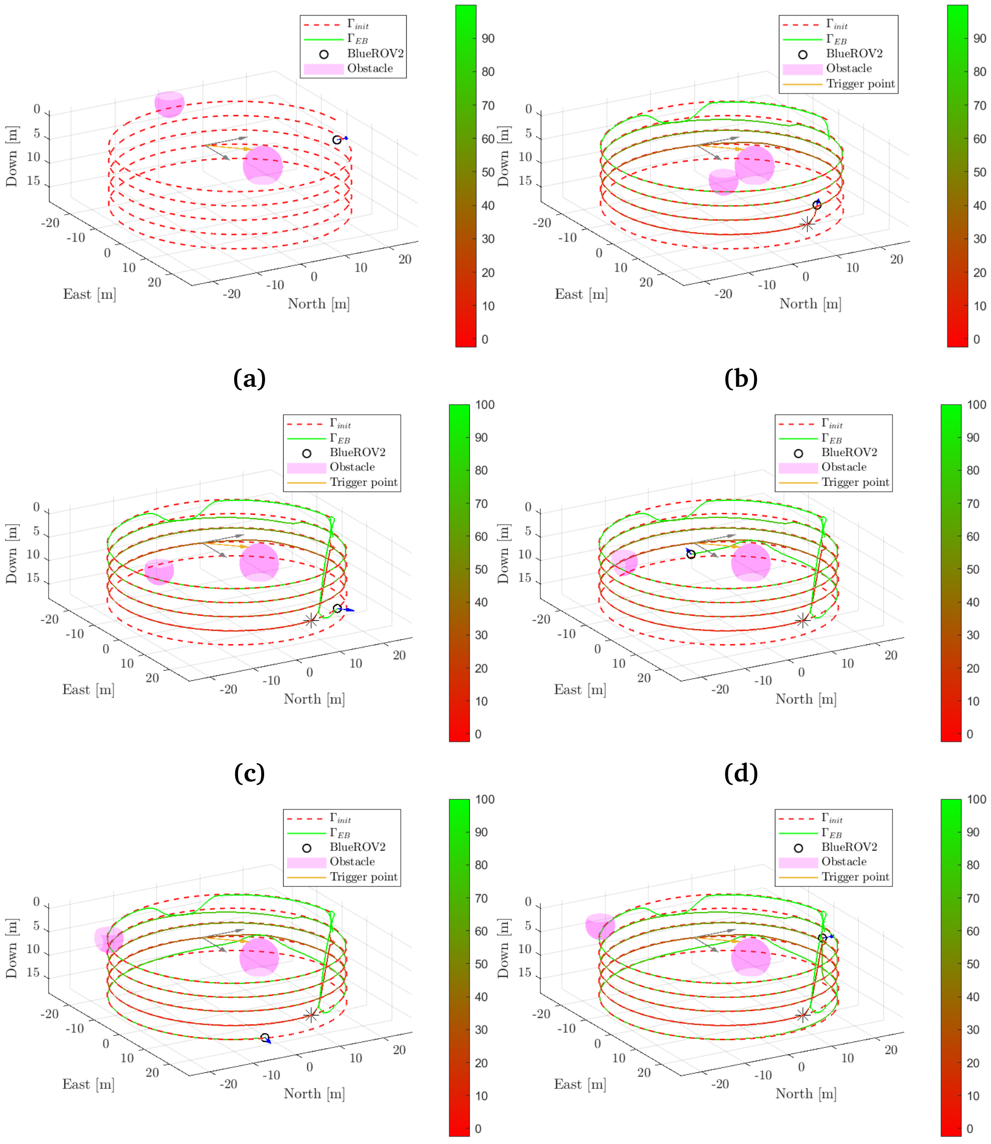
(b)

(c)

(d)

Figure A.7: Illustrating the animation in Figure 4.22.

A.2.5 BlueROV2 - Case 5



(a) **(b)** **(c)** **(d)** **(e)** **(f)**

Figure A.8: Illustrating the animation in Figure 4.25.

Appendix B

Abstract submitted to OCEANS 2021

The abstract sent to OCEANS 2021 [56] is attached below. A full paper will be submitted later with the results obtained in this thesis. This also gives a good basis for combining theoretical work from this thesis with the planned lab trials for a journal publication in the future.

Title: Adaptive motion planning and path following for permanent resident biofouling prevention robot operating in fish farms

Authors:

Sverre Fjæra (NTNU ITK), Sveinung Ohrem (SINTEF Ocean), Martin Føre (NTNU ITK), Nina Bloecher (SINTEF Ocean), Eleni Kelasidi (SINTEF Ocean)

Abstract

This paper presents a study where the potential of achieving autonomous net cleaning operations in commercial sea-cages is demonstrated through numerical simulations. Biofouling, the unwanted growth of organisms (e.g. hydroids, mussels, algae) at artificial substrates, is a challenge for the marine salmon farming industry (Bannister et al., 2019). Fouling of net cages is particularly challenging, as it may lead to various challenges such as reduced flow through the net wall (and hence cage water exchange), and increased net weight. These factors may affect production efficiency, fish welfare and the risk of wear resulting in unwanted incidents. While biofouling management is today primarily handled through periodic manual high-pressure cleaning, there is a desire to explore approaches where fouling levels are kept persistently low by autonomous devices continously grooming the net. The benefits of such methods include: 1) reducing personell needs and hence HSE risks during cleaning operations, and 2) limiting the fish's exposure to high levels of potentially harmful waste during cleaning. These benefits may become particularly important in the future due to ongoing trends in moving operations to more exposed locations with more challenging environmental conditions (Bjelland et al., 2014). SINTEF Ocean Mithal AS and other partners[1], have an ongoing research project called NetClean24/7 exploring the possibilities of achieving autonomous cleaning in fish farms. The present work focuses on the development of a control framework using adaptive motion planning and control strategies to achieve autonomous biofouling prevention operation in net pens.

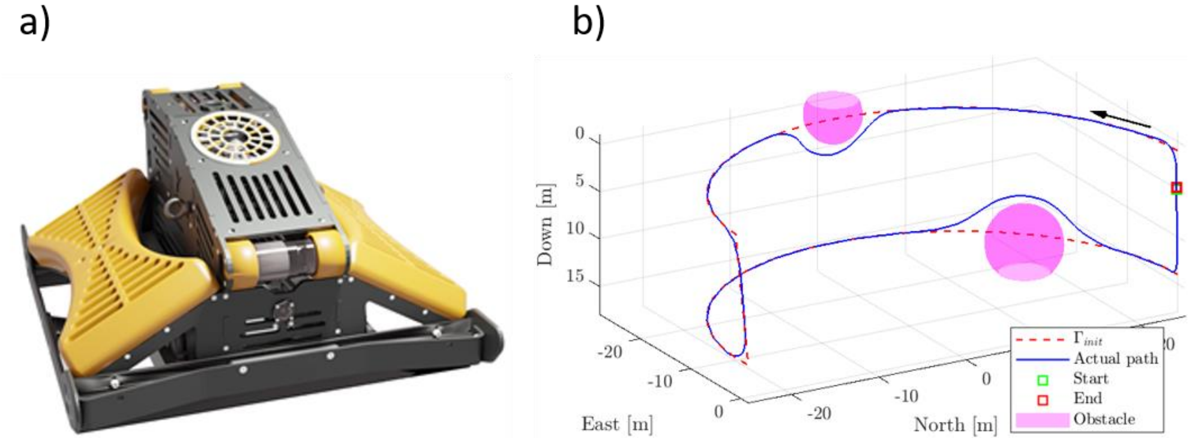


Figure 1: a) Remora biofouling prevention and net cleaning robot developed by Mithal AS, b) Simulation of the vehicle with two static regions of avoidance. The algorithm adapts the path, but not completely away from the regions of avoidance. The resulting path appears to be feasible.

Implementation was done in the SINTEF software platform FhSim that offers numerical models of various sub-systems relevant for simulating aquaculture operations, including net-cages, ropes/cables, UW vehicles and fish (Reite et al., 2014). A mathematical model of Mithal's new biofouling prevention robotic concept Remora (Figure 1 a, Ohrem et al., submitted) was used in simulations to test if autonomous continuous net cleaning practices are possible using a specialized vehicle designed to crawl on the net in combination with the new control system

[1] https://www.sintef.no/en/projects/2019/netclean-247/

algorithms. Simulations were also done with a model of an ROV (BlueRov Inc.) to compare the performance of the Remora concept with that of more conventional cleaning solutions using free-swimming cleaning rigs, and to demonstrate the applicability of the control framework to different types of underwater robotic systems.

After a brief introduction to biofouling/cleaning, the paper presents different motion planning methods that could potentially be used in adaptive and interactive motion planning algorithms that cope with unforeseen challenges in dynamically changing environments such as fish farms. By evaluating their respective properties, the elastic band method (Lee and Tsai, 2011) was found most suitable for this application and was hence chosen for further implementation. The method was adapted and expanded to fit the specific requirements of robotic systems operating on deformable net structures. The paper elaborates these modifications and their integration in FhSim, and how they together with control strategies enabling path following and the vehicle model comprise a general control framework for autonomous cleaning operations. A series of stepwise simulation experiments for testing the method are then presented, starting with simulations to verify basic elements, before more complex operations resembling actual autonomous biofouling prevention and cleaning operations are simulated:

1. Grooming and cleaning of a plane net panel: The robot first plans an optimal trajectory to a destination. Static and dynamic obstacles are introduced, creating regions of avoidance the motion planner needs to adapt to and find an alternative feasible path.
2. Case 1 is expanded by the robot having to operate on a dynamic net structure subjected to ocean currents, and plan an optimal path to accomplish the mission and thus compensate for environmental disturbances while adapting the path of the vehicle to avoid collisions.
3. Like 2, but with added complexity in the vehicle needing to return to a docking station (provided as a fixed point) to recharge during the operation.

Robot performance was evaluated based on a set of criteria including the ability to replan the path in real-time, the error between desired and actual path, and various controller performance metrics. The performance of the Remora model was also compared with the ROV model to investigate eventual advantages in using a system crawling on the net. In future work, the authors will demonstrate fully autonomous navigation of robotic systems operating in lab and field trials using both the Remora and BlueROV robots.

Acknowledgment

This work was funded by the Research Council of Norway through the project: Tetherless robot for biofouling prevention and inspection in salmon farming (NetClean 24/7, RCN 296392).

References

Bannister, J., Sievers, M., Bush, F. and Bloecher, N., 2019. Biofouling in marine aquaculture: a review of recent research and developments. Biofouling, 35(6), pp.631-648.

Bjelland, H.V., Føre, M., Lader, P., Kristiansen, D., Holmen, I.M., Fredheim, A., Grøtli, E.I., Fathi, D.E., Oppedal, F., Utne, I.B. and Schjølberg, I., 2015, October. Exposed aquaculture in Norway. In OCEANS 2015-MTS/IEEE Washington (pp. 1-10). IEEE.

Lee, C.T. and Tsai, C.C., 2011, August. 3D collision-free trajectory generation using elastic band technique for an autonomous helicopter. In FIRA RoboWorld Congress (pp. 34-41). Springer, Berlin, Heidelberg.

Ohrem, S. J., Amundsen, H., and Kelasidi, E., Modelling and control of an underwater biofouling prevention and inspection robot. In IROS 2021 (Submitted for publication).

Reite, K.J., Føre, M., Aarsæther, K.G., Jensen, J., Rundtop, P., Kyllingstad, L.T., Endresen, P.C., Kristiansen, D., Johansen, V. and Fredheim, A., 2014, June. Fhsim—time domain simulation of marine systems. In ASME 2014 33rd International Conference on Ocean, Offshore and Arctic Engineering. American Society of Mechanical Engineers Digital Collection.

Sverre Fjæra

Optimal and Adaptive Path Planning and Following for Permanent Resident Cleaning Robot Operating in Fish Farms

NTNU
Norwegian University of
Science and Technology