NTNU – Trondheim
Norwegian University of
Science and Technology

Horizontal Collision Avoidance for Autonomous Underwater Vehicles

Bjørn-Olav Holtung Eriksen

Master of Science in Cybernetics and Robotics

Submission date:	June 2015
Supervisor:	Kristin Ytterstad Pettersen, ITK
Co-supervisor:	Martin Syre Wiig, FFI Thomas Røbekk Krogstad, FFI

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem formulation

Autonomous underwater vehicles (AUVs) can be used in operations where manned presence is either unwanted of impossible. Examples of such operations are oceanographic research in remote areas, mine countermeasures as part of a military operation, or seabed surveying for the oil and gas industry. Future AUVs will have the capability to operate for weeks or months in areas with little or no infrastructure, unknown environment and limited if any operator communication. In such scenarios vehicle robustness and safety are important. A key enabling element for such operations is collision avoidance.

The HUGIN series of vehicles is equipped with a forward looking sonar (FLS) that has 120 vertically aligned beams. The FLS is used for terrain following, and for collision avoidance in the vertical dimension. The collision avoidance algorithm makes the vehicle circle upwards when an obstacle is met, in order to pass above the obstacle. This works well for missions in deep waters where there is room to maneuver in the vertical dimension. However, in shallow waters or when presented with tall obstacles (possibly reaching to the surface) it is more desirable to go around the obstacle while maintaining depth or altitude. Such horizontal collision avoidance is the subject of this MSc project.

A preliminary project has been completed, where two different strategies for horizontal collision avoidance were investigated. One of these was the Dynamic window algorithm. The task for this project is to further investigate the Dynamic window algorithm. The following tasks are to be done in the thesis:

- Investigate the Dynamic window method in depth:
  - Further investigate what extensions that has been made to the algorithm.
  - Is it possible to predict the vehicle trajectory as a more complex path than a circular arc?
  - Can the time varying acceleration limits be taken into account?
  - Are there other optimization criteria or methods that provide good results?
  - Is it possible to control the obstacle clearance?
  - Will the algorithm avoid collisions in local minima?
- Introduce constant irrotational ocean current in the scenario. How does this affect the performance?
- Can the Dynamic window algorithm be modified to handle constant irrotational ocean current?
- Investigate if Lyapunov theory can be used to prove stability for the collision avoidance system.

Preface

Technology, in particular autonomous systems, have always intrigued me. The idea of making a system capable of controlling itself, and enabling it to base decisions on external variables is truly fascinating. Autonomous systems will take more and more presence in our lives, and one can only imagine what will be possible in the future. Collision avoidance is a necessity for enabling autonomous navigation in unknown environments.

This master thesis is written as a compulsory part of the two-year MSc programme in Engineering Cybernetics at NTNU. It has consumed a great deal of time, but in return rewarded me with new knowledge, experience and (hopefully) a MSc degree in Engineering Cybernetics.

I would like to thank my supervisors, especially Kristin Y. Pettersen at NTNU and Martin Syre Wiig at FFI, for their help and support throughout the project. They have steered me away from collisions; thus humans can be said to inherit excellent collision avoidance capabilities. In addition, I would like to express my gratitude to my fellow students for the time we spent together at NTNU, and in particular Sveinung, Thomas and Stine from the office for all the interesting discussions.

Bjørn-Olav Holtung Eriksen
Trondheim, June 1, 2015

Abstract

Vast amounts of research has been done on the subject of collision avoidance. However, limited effort has been put into adapting collision avoidance algorithms to vehicles with second order non-holonomic constraints, such as Autonomous underwater vehicles (AUVs). This thesis assesses the Dynamic window algorithm applied to a HUGIN 1000 AUV for reactive horizontal collision avoidance, with constant irrotational ocean current. A simulator for the AUV with sonar sensors and an integral line of sight (ILOS) guidance system has been developed, and is used to implement and test the algorithm.

The original Dynamic window algorithm is not intended for use on vehicles with second order non-holonomic constraints. A thorough literature study on modifications and extensions to the algorithm is presented. A number of modifications have been made to the original algorithm to make it better suited for use with the HUGIN 1000 AUV. In particular, a new method for predicting AUV trajectories which accounts for second order non-holonomic constraints and ocean current has been developed.

Using Lyapunov theory, convergence to a straight line path is proved with UGAS and ULES stability properties under the assumption that no obstacles are present. The proof include ocean current, and is a first step of a complete stability proof for the collision avoidance system. Simulations support the proof. The performance of the modified Dynamic window algorithm has been assessed through simulations. When not faced with local minima, collisions are avoided both with and without ocean current. The simulations infer that the algorithm also avoids collision when faced with local minima, but due to inaccuracies in the simulation model no definitive conclusion can be made.

The new trajectory prediction method reduces the mean square prediction error to about one percent compared to the original method, and makes the Dynamic window algorithm well suited for vehicles with second order non-holonomic constraints such as the HUGIN 1000 AUV.

Sammendrag

Svært mye forskning har blitt gjort på kollisjonsunngåelse for roboter. Imidlertid har lite blitt gjort for å tilpasse kollisjonsunngåelsesmetoder til roboter med andre ordens ikke-holonomiske beskrankninger, som for eksempel Autonome undervannsfarkoster (AUVer). Denne masteroppgaven ser på en antikollisjonsmetode kalt "Dynamic window algorithm" for bruk på en HUGIN 1000 AUV til et reaktivt antikollisjonssystem i horisontalplanet, under påvirkning av konstant ikke-roterende havstrøm. Et simuleringssystem for AUVen sammen med sonarer og et "Integral line of sight" (ILOS) navigasjonssystem har blitt utviklet, og er brukt til å implementere og teste algoritmen.

Den originale Dynamic window algoritmen er ikke tiltenkt brukt på roboter med andre ordens ikke-holoniske beskrankninger. Et omfattende litteraturstudie med fokus på modifikasjoner tidligere gjort på algoritmen er presentert. For å gjøre algoritmen bedre egnet for bruk på HUGIN 1000 har en del modifikasjoner blitt gjort. Især har en ny metode for å predikere banene til AUVen, som tar hensyn til andre ordens ikke-holonomiske beskrankninger og havstrømmer, blitt utviklet.

Ved bruk av Lyapunovteori har konvergens til en rettlinjet bane blitt bevist med UGAS og ULES stabilitetsegenskaper. Beviset inkluderer havstrøm, og antar at ingen hindringer er tilstede. Dette beviset er et første steg i et komplett stabilitetsbevis for antikollisjonssystemet. Simuleringer understøtter beviset. Ytelsen til den modifiserte Dynamic window algoritmen er vurdert gjennom en rekke simuleringer. Når AUVen ikke blir presentert med lokale minimum viser simuleringene at kollisjoner unngås, både med og uten havstrøm. Simuleringene antyder også at kollisjon unngås i lokale minimum, men grunnet unøyaktigheter i simuleringsmodellen kan ingen klar konklusjon trekkes.

Den nye prediksjonsmetoden for AUVens baner reduserer den gjennomsnittlige kvadratiske prediksjonsfeilen til omtrent en prosent sammenlignet med den originale metoden, og gjør Dynamic Window algoritmen godt egnet for bruk på roboter med andre ordens ikke-holonomiske beskrankninger, som HUGIN 1000.

Contents

List of Figures **xiii**

List of Tables **xv**

1 Introduction **1**

1.1 Motivation 1

1.2 Objective and scope 2

1.3 Contributions 2

1.4 Outline 3

1.5 Notation 3

1.6 Abbreviations 5

2 Theoretical background **7**

2.1 Mathematical modeling 7

2.1.1 6DOF hydrodynamical model 7

2.1.2 Ocean current modeling 10

2.1.3 Actuator model 11

2.2 Collision avoidance theory 12

2.3 The Dynamic window algorithm 13

2.3.1 Original method 13

2.3.2 Proposed extensions 17

3 Modifications to the Dynamic window algorithm **21**

3.1 Control plant model 21

3.2 Environment modeling 24

3.3 Search space 26

3.4 Trajectory prediction 31

3.5 Objective function 39

3.A Appendix: Functional expressions 41

3.B Appendix: Trajectory prediction without pivot point transformation 42

4 Simulator development **49**

4.1 Simulator overview 49

4.2 AUV model 49

4.3 Sonar modeling 51
4.3.1 Sonar configuration 51
4.3.2 Seabed modeling 52
4.3.3 Single beam sonars 52
4.3.4 Horizontal forward looking sonar 56
4.3.5 Side scan sonars 57
4.4 Horizontal guidance system 58
4.5 Feedback controllers 59
4.5.1 Surge speed and yaw rate controller 59
4.5.2 Yaw controller 61
4.5.3 Depth controller 63
4.6 Dynamic window algorithm 63
4.6.1 Environment representation 63
4.6.2 Search space and predicted trajectories 64
4.6.3 Velocity pair selection 68

5 Stability analysis 71
5.1 AUV model in component form 71
5.2 Control objective 72
5.3 Control system 72
5.4 Stability of the closed loop system 74
5.4.1 Proof of theorem 5.1 74
5.A Appendix: Functional expressions 77
5.B Appendix: Proof of lemma 5.1 78

6 Simulation results 81
6.1 Test cases 81
6.2 Test case 1 84
6.2.1 Comparison of the old and new Dynamic window implementation 84
6.2.2 Performance with ocean current 85
6.3 Test case 2 88
6.3.1 Comparison of the old and new Dynamic window implementation 91
6.3.2 Performance with ocean current 94
6.4 Test case 3 95
6.5 Test case 4 97
6.6 Simulations in randomly generated environments 98

7 Concluding remarks and suggestions for future work 101

Appendices 105

A Stability of cascaded systems 107

B Simulations in randomly generated environments 109

List of Figures

2.1 The HUGIN 1000 AUV 8

3.1 AUV heading is part of $\mathcal{C}_{free}$ and $\mathcal{C}_{obs}$ 25
3.2 Circular approximation of the AUV footprint 25
3.3 Expanded obstacles to collapse the configuration space 26
3.4 Antitarget and avoid region 26
3.5 Possible combinations of surge speed and yaw rate, with respect to actuator saturation limits 30
3.6 Function to find possible velocities 30
3.7 The dynamically feasible velocity set, together with the boundaries of the dynamic velocity window and the possible velocity set. . . . 31
3.8 Actual and approximated AUV trajectories, given initial velocity $\bar{\boldsymbol{\nu}}_r(t_0) = [2 \quad 0 \quad 0]^T$ 38

4.1 Simulator overview 50
4.2 HUGIN 1000 sonar sensors 53
4.3 Example DTM matrix 53
4.4 Seabed map of Jesusbukta in the Breiangen area 54
4.5 Search lines of a single HFLS beam 56
4.6 ILOS guidance law 59
4.7 Steady state ILOS heading 60
4.8 Step response of closed loop surge dynamics 62
4.9 Step response of closed loop yaw rate dynamics 62
4.10 Step response of closed loop yaw dynamics 63
4.11 Step response of closed loop depth dynamics 64
4.12 Local obstacle map. The estimated obstacle boundary is marked with a red line. 65
4.13 A discrete set of dynamically feasible velocities 66
4.14 AUV distances $\bar{\rho}_i$ and ρ_i given the trajectory for velocity pair i . . 68
4.15 The Dynamic window objective function parts (a)-(c) and the combined objective function (d). 69
4.16 Predicted AUV trajectories with the selected trajectory in yellow . 70

6.1 Simulation environments 82

6.2 AUV trajectories for test case 1 84
6.3 Desired and actual surge speed and yaw rate, case 1, new Dynamic window implementation 86
6.4 Desired and actual surge speed and yaw rate, case 1, old Dynamic window implementation 86
6.5 Actuator usage, case 1, new Dynamic window implementation . . . 87
6.6 Actuator usage, case 1, old Dynamic window implementation . . . 87
6.7 Desired and actual surge speed and yaw rate, case 1, correct current information 89
6.8 Desired and actual surge speed and yaw rate, case 1, incorrect current information 89
6.9 Actuator usage, case 1, correct current information 90
6.10 Actuator usage, case 1, incorrect current information 90
6.11 AUV trajectories for test case 2 91
6.12 Desired and actual surge speed and yaw rate, case 2, new Dynamic window implementation 92
6.13 Desired and actual surge speed and yaw rate, case 2, old Dynamic window implementation 93
6.14 Actuator usage, case 2, new Dynamic window implementation . . . 93
6.15 Actuator usage, case 2, old Dynamic window implementation . . . 94
6.16 Cross track error and AUV heading, case 2, with correct and wrong current information 95
6.17 AUV trajectories, test case 3, with and without ocean currents . . 96
6.18 AUV trajectories for test case 4 97
6.19 Prediction error using linear and circular trajectory prediction, case 4 98
6.20 Passive response to sway and yaw dynamics 100

List of Tables

3.1 Mean square error (MSE) and Root mean square error (RMSE) of predicted AUV trajectories . . . 37

4.1 Position and orientation of sonar sensors . . . 52

6.1 Simulation parameters . . . 83
6.2 Trajectory data, case 1, old and new Dynamic window implementation . . . 85
6.3 Trajectory data, case 1, with ocean current . . . 88
6.4 Trajectory data, case 2, old and new Dynamic window implementation . . . 92
6.5 Trajectory data, case 2, with ocean current . . . 94
6.6 Trajectory data, case 3 . . . 96
6.7 Prediction error of the predicted AUV trajectories, case 4 . . . 98
6.8 Summary of trajectories in randomly generated environments . . . 99

B.1 Trajectory data for simulations in randomly generated environments110

Chapter 1

Introduction

1.1 Motivation

Collision avoidance is a necessity for safe autonomous operation of vehicles, Autonomous underwater vehicles (AUVs) included. Only a small percentage of the oceans have been mapped to a detail applicable for planning collision free AUV trajectories, hence a planned route may very well result in collisions if it were to be followed without local adaption. Clearly, once a collision occurs, and assuming that the AUV structure is breached, then it is inevitable that a catastrophic failure will follow [Tan et al., 2004a].

The topic of collision avoidance for AUVs is complicated, but may be split into two main subjects; *obstacle detection* which focuses on detecting obstacles usually based on sonar data, and *obstacle avoidance* which consists of finding a collision-free trajectory around detected obstacles [Tan et al., 2004a]. A *collision avoidance system* must contain functionality for both obstacle detection and obstacle avoidance in order to avoid collisions. The main focus in this thesis will be obstacle avoidance. For obstacle detection, it is proposed to use the sonars currently employed on the HUGIN. The HUGIN series of vehicles is equipped with a number of sonars, including a forward looking sonar (FLS) that has 120 vertically aligned beams. By rotating the existing FLS 90 degrees it can be used as a Horizontal forward looking sonar (HFLS), and detect obstacles in the horizontal plane.

The collision avoidance system currently employed on the HUGIN 1000 AUV avoids obstacles by moving in the vertical dimension. At given times, for example when presented with tall obstacles or when operating in shallow waters, it is more desirable to avoid obstacles by moving in the horizontal dimension. The work in this thesis is a continuation of [Eriksen, 2014], where the Edge Following [Tan et al., 2004b] and the Dynamic window algorithm [Fox et al., 1997] were compared for use as a horizontal obstacle avoidance algorithm. The Dynamic window algorithm was considered superior to the Edge Following algorithm. It has, however, some shortcomings for use on vehicles with second order non-holonomic constraints and time varying acceleration limits, such as AUVs. The focus of this thesis is to investigate the Dynamic window algorithm in depth, and modify it to improve the algorithm for use with AUVs.

1.2 Objective and scope

The following subtasks are proposed:

1. Perform a literature study on extensions and modifications made to the Dynamic window algorithm.
2. Investigate possible modifications to the Dynamic window algorithm to make it more suited for application on an AUV:
    - Improve the accuracy of the AUV trajectory prediction.
    - Modify the search space to include time varying acceleration limits.
    - Can the the objective function be modified to improve the algorithm?
    - Modify the algorithm to achieve a desired minimum obstacle clearance.
    - Can the Dynamic window algorithm be modified to account for ocean currents?
3. Extend the MATLAB\SIMULINK simulation environment in [Eriksen, 2014]:
    - Include constant irrotational ocean current.
    - Modify the Dynamic window algorithm to include the possible modifications.
4. Attempt to prove stability for the collision avoidance system.
5. Simulate the collision avoidance system in different scenarios to demonstrate strengths and weaknesses.

To simplify the tasks, noiseless measurements are assumed. Furthermore, since the focus of the thesis is horizontal collision avoidance, all obstacles are assumed to be tall with vertical faces. Hence, horizontal collision avoidance is always preferred. In a practical application some mechanism for choosing between a vertical or horizontal approach must be implemented.

1.3 Contributions

The contributions of this thesis is:

- A review of previous extensions and modifications to the Dynamic window algorithm.
- A series of modifications to the Dynamic window algorithm:
  - A new method of partially linearizing the AUV dynamics and predicting the AUV trajectories using a linear approximation of the AUV dynamics. This new method greatly improves the accuracy of the predicted AUV trajectories and takes constant irrotational ocean current into account.
  - A new search space definition accounting for the time varying acceleration limits of the AUV.
  - A new objective function which makes the algorithm more general.
- A Lyapunov based stability proof of the collision avoidance system in conjunction with an ILOS guidance law, under the assumption that no obstacles are present. The proof includes constant irrotational ocean current.
- Thorough testing of the algorithm through a number of simulations.

1.4 Outline

This thesis is divided into several distinctive chapters.

Chapter 2 presents mathematical background and theory for vehicle modeling and collision avoidance, and a literature review of the Dynamic window algorithm. Chapter 3 contains derivations of the modifications made to the Dynamic window algorithm. The development of the simulator used to simulate the AUV, including sonars and collision avoidance system is presented in chapter 4. A Lyapunov based stability proof for the collision avoidance system is conjunction with a horizontal guidance and control system is presented in chapter 5. Chapter 6 contains data and comments from the conducted simulations. Some concluding remarks, and suggestions for future work is presented in chapter 7.

1.5 Notation

The notation list is intended as a reference for the reader. Symbols only used in a small part of the thesis is not included in the notation list. All symbols are explained as they are introduced throughout the thesis. Bold symbols, e.g $\boldsymbol{\eta}$, denote a vector or a matrix.

Symbol	**Comment**
$\boldsymbol{\eta}$	AUV pose
$\boldsymbol{\nu}$	Body velocity
$\boldsymbol{\nu}_r$	Relative body velocity
$\boldsymbol{\nu}_c$	Ocean current velocity in body frame
$\boldsymbol{V}_c$	Ocean current in inertial frame
$\boldsymbol{\tau}$	Force vector
$\boldsymbol{J}(\boldsymbol{\eta})$	Velocity transformation matrix
$\boldsymbol{M}$	Mass matrix
$\boldsymbol{C}(\boldsymbol{\nu})$	Coriolis-centripetal matrix
$\boldsymbol{D}(\boldsymbol{\nu})$	Damping matrix
$\boldsymbol{g}(\boldsymbol{\eta})$	Vector of restoring forces matrix
$\boldsymbol{\delta}$	Rudder deflection vector
δ_{yaw}	Yaw rudder deflection
δ_{pitch}	Pitch rudder deflection
n_p	Propeller speed
$\boldsymbol{B}$	Input matrix
$\boldsymbol{f}$	Force vector
$\boldsymbol{H}$	Pivot point transformation matrix
$\boldsymbol{\Gamma}_1$	Separation matrix
$\boldsymbol{\Gamma}_2$	Separation matrix
$\boldsymbol{n}(\boldsymbol{\nu})$	Nonlinear forces
$\boldsymbol{N}$	Jacobian matrix
$\boldsymbol{A}$	State matrix
$\boldsymbol{\beta}$	Input matrix
$\boldsymbol{G}$	Bias vector
$X(u_r)$	Force coefficient in sway
$Y(u_r)$	Force coefficient in sway
Δ	ILOS lookahead distance
σ	ILOS integral gain constant
k_u	Surge controller gain
k_r	Yaw rate controller gain
k_ψ	Yaw controller gain
α	Dynamic window yaw rate scaling constant
β	Dynamic window distance scaling constant
γ	Dynamic window surge speed scaling constant
$\mathbb{C}$	Configuration space
$\mathbb{W}$	Workspace
$\mathcal{T}$	Antitarget region
Ω	Avoid region

To separate small variations of variables, accents are used.

Accent	e.g	Comment
Bar	$\bar{\boldsymbol{\nu}}$	General variable separation. When used with variables concerning the AUV model, e.g $\bar{\boldsymbol{\eta}}, \bar{\boldsymbol{\nu}}, \bar{\boldsymbol{M}}, \bar{\boldsymbol{C}}, \bar{\boldsymbol{\tau}}$, it refers to the variable being used in conjunction with the *pivot point*.
Tilde	$\tilde{\boldsymbol{\nu}}$	Error variable.
Prime	$\boldsymbol{n}'$	General variable separation.

1.6 Abbreviations

The list of abbreviations is intended as a reference for the reader. Abbreviations are explained as they are introduced throughout the thesis.

Abbreviation	Comment
AUV	Autonomous underwater vehicle
CB	Center of buoyancy
CG	Center of gravity
CO	Center of origin
DOF	Degrees of freedom
DTM	Digital terrain model
DVL	Doppler velocity log
DWA	Dynamic window algorithm
FD*	Focused D*
FTS	Forward tilted sonar
FLS	Forward looking sonar
GDWA	Global dynamic window approach
HFLS	Horizontal forward looking sonar
ILOS	Integral line of sight
LFC	Lyapunov function candidate
LOS	Line of sight
MPC	Model predictive control
MSE	Mean square error
NED	North east down
NF1	Neuro-Fibromatosis type-1
RMSE	Root mean square error
SSS	Side scan sonar
UGAS	Uniform global asymptotic stability
ULES	Uniform local exponential stability
VS	Vertical sonar

Chapter 2

Theoretical background

2.1 Mathematical modeling

The North east down (NED) reference frame $\{n\}$ is used for modeling the AUV kinematics. For marine vehicles operating in a local area at low velocities, $\{n\}$ can be considered to be inertial [Fossen, 2011]. In addition, the local body $\{b\}$ frame is used for modeling the AUV dynamics. The origin of $\{b\}$ is located in the center of buoyancy (CB). The model is presented in 6 degrees of freedom (DOF).

The AUV considered in this thesis is the HUGIN 1000 AUV. The HUGIN 1000 is dedicated towards naval applications where operations close to the seabed is important [Kongsberg Maritime, 2014]. The operation space may also include shallow waters, possibly with the presence of surface ice. This can make the approach of passing obstacles in the vertical plane infeasible. A picture of the HUGIN AUV is shown in Figure 2.1.

2.1.1 6DOF hydrodynamical model

The mathematical model used in this project is developed by the HUGIN 1000 design team, using the theory in [Fossen, 2011]. It is structured in a vectorial form, given in (2.1). The model is valid for positive surge velocities.

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu} \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{g}(\boldsymbol{\eta}) &= \boldsymbol{\tau} \end{aligned} \tag{2.1}$$

Where,

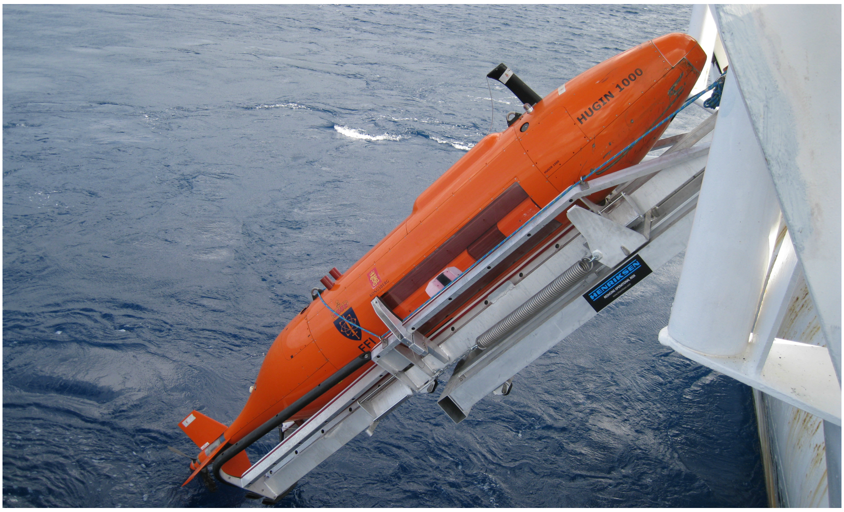


Figure 2.1: The HUGIN 1000 AUV

$\boldsymbol{\eta}$ NED pose, $\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{p}_{b/n}^{n^T} & \boldsymbol{\Theta}_{nb}^T \end{bmatrix}^T = \begin{bmatrix} N & E & D & \phi & \theta & \psi \end{bmatrix}^T$.

$\boldsymbol{\nu}$ Body velocity, $\boldsymbol{\nu} = \begin{bmatrix} \boldsymbol{v}_{b/n}^{b^T} & \boldsymbol{\omega}_{b/n}^{b^T} \end{bmatrix}^T = \begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^T$.

$\boldsymbol{\nu}_r$ Relative body velocity, $\boldsymbol{\nu}_r = \boldsymbol{\nu} - \boldsymbol{\nu}_c$, where $\boldsymbol{\nu}_c$ is the ocean current velocity in $\{b\}$.

$\boldsymbol{\tau}$ Generalized forces, $\boldsymbol{\tau} = \begin{bmatrix} X & Y & Z & K & M & N \end{bmatrix}^T$.

$\boldsymbol{J}(\boldsymbol{\eta})$ Transformation matrix from {b} velocity to {n} velocity, given in (2.2a).

$\boldsymbol{M}$ Mass matrix, given in (2.2b).

$\boldsymbol{C}(\boldsymbol{\nu})$ Coriolis-centripetal matrix, given in (2.2c).

$\boldsymbol{D}(\boldsymbol{\nu}_r)$ Damping matrix, given in (2.2d).

$\boldsymbol{g}(\boldsymbol{\eta})$ Restoring forces vector, given in (2.2e).

The SNAME [SNAME, 1950] notation is used.

The transformation matrix $\boldsymbol{J}(\boldsymbol{\eta})$ is defined as:

$$\boldsymbol{J}(\boldsymbol{\eta}) = \begin{bmatrix} \boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb}) & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{T}_{\Theta}(\boldsymbol{\Theta}_{nb}) \end{bmatrix} \tag{2.2a}$$

Where $\boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb})$ is the rotation matrix from $\{n\}$ to $\{b\}$ and $\boldsymbol{T}_{\Theta}(\boldsymbol{\Theta}_{nb})$ is the angular velocity transformation matrix from $\{b\}$ to $\{n\}$. It is worth noting that $\boldsymbol{T}_{\Theta}(\boldsymbol{\Theta}_{nb})$ has a singularity at $\theta = \frac{\pi}{2} + n\pi, n \in \mathbb{Z}$. This is a limitation when using Euler angles to parameterize the orientation, and can be avoided by using Euler parameters or quaternions as alternative parameterizations [Egeland and Gravdahl, 2003]. The AUV will, however, never operate close to the singular points. Therefore Euler angles are considered as suited for parameterizing the

AUV orientation.

The mass matrix $\boldsymbol{M}$ is:

$$\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A = \underbrace{\begin{bmatrix} m\boldsymbol{I}_{3x3} & -m\boldsymbol{S}(\boldsymbol{r}_g^b) \\ m\boldsymbol{S}(\boldsymbol{r}_g^b) & \boldsymbol{I}_b \end{bmatrix}}_{\boldsymbol{M}_{RB}} + \underbrace{\begin{bmatrix} X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} \\ Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} \\ Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} \\ K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} \\ M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} \\ N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}} \end{bmatrix}}_{\boldsymbol{M}_A} \tag{2.2b}$$

Where $\boldsymbol{M}_{RB}$ is the rigid-body mass matrix, $\boldsymbol{M}_A$ contains the added mass. m is the AUV mass, $\boldsymbol{I}_b$ is a matrix of inertia in CB and $\boldsymbol{r}_g^b = \begin{bmatrix} x_g & y_g & z_g \end{bmatrix}^T$ is the location of the center of gravity (CG) given in $\{b\}$. $\boldsymbol{S}(\boldsymbol{\lambda})$ is a skew-symmetric matrix, defined as $\boldsymbol{S}(\boldsymbol{\lambda}) = \begin{bmatrix} 0 & -\lambda_3 & \lambda_2 \\ \lambda_3 & 0 & -\lambda_1 \\ -\lambda_2 & \lambda_1 & 0 \end{bmatrix}$, $\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 \end{bmatrix}^T$. It corresponds to the cross-product operator $\boldsymbol{\lambda} \times \boldsymbol{a} = \boldsymbol{S}(\boldsymbol{\lambda})\boldsymbol{a}$. The mass matrix is symmetric and positive definite, $\boldsymbol{M} = \boldsymbol{M}^T > 0$ [Fossen, 2011].

The Coriolis-centripetal matrix can be parameterized in many ways. A skew-symmetric property, $\boldsymbol{C} = -\boldsymbol{C}^T$, can be achieved by choosing [Fossen, 2011]:

$$\boldsymbol{C}(\boldsymbol{\nu}) = \begin{bmatrix} \boldsymbol{0}_{3\times3} & -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{v}_{b/n}^b + \boldsymbol{M}_{12}\boldsymbol{\omega}_{b/n}^b) \\ -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{v}_{b/n}^b + \boldsymbol{M}_{12}\boldsymbol{\omega}_{b/n}^b) & -\boldsymbol{S}(\boldsymbol{M}_{21}\boldsymbol{v}_{b/n}^b + \boldsymbol{M}_{22}\boldsymbol{\omega}_{b/n}^b) \end{bmatrix} \tag{2.2c}$$

Where $\boldsymbol{M}_a \in \mathbb{R}^{3\times3}$, $a \in \{11, 12, 21, 22\}$ are defined as $\boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{M}_{12} \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} \end{bmatrix}$.

The damping matrix $\boldsymbol{D}(\boldsymbol{\nu}_r)$ is:

$$\boldsymbol{D}(\boldsymbol{\nu}_r) = -\begin{bmatrix} X_u + X_{|u|u}|u_r| & 0 & 0 & 0 & 0 & 0 \\ 0 & Y_v & 0 & 0 & 0 & Y_r \\ 0 & 0 & Z_w & 0 & Z_q & 0 \\ 0 & 0 & 0 & K_p & 0 & 0 \\ 0 & 0 & M_w & 0 & M_q & 0 \\ 0 & N_v & 0 & 0 & 0 & N_r \end{bmatrix} \frac{|u_r|}{u_0} \tag{2.2d}$$

Where $\frac{|u_r|}{u_0}$ is used for speed-scaling of the damping. To keep some damping at zero surge speed, the scaling term is approximated as $\frac{|u_r|}{u_0} \approx \frac{|u_r'|}{u_0}$, $|u_r'| = max(|u_r|, \mu)$ where $\mu > 0$ is a small constant.

CG and CB are vertically aligned in $\{b\}$. Furthermore, the HUGIN 1000 is naturally buoyant. The restoring forces is given as:

$$\boldsymbol{g}(\boldsymbol{\eta}) = \begin{bmatrix} 0 & 0 & 0 & \overline{BG}_z W c\theta s\phi & \overline{BG}_z W s\theta & 0 \end{bmatrix}^T \tag{2.2e}$$

Where $\overline{BG}_z$ is the z-component of a vector connecting CB and CG in $\{b\}$ and $W = mg$, where g is the acceleration of gravity, is the weight of the vehicle in air. The notation $c(\cdot) = \cos(\cdot)$ and $s(\cdot) = \sin(\cdot)$ is introduced to simplify the notation.

Assumption 2.1. *The relative velocity $\boldsymbol{\nu}_r$ and the pose vector $\boldsymbol{\eta}$ is considered to be measured accurately without noise and drift.*

Remark 2.1. *The relative velocity $\boldsymbol{\nu}_r$ can for example be measured using a Doppler Velocity Log (DVL) [Morgado et al., 2011]. The HUGIN 1000 AUV employs a DVL and a number of other sensors for navigation purposes. Therefore, $\boldsymbol{\nu}_r$ can be measured using the DVL. Further, an estimate of $\boldsymbol{\eta}$ is generated by a DVL aided inertial navigation system which also can interface other sensors [Jalving et al., 2004].*

2.1.2 Ocean current modeling

Assumption 2.2. *The ocean current is assumed to be constant, irrotational, unknown and bounded in $\{n\}$. It is also assumed to have no vertical component, hence it is given as $\boldsymbol{V}_c = \begin{bmatrix} V_N & V_E & 0 \end{bmatrix}^T$ in $\{n\}$. There exists a constant $V_{max} > 0$ such that $V_{max} \geq \|\boldsymbol{V}_c\|_2 = \sqrt{V_N^2 + V_E^2}$.*

Remark 2.2. *Modeling the ocean current as constant and irrotational in the inertial frame is widely accepted to describe the effects of slowly varying disturbances [Caharija, 2014].*

The ocean current is given in $\{b\}$ as:

$$\boldsymbol{\nu}_c = \begin{bmatrix} u_c \\ v_c \\ w_c \\ 0 \\ 0 \\ 0 \end{bmatrix} \triangleq \begin{bmatrix} \boldsymbol{\nu}_{c_1} \\ \boldsymbol{0}_{3\times1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb})\boldsymbol{V}_c \\ \boldsymbol{0}_{3\times1} \end{bmatrix} \tag{2.3}$$

To include the ocean current in the model, it is useful to model the relative velocity of the vehicle. It is possible to show that if the current is constant and irrotational in the inertial frame, the following identity holds [Fossen, 2011]:

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{M}\dot{\boldsymbol{\nu}}_r + \boldsymbol{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r \tag{2.4}$$

The vehicle equations of motion in (2.1) can then be written in terms of relative velocity as:

$$\begin{aligned}\dot{\boldsymbol{\eta}} &= \boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{\nu}_r + \begin{bmatrix}\boldsymbol{V}_c \\ \mathbf{0}_{3\times 1}\end{bmatrix} \\ \boldsymbol{M}\dot{\boldsymbol{\nu}}_r + \boldsymbol{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r &+ \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau}\end{aligned} \tag{2.5}$$

Where $\boldsymbol{\nu}_r = \begin{bmatrix}u_r & v_r & w_r & p & q & r\end{bmatrix}^T$.

2.1.3 Actuator model

The HUGIN 1000 is equipped with one propeller, and four rudders. The propeller can be controlled by specifying the propeller angular rate, while the rudders are controlled by specifying the deflection angles. All the rudders may be controlled separately, which makes it possible to assert roll, pitch and yaw moments. The propeller mainly produce surge force, but also some roll moment due to propeller drag. The resulting force can be modeled as $\boldsymbol{\tau} = \boldsymbol{\tau}_{propeller} + \boldsymbol{\tau}_{rudder}$.

The propeller is modeled as:

$$\boldsymbol{\tau}_{propeller} = \begin{bmatrix}T & 0 & 0 & Q & 0 & 0\end{bmatrix}^T \tag{2.6}$$

Where T and Q is given as:

$$T = T_{nn}|n_p|n_p + T_{un}u_r n_p \tag{2.7a}$$

$$Q = Q_{nn}|n_p|n_p + Q_{un}u_r n_p \tag{2.7b}$$

Where n_p is the propeller angular rate, given in revolutions per second.

The rudder is modeled as:

$$\boldsymbol{\tau}_{rudder} = \boldsymbol{B}_{rudder}\boldsymbol{\delta}u_r^2 \tag{2.8}$$

Where $\boldsymbol{B}_{rudder}$ and δ is given as:

$$\boldsymbol{B}_{rudder} = \begin{bmatrix}0 & 0 & 0 & 0 \\ Y_{\delta u^2} & Y_{\delta u^2} & 0 & 0 \\ 0 & 0 & Z_{\delta u^2} & Z_{\delta u^2} \\ Y_{\delta u^2}l_z & -Y_{\delta u^2}l_z & -Z_{\delta u^2}l_y & Z_{\delta u^2}l_y \\ 0 & 0 & Z_{\delta u^2}l_x & Z_{\delta u^2}l_x \\ -Y_{\delta u^2}l_x & -Y_{\delta u^2}l_x & 0 & 0\end{bmatrix} \tag{2.9a}$$

$$\boldsymbol{\delta} = \begin{bmatrix}\delta_{top} & \delta_{bottom} & \delta_{port} & \delta_{starboard}\end{bmatrix}^T \tag{2.9b}$$

Where $l_x, l_y, l_z > 0$ are the distances from the center of origin (CO) to the rudders along the x, y and z axes respectively. $\delta_a, a \in \{top, bottom, port, starboard\}$ is the rudder deflection angles, given in radians. Some drag in surge will be present when the rudders are used, but this is not included in the model provided by FFI.

The actuators are limited by saturation limits. In addition, the rudder rate of change is limited. Therefore, n_p and $\boldsymbol{\delta}$ is required to satisfy the constraints:

$$n_p \in [n_{p_{min}}, n_{p_{max}}] \tag{2.10a}$$

$$\|\boldsymbol{\delta}\|_\infty \leq \delta_{max} \tag{2.10b}$$

$$\left\|\dot{\boldsymbol{\delta}}\right\|_\infty \leq \dot{\delta}_{max} \tag{2.10c}$$

Where n_{min} and n_{max} are minimum and maximum propeller speeds, δ_{max} is the maximum rudder deflection angle and $\dot{\delta}_{max}$ is the maximum rudder rate of change.

2.2 Collision avoidance theory

Collision avoidance systems can be divided into three major categories based on their architecture [Tan et al., 2004a]:

1. Deliberative architecture
2. Reactive architecture
3. Hybrid architecture

The *deliberative* architecture is also known as a sense-plan-act approach or a *global* method. Global methods are often referred to as *path planning* or *motion planning* methods [Loe, 2008]. The architecture maintains an internal representation of the environment at all times, and stores the information gathered by the sensors for later use. This is often done by using a priori information, and continuously building a map of the environment based on the current sensor measurement. By maintaining the internal environment representation, reasoned decisions can be made based on sensor data gathered over a long period of time. The deliberative approaches often guarantee that a path to a goal is found if it exists. The downside is that they are computationally expensive, which can result in unresponsive behavior when presented with unpredicted situations.

The *reactive* architecture is also known as a sense-act or a *local* method. The sensor measurements are used directly in the control of the vehicle behavior, without building a map of the environment. This may lead to non-optimal paths [Borenstein and Koren, 1991], and even trap the vehicle when presented with *local minima*, for example dead ends. On the other hand, the responsiveness of the system is excellent due to the low computational cost. This makes the reactive architecture well suited for real-time applications where fast response is required.

The *hybrid* architecture tries to combine the deliberative and reactive architectures into a superior architecture. It applies a reactive layer responding to rapid unforeseen changes in the environment, and a deliberative layer which simultaneously apply high level planning with a global path planning algorithm. The reactive layer contains the time critical safety functions, while the deliberative layer ensures that the vehicle is guided along a path leading to the goal.

The Dynamic window algorithm is a local method. It is therefore expected to sometimes choose non-optimal paths, which may get the AUV stuck in local minima. A practical implementation should be of the hybrid architecture, using the Dynamic window algorithm as the reactive layer. However, the main objective of this thesis is to investigate the Dynamic window algorithm in depth, thus a reactive architecture is chosen.

Local collision avoidance methods may in general be divided into directional and velocity space based methods. The directional methods generate a direction which the AUV is required to travel, while velocity space based approaches searches in the velocity space of the AUV and can hence take kinematic and dynamic constraints directly into account [Seder et al., 2005]. The Dynamic window algorithm [Fox et al., 1997], velocity obstacle [Castro et al., 2002] and curvature-velocity [Simmons, 1996] approaches are examples of velocity space based methods. Examples of directional methods are the potential field [Khatib, 1985], the vector field histogram (VFH) [Borenstein and Koren, 1991] method and the Edge following algorithm [Tan et al., 2004b].

2.3 The Dynamic window algorithm

The Dynamic window algorithm (DWA) was introduced in [Fox et al., 1997]. The method is popular as a reactive collision avoidance algorithm, and a number of modifications have been suggested.

2.3.1 Original method

The Dynamic window approach to collision avoidance prohibits infeasible commands to the vehicle, by taking the acceleration limits into consideration. The algorithm was first designed for a car-like mobile robot with first order nonholonomic constraints, moving in 3DOF [Fox et al., 1997]. This application have similarities to the horizontal AUV collision avoidance problem. The essence of the algorithm is to neglect sway motion, and predict the robot trajectory as a circular arc with constant surge speed and yaw rate. This results in a 2 dimensional search space. The optimal trajectory (and the corresponding desired surge speed and yaw rate) is found from this search space by maximizing an objective function.

It should be noted that the original method does not include ocean current, hence it is assumed that $\boldsymbol{V}_c = \mathbf{0}$ in this section. This results in $\boldsymbol{\nu} = \boldsymbol{\nu}_r$. Therefore, the rest of this section uses the absolute velocity $\boldsymbol{\nu}$.

Kinematic representation

The method assumes that the sway speed and the pitch and roll angles are identically zero. The north and east positions can then be written as:

$$\begin{aligned} N(t_n) &= N(t_0) + \int_{t_0}^{t_n} u(t)\cos(\psi(t))\mathrm{d}t \\ E(t_n) &= E(t_0) + \int_{t_0}^{t_n} u(t)\sin(\psi(t))\mathrm{d}t \end{aligned} \tag{2.11}$$

The surge speed and yaw angle can be expressed as:

$$\begin{aligned} u(t) &= u(t_0) + \int_{t_0}^{t} \dot{u}(\tilde{t})\mathrm{d}\tilde{t} \\ \psi(t) &= \psi(t_0) + \int_{t_0}^{t} r(\tilde{t})\mathrm{d}\tilde{t} \\ &= \psi(t_0) + \int_{t_0}^{t} \left(r(t_0) + \int_{t_0}^{\tilde{t}} \dot{r}(\hat{t})\mathrm{d}\hat{t} \right)\mathrm{d}\tilde{t} \end{aligned} \tag{2.12}$$

The control input to the AUV will affect $\dot{u}(t)$ and $\dot{r}(t)$. Inserting (2.12) into (2.11), the AUV position can be expressed in terms of the initial dynamic configuration, and the vehicle accelerations. By neglecting the actuator dynamics, the accelerations can be considered as directly controllable. The north position can be expressed as:

$$\begin{aligned} N(t_n) = N(t_0) + \int_{t_0}^{t_n} \left(u(t_0) + \int_{t_0}^{t} \dot{u}(\tilde{t})\mathrm{d}\tilde{t} \right) \cos\Bigg(\psi(t_0) \\ + \int_{t_0}^{t} \left(r(t_0) + \int_{t_0}^{\tilde{t}} \dot{r}(\hat{t})\mathrm{d}\hat{t} \right)\mathrm{d}\tilde{t} \Bigg)\mathrm{d}t \end{aligned} \tag{2.13}$$

Practical control systems have a specified sample frequency, hence the propeller speed and rudder deflection can only be changed at specific times. By assuming that the acceleration is constant between each sample point, (2.13) is rewritten to (2.14), in discrete time.

$$\begin{aligned} N(t_n) = N(t_0) + \sum_{i=0}^{n-1} \int_{t_i}^{t_{i+1}} \Big(u(t_i) + \dot{u}_i \Delta_i(t) \Big) \cos\Big(\psi(t_i) \\ + r(t_i)\Delta_i(t) + \frac{1}{2}\dot{r}_i \Delta_i^2(t) \Big)\mathrm{d}t \end{aligned} \tag{2.14}$$

Where $\Delta_i(t) = t - t_i$ is a shifted time variable, $\dot{u}_i$ and $\dot{r}_i$ are the constant surge and yaw rate accelerations in the time interval $t = [t_i, t_{i+1})$.

If the time steps $[t_i, t_{i+1})$ are sufficiently small, the equation can be simplified further by approximating the velocities by a constant value in each time step. The term $u(t_i) + \dot{u}_i\Delta_i$ may be approximated by a constant $u_i \in [u(t_i), u(t_{i+1})]$, while $\psi(t_i) + r(t_i)\Delta_i(t) + \frac{1}{2}\dot{r}_i\Delta_i^2(t)$ may be approximated by $\psi(t_i) + r_i\Delta_i(t)$, where $r_i \in [r(t_i), r(t_{i+1})]$. This results in (2.15).

$$N(t_n) = N(t_0) + \sum_{i=0}^{n-1} \int_{t_i}^{t_{i+1}} u_i \cos\Big(\psi(t_i) + r_i(t - t_i)\Big)\mathrm{d}t \tag{2.15}$$

By solving the integral, one finally ends up with:

$$\begin{aligned} N(t_n) &= N(t_0) + \sum_{i=0}^{n-1} F_N^i(t_{i+1}) \\ F_N^i(t) &= \begin{cases} \frac{u_i}{r_i}\Big(\sin(\psi(t_i)) - \sin\big(\psi(t_i) + r_i(t - t_i)\big)\Big) & , r_i \neq 0 \\ u_i \cos(\psi(t_i))t & , r_i = 0 \end{cases} \end{aligned} \tag{2.16}$$

The equation for the east position can be derived in the same fashion. The expression for the east position is:

$$\begin{aligned} E(t_n) &= E(t_0) + \sum_{i=0}^{n-1} F_E^i(t_{i+1}) \\ F_E^i(t) &= \begin{cases} -\frac{u_i}{r_i}\Big(\cos(\psi(t_i)) - \cos\big(\psi(t_i) + r_i(t - t_i)\big)\Big) & , r_i \neq 0 \\ u_i \sin(\psi(t_i))t & , r_i = 0 \end{cases} \end{aligned} \tag{2.17}$$

If the i-th yaw rate is zero, $r_i = 0$, the i-th segment will be a straight line. Otherwise, it will be a circular arc-segment with radius $M_r^i = \frac{u_i}{r_i}$. It is worth noting that the acceleration will be discontinuous when r_i changes.

Search space

The search space for the algorithm is defined by modeling the AUV trajectory as circular arcs, using (2.16) and (2.17). Each trajectory is then defined by n velocity pairs, (u_i, r_i). Hence, one has to specify n velocity pairs to define a trajectory. To limit the search space, only the first velocity pair in a trajectory is specified, while the remaining $n - 1$ pairs are considered constant. This defines the AUV trajectories by a single velocity pair, creating a 2 dimensional search space. Only positive velocities are considered.

To reduce the search space further, non-admissible velocities are removed. For a velocity pair to be considered admissible, the AUV must be able to stop before it reaches the closest obstacle on the corresponding trajectory. The space of admissible velocities are defined in (2.18). The accelerations of the AUV is limited to the intervals $\dot{u} \in [\dot{u}_{min}, \dot{u}_{max}]$ and $\dot{r} \in [-\dot{r}_{max}, \dot{r}_{max}]$.

$$V_a = \Big\{(u,r) \in \mathbb{R} \times \mathbb{R} \Big| u \leq \sqrt{-2 \cdot \text{dist}(u,r) \cdot \dot{u}_{min}} \\ \wedge r \leq \sqrt{2 \cdot \text{dist}(u,r) \cdot \dot{r}_{max}} \Big\} \quad (2.18)$$

Where $\text{dist}(u, r)$ expresses the distance to the closest obstacle on the corresponding trajectory.

Further, the *dynamic velocity window* is introduced. This defines a set of velocities, centered around the current velocity (u_a, r_a), which are reachable during the next time step. Using T as the time allowed for acceleration during the next time step, the dynamic velocity window can be defined as:

$$V_d = \Big\{(u,r) \in \mathbb{R} \times \mathbb{R} \Big| u \in [u_a + \dot{u}_{min}T, u_a + \dot{u}_{max}T] \\ \wedge\, r \in [r_a - \dot{r}_{max}T, r_a + \dot{r}_{max}T] \Big\} \quad (2.19)$$

Lastly, the set of possible velocities are defined:

$$V_s = \Big\{(u,r) \in \mathbb{R} \times \mathbb{R} \Big| u \in [0, u_{max}] \wedge r \in [-r_{max}, r_{max}] \Big\} \quad (2.20)$$

Where u_{max} and r_{max} are the maximum surge speed and yaw rate of the AUV respectively.

The resulting search space, denoted V_r, can be defined as:

$$V_r = V_s \cap V_a \cap V_d \quad (2.21)$$

Velocity selection

By defining an objective function, an optimal velocity pair may be defined and selected. The objective function, given in (2.22), consists of three parts. The optimal velocity pair is found by maximizing the function.

$$G(u,r) = \sigma\left(\alpha \cdot \text{heading}(u,r) + \beta \cdot \text{dist}(u,r) + \gamma \cdot \text{velocity}(u,r)\right) \quad (2.22)$$

Where *heading*(u, r), *dist*(u, r) and *velocity*(u, r) are functions normalized to $[0, 1]$. $\alpha, \beta, \gamma > 0$ are scaling factors and σ is a low pass filter, for smoothing the function.

The *heading*(u, r) function assigns a value to keeping the desired heading. For a guidance scheme, the function would simply compare the alignment of the AUV with the desired heading from the guidance system. Since the algorithm specifies a yaw rate, the resulting heading for a velocity pair has to be computed. The heading is estimated as the heading the AUV would have, after stopping the yaw rotation by exerting maximum yaw deceleration after the first time step.

The *dist*(u, r) function assigns a value to keeping distance to obstacles. The function calculates the distance the AUV can travel along the predicted trajectories, without colliding with an obstacle. The function is the same that is used in (2.18), but scaled to $[0, 1]$.

The *velocity*(u, r) function assigns a value to using a high velocity, and is simply a linear function in surge speed, u. This favors progress of the AUV, and prohibits the AUV from stopping when it is avoidable.

Selection of the optimal velocity pair can be summarized as:

$$\begin{aligned} \max_{(u,r)} \; & G(u, r) = \sigma \left(\alpha \cdot \text{heading}(u, r) + \beta \cdot \text{dist}(u, r) + \gamma \cdot \text{velocity}(u, r)\right) \\ & \text{s.t.} \;\; (u, r) \in V_r \end{aligned} \tag{2.23}$$

The selected velocity pair is only implemented in one time step, before the algorithm is rerun. Hence, only a small part of the selected trajectory will be followed, before a new trajectory is planned. The maximization problem is solved numerically by uniformly sampling the search space V_r, and comparing the objective function value of all the velocities in the sampled search space.

Summary of the method

The Dynamic window algorithm is a local collision avoidance method, that considers the vehicle dynamics. This makes the approach more computationally expensive than other local collision avoidance methods as the Edge following algorithm [Tan et al., 2004b] and Potential field methods [Khatib, 1985, Koren and Borenstein, 1991, Borenstein and Koren, 1991], but it may also perform better. The results in [Eriksen, 2014] favor the Dynamic window algorithm over the other named approached.

Assuming zero sway speed is not valid for an AUV, as it will sideslip. The assumption may be somewhat more valid by transforming the AUV model to the *pivot point* [Caharija et al., 2012, Caharija, 2014, Fossen, 2011], which removes the rudder actuation in sway. However, the vehicle will still sideslip due to the AUV dynamics, so this should be included in the derivation.

2.3.2 Proposed extensions

The original Dynamic window algorithm [Fox et al., 1997] for collision avoidance has been investigated by quite some researchers. Several modifications to the original method has been suggested.

A problem with the algorithm is the lack of global information. This makes the robot sensitive to local minima, since it may get stuck. To resolve this issue, several modifications has been done to include global information in the algorithm. [Brock and Khatib, 1999] introduced the Global dynamic window approach (GDWA). It combines planning and real-time obstacle avoidance to avoid the robot to be trapped in local minima by including a Neuro-Fibromatosis type-1 (NF1) navigation function in the objective function. The navigation function contains information about the free space connectivity of the environment, and by following the gradient of the function a path to the goal is found. The NF1 function can be constructed by computing the shortest path problem on a graph defined on a grid based map of the environment [Ögren and Leonard, 2002]. Naturally, this requires global knowledge of the environment. They also extended the method to apply to holonomic robots by using a circular search space of translational velocities. [Tusseyeva et al., 2013] applied the GDWA on an AUV, but did not consider the second order non-holonomic constraints of the vehicle. [Seder et al., 2005] further modified the algorithm by integrating the Dynamic window algorithm with a focused D* (FD*) global planner.

Velocity space based approaches typically assumes that the robot travels along circular arcs [Seder et al., 2005]. The Dynamic window algorithm was originally designed for robots with first order non-holonomic constraints, more specifically a car-like robot without any sideways movement. This makes it accurate to predict the trajectories using circle segments. AUVs have second order non-holonomic constraints, hence AUVs will be subject to sideways movement [Oriolo and Nakamura, 1991]. This makes predicting the trajectories as circular arcs inaccurate. [Tusseyeva et al., 2013] argues that using circles is accurate when only allowing small vehicle accelerations. This highly depends on the dynamics of the vehicle, as any lateral speed will affect the trajectory regardless of the accelerations involved. Using clothoid curves instead of circular arcs was suggested by [Schröter et al., 2007]. Clothoids are more suitable for modeling robot trajectories as they have a continuous curvature. This results in continuous accelerations when traveling along the curve [Fleury et al., 1995]. [Loe, 2008] applied the Dynamic window algorithm to a surface vehicle. He estimated the lateral speed by computing the steady state solution to the sway dynamics in each time step, and then simulating a simplified set of equations of motion to predict the trajectories. He also included time varying acceleration limits to account for the dynamics changing as a function of the vehicle velocity. Further, [Kiss and Tevesz, 2012] uses a linear model of the vehicle dynamics to generate the trajectories. This produces accurate predictions, but requires a linear model of the vehicle.

Several modifications has been done to the objective function. To integrate the DWA with a global planner, the *heading* term is often exchanged for a measure of alignment, which measures the alignment between a trajectory and a desired path [Brock and Khatib, 1999]. [Seder et al., 2005, Tusseyeva et al., 2013] considered time parameterized paths and included time in the alignment measure, so the *velocity* term in the objective function could be omitted. This resulted in an objective function with only two terms, which was easily tuned. They also considered the time until collision instead of the distance until collision. When considering distance until collision, a singularity occurs when the translational velocity is zero since a pure rotation produces no robot movement in 2D. This singularity is not present when considering time until collision, which also is more intuitive as it normalizes for the trajectory velocity. [Berti et al., 2008] used a Lyapunov motivated control law for the desired velocity to focus the objective function. They were able to guarantee global asymptotic convergence to a goal point, with the assumption that no obstacles were present. The "Shared Control Dynamic Window Approach" was proposed by [Inñigo Blasco et al., 2014], which is designed for semi-autonomous non-holonomic robots. It focuses the *heading* and *velocity* terms to a user specified input, and acts like a navigation assistance system. When the environment gets cluttered, a "dangerousness metric" makes the algorithm take more control and ensure safe operation.

The Dynamic window algorithm have a tendency to select robot trajectories close to the obstacle boundaries. To increase the obstacle clearance [Brock and Khatib, 1999] modified the *heading* term when the robot selected trajectories with too little obstacle clearance. [Seder and Petrović, 2007] proposed using a velocity dependent safety contour around the robot. Hence, a large translational velocity prohibits the robot to travel close to obstacles, while a small translational velocity allows the robot to travel closer to obstacles. [Eriksen, 2014] applied a 2D filter to the objective function to increase the obstacle clearance. This increased the minimum obstacle clearance, but the dynamic response of the robot was reduced.

A number of other modifications has also been proposed. [Ögren and Leonard, 2002, Kiss and Tevesz, 2012] developed an alternative formulation of the DWA using Model predictive control (MPC), and [Ögren and Leonard, 2002] was able to guarantee convergence to a goal using Lyapunov analysis. The "Trajectory Particle Filter", suggested by [Schröter et al., 2007], draw analogs between the Dynamic window algorithm and particle filters, and uses a normal distribution to sample the search space. [Castro et al., 2002] included a velocity cone, inspired by velocity obstacles, in the DWA to account for moving obstacles. [Seder and Petrović, 2007] modeled moving obstacles as moving cells in a grid map. This makes arbitrary obstacle configurations possible, and offers great flexibility. The downside is that each cell in the grid map must be considered as a separate object, requiring a large amount of calculations to predict the movement of each cell instead of the movement of one obstacle.

Chapter 3

Modifications to the Dynamic window algorithm

The original Dynamic window algorithm is modified in a number of ways to improve the performance when applied to the HUGIN1000 AUV:

- Create a local obstacle grid map, to add safety sectors around the detected obstacles and hence control the obstacle clearance.
- Include time varying acceleration limits in the search space.
- Partially linearize the AUV dynamics and predict the AUV trajectories using a linear approximation of the dynamics instead of using circular arcs.

The modifications presented in this chapter is derived by the author, unless otherwise it noted.

3.1 Control plant model

A control plant model is found by simplifying the 6DOF model in section 2.1. With a slight abuse of notation, $\boldsymbol{\eta}$, $\boldsymbol{\nu}_r$, $\boldsymbol{R}(\boldsymbol{\eta})$, $\boldsymbol{M}$, $\boldsymbol{C}(\boldsymbol{\nu}_r)$ and $\boldsymbol{D}(\boldsymbol{\nu}_r)$ are redefined in this chapter to model the AUV in 3DOF.

Assumption 3.1. *Heave speed, and the roll and pitch angles are assumed constant and equal to zero.*

Remark 3.1. *Assuming zero roll angle is a common assumption for slender body vehicles such as AUVs [Fossen, 2011]. Their CG is normally located below the CB, at the same (x,y) coordinate in $\{b\}$, which passively stabilizes the roll motion. Further, by equally utilizing the top and bottom rudders, and the port* *and starboard rudders, no roll moment is generated by the rudders. For an AUV equipped with a depth controller, close to zero heave speed and pitch angle is achieved when the AUV is traveling in a horizontal plane, without vertical ocean current (as of Assumption 2.2).*

Assumption 3.2. *Actuator dynamics and saturation limits are neglected.*

Remark 3.2. *The search space of the DWA is constructed to account for actuator saturation, and the actuator dynamics are considered fast enough to be neglected.*

Assumption 3.1 allow the control plant model to be formulated in 3 DOF (surge, sway and yaw). The 3 DOF model is given as (see section 2.1 for details):

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\boldsymbol{\eta})\boldsymbol{\nu}_r + \boldsymbol{V}_c \tag{3.1a}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}}_r + \boldsymbol{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r = \boldsymbol{B}\boldsymbol{f} \tag{3.1b}$$

Where $\boldsymbol{\eta} = \begin{bmatrix} N & E & \psi \end{bmatrix}^T$, $\boldsymbol{V}_c = \begin{bmatrix} V_N & V_E \end{bmatrix}^T$, $\boldsymbol{\nu}_r = \begin{bmatrix} u_r & v_r & r \end{bmatrix}^T$, $\boldsymbol{f} = \begin{bmatrix} X & N \end{bmatrix}^T$ and:

$$\boldsymbol{R}(\boldsymbol{\eta}) = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2a}$$

$$\boldsymbol{M} = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{23} & m_{33} \end{bmatrix} \tag{3.2b}$$

$$\boldsymbol{C}(\boldsymbol{\nu}_r) = \begin{bmatrix} 0 & 0 & -m_{22}v_r - m_{23}r \\ 0 & 0 & m_{11}u_r \\ m_{22}v_r + m_{23}r & -m_{11}u_r & 0 \end{bmatrix} \tag{3.2c}$$

$$\boldsymbol{D}(\boldsymbol{\nu}_r) = -\begin{bmatrix} X_u + X_{uu}|u_r| & 0 & 0 \\ 0 & Y_v & Y_r \\ 0 & N_v & N_r \end{bmatrix} \frac{|u_r'|}{u_0} \tag{3.2d}$$

$$\boldsymbol{B} = \begin{bmatrix} b_{11} & 0 \\ 0 & b_{22} \\ 0 & b_{32} \end{bmatrix} \tag{3.2e}$$

Note that to simplify the modeling, $\boldsymbol{f}$ is selected as the control input. For positive surge speeds, and based on assumption 3.2, it is always possible to calculate a proper propeller speed and rudder deflection given $\boldsymbol{f}$. The constants in the $\boldsymbol{B}$ matrix is given as $b_{11} = b_{32} = 1$, while b_{22} captures the coupling from the yaw torque to the sway force in the actuator model, given from $Y = -\frac{1}{l_x}N \triangleq b_{22}N$.

The control input $\boldsymbol{f}$ affects the sway dynamics through the coupling term b_{22}. This complicates the control design. To remove this coupling, the model is translated to the pivot point [Fossen, 2011, Fredriksen and Pettersen, 2006, Caharija, 2014]. The transformation is given as a coordinate transformation:

$$u_r = u_r, \quad \bar{v}_r = v_r + \epsilon r, \quad r = r \tag{3.3}$$

Where ϵ is given as:

$$\epsilon \triangleq -\frac{m_{33}b_{22} - m_{23}b_{32}}{m_{22}b_{32} - m_{23}b_{22}} \tag{3.4}$$

ϵ is well defined as long as the system is controllable in yaw [Caharija, 2014], which it naturally is since $b_{32} = 1$.

The new relative velocity vector $\bar{\nu}_r$ is given as:

$$\bar{\boldsymbol{\nu}}_r = \begin{bmatrix} u_r \\ \bar{v}_r \\ r \end{bmatrix} \triangleq \boldsymbol{H}^{-1}\boldsymbol{\nu}_r \tag{3.5}$$

Where the transformation matrix is given as:

$$\boldsymbol{H} \triangleq \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -\epsilon \\ 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

The transformation corresponds to a physical translation of the model along the $\{b\}$ x-axis, of a distance ϵ [Caharija, 2014]. Without loss of generality, the model is therefore transformed to describe the motion of a point located at $\boldsymbol{p}^b = \begin{bmatrix} \epsilon & 0 & 0 \end{bmatrix}^T$. The point $\boldsymbol{p}$ is given in $\{n\}$ as [Fredriksen and Pettersen, 2006]:

$$\bar{N} = N + \epsilon\cos(\psi), \quad \bar{E} = E + \epsilon\sin(\psi) \tag{3.7}$$

The new position vector $\bar{\boldsymbol{\eta}}$ is given as:

$$\bar{\boldsymbol{\eta}} = \begin{bmatrix} \bar{N} \\ \bar{E} \\ \psi \end{bmatrix} \tag{3.8}$$

Notice that no down-position is given, as the model is in 3DOF.

The transformed system is given as:

$$\dot{\bar{\boldsymbol{\eta}}} = \boldsymbol{R}(\bar{\boldsymbol{\eta}})\bar{\boldsymbol{\nu}}_r + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \tag{3.9a}$$

$$\bar{\boldsymbol{M}}\dot{\bar{\boldsymbol{\nu}}}_r + \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r + \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r = \bar{\boldsymbol{B}}\boldsymbol{f} \tag{3.9b}$$

Where $\bar{\boldsymbol{M}} = \boldsymbol{H}^T\boldsymbol{M}\boldsymbol{H}$, $\bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r) = \boldsymbol{H}^T\boldsymbol{D}(\bar{\boldsymbol{\nu}}_r)\boldsymbol{H}$ and $\bar{\boldsymbol{B}} = \boldsymbol{H}^T\boldsymbol{B}$. The Coriolis-centripetal matrix is not transformed, but rather parameterized using $\bar{\boldsymbol{M}}$. The matrices are given such that the following property holds [Caharija et al., 2012]:

$$\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} = \begin{bmatrix} \frac{b_{11}}{m_{11}}X \\ 0 \\ \frac{m_{22}b_{32} - m_{23}b_{22}}{m_{22}m_{33} - m_{23}^2}N \end{bmatrix} \tag{3.10}$$

Finally, it is often useful to write the AUV accelerations in (3.9b) in component form. To simplify the notation, a transformation is defined:

$$\begin{bmatrix}\tau_u\\ \tau_r\end{bmatrix} \triangleq \begin{bmatrix}\frac{b_{11}}{m_{11}} & 0\\ 0 & \frac{m_{22}b_{32}-m_{23}b_{22}}{m_{22}m_{33}-m_{23}^2}\end{bmatrix}\begin{bmatrix}X\\ N\end{bmatrix} \tag{3.11}$$

This transformation is well-defined. By applying (3.11), the AUV dynamics (3.9b) can be written as:

$$\begin{aligned}\dot{\boldsymbol{\nu}}_r &= \bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} - \bar{\boldsymbol{M}}^{-1}\left(\bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r + \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r\right)\\ &= \begin{bmatrix}\tau_u\\ 0\\ \tau_r\end{bmatrix} - \bar{\boldsymbol{M}}^{-1}\left(\bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r + \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r\right)\end{aligned} \tag{3.12}$$

3.2 Environment modeling

[Eriksen, 2014] applied a 2D FIR filter to the objective function in order to make the DWA select trajectories with an applicable side clearance to obstacles. This made the AUV circumnavigate obstacles with some side clearance, but the dynamic response suffered from the filtering. The magnitude of the side clearance was also difficult to control. Inspired by [Rodriguez-Seda et al., 2014], a different approach to achieve side clearance is suggested. By preprocessing the sonar measurements, and creating a local map of the environment, safety regions can be added to the obstacles. This is a more analytical way of achieving a desired side clearance.

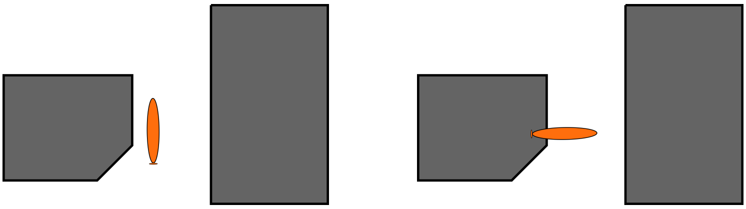
In order to model the environment, the *configuration space* $\mathcal{C}$ must be defined. The configuration space is a complete specification of the AUVs location and orientation. For an AUV traveling in the horizontal plane, as described by (3.9a) and (3.9b), the configuration space consists of the AUV position and orientation in the horizontal plane. It can be expressed as $\mathcal{C} = \mathbb{R}^2 \times SO(2)$ [Spong et al., 2006], where $\mathbb{R}^2$ specifies the position of the AUV and $SO(2)$ specifies the orientation of the AUV using a rotation matrix in the *special orthogonal group of order 2* [Fossen, 2011]. This results in a configuration space of three dimensions, which relate to the 3DOF model of the AUV.

Further, the *workspace* $\mathcal{W}$ specify the total area swept out by the AUV through all the possible AUV configurations. Since no constraints is put on the movement of the AUV, the workspace is given as $\mathcal{W} = \mathbb{R}^2$. Let the set of obstacles in $\mathcal{W}$ be denoted as $\mathcal{O}$, and the subset of $\mathcal{W}$ occupied by the AUV given the configuration $\boldsymbol{\eta}$ as $\mathcal{A}(\boldsymbol{\eta})$. The set of configurations which result in a collision is given as:

$$\mathcal{C}_{obs} = \{\boldsymbol{\eta} \in \mathcal{C} | \mathcal{A}(\boldsymbol{\eta}) \cap \mathcal{O} \neq \emptyset\} \tag{3.13}$$

The set of collision-free configurations is then:

$$\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs} \tag{3.14}$$



(a) No collision

(b) Collision

Figure 3.1: AUV heading is part of $\mathcal{C}_{free}$ and $\mathcal{C}_{obs}$

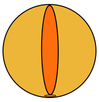


Figure 3.2: Circular approximation of the AUV footprint

Notice that $\mathcal{C}_{free}$ and $\mathcal{C}_{obs}$ is given in the space $\mathbb{R}^2 \times SO(2)$. This is illustrated in figure 3.1, where the heading in figure 3.1a does not cause a collision while the one in figure 3.1b causes one. To simplify the representation of $\mathcal{C}_{free}$ and $\mathcal{C}_{obs}$, the configuration space is collapsed by removing the AUV heading. This is done by approximating the AUV footprint as a circle, shown in figure 3.2, which makes the heading irrelevant for computing $\mathcal{C}_{free}$ and $\mathcal{C}_{obs}$ and hence makes it possible to collapse $\mathcal{C}_{free}$ and $\mathcal{C}_{obs}$ into $\mathbb{R}^2$.

In practice, this is done by expanding the obstacles with the maximum radius of the AUV and representing the AUV as a point $\boldsymbol{p} = \begin{bmatrix}\bar{N} & \bar{E}\end{bmatrix}^T$ in a reduced configuration space $\bar{\mathcal{C}} = \mathbb{R}^2$. Let the set of expanded obstacles in $\mathcal{W}$ be given as $\bar{\mathcal{O}}$ (shown in figure 3.3), and $\bar{\mathcal{A}}(\boldsymbol{p})$ represent a point in $\mathcal{W}$ occupied by the AUV. The reduced set of configurations which result in collision is then:

$$\bar{\mathcal{C}}_{obs} = \{\boldsymbol{p} \in \bar{\mathcal{C}} | \bar{\mathcal{A}}(\boldsymbol{p}) \cap \bar{\mathcal{O}} \neq \emptyset\} \tag{3.15}$$

Further, the set of collision-free configurations is then given as:

$$\bar{\mathcal{C}}_{free} = \bar{\mathcal{C}} \setminus \bar{\mathcal{C}_{obs}} \tag{3.16}$$

$\bar{\mathcal{C}}_{obs}$ and $\bar{\mathcal{C}}_{free}$ are now part of $\mathbb{R}^2$, which makes it sufficient to use the AUV position to check for a collision.

Two sets which enclose the obstacles are defined; the *antitarget* region, $\mathcal{T}$ (which corresponds to $\bar{\mathcal{O}}$), and the *avoid* region, Ω. The sets are defined as:

$$\begin{aligned}\mathcal{T} &= \left\{\boldsymbol{p} \in \bar{\mathcal{C}} \middle| \, \|\boldsymbol{p} - \boldsymbol{p}_{obs}\|_2 \leq r^* \right\} \\ \Omega &= \left\{\boldsymbol{p} \in \bar{\mathcal{C}} \middle| \, \|\boldsymbol{p} - \boldsymbol{p}_{obs}\|_2 \leq \bar{r} \right\}\end{aligned} \tag{3.17}$$

Where $\boldsymbol{p}_{obs} \in \mathbb{R}^2$ is the position of obstacles and $\bar{r} > r^* > 0$ are scalars defining the size of the regions. In particular, r^* is the largest radius of the AUV corre-

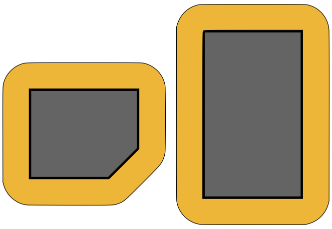


Figure 3.3: Expanded obstacles to collapse the configuration space

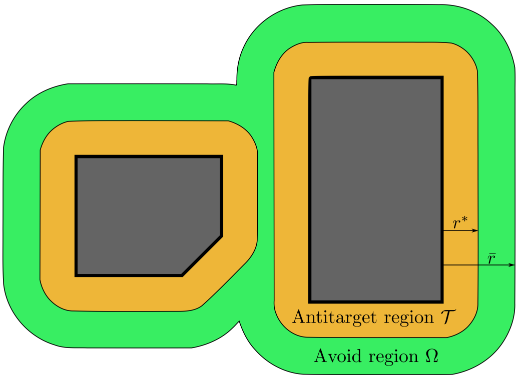


Figure 3.4: Antitarget and avoid region

sponding to approximating the AUV footprint as a circle. The antitarget region is interpreted as the region where a collision will occur, while the avoidance region is interpreted as a safety region that is not desirable to enter. The regions are illustrated in figure 3.4. From the definition of the sets, it is clear that $\mathcal{T} \subset \Omega$.

The regions can be implemented using a local 2D occupancy grid map [Elfes, 1987, Borenstein and Koren, 1991], where each cell in the map is given values in the domain $\{0, 1, 2\}$. The cell values are interpreted as:

$$c_{i,j} = \begin{cases} 0 & : \boldsymbol{p}_{i,j} \notin \Omega & \text{Cell is free to visit} \\ 1 & : \boldsymbol{p}_{i,j} \in \Omega \cap \setminus \mathcal{T} & \text{Cell is undesirable to visit} \\ 2 & : \boldsymbol{p}_{i,j} \in \mathcal{T} & \text{Cell is not visitable} \end{cases} \tag{3.18}$$

3.3 Search space

The search space is intended to take the dynamic constraints of the vehicle into account, by only containing feasible velocity commands. Most implementations consider the acceleration limits to be constant, and independent from the vehicle velocity. Hence, it is usually assumed that the robots ability to accelerate and decelerate in surge and yaw is independent of the surge speed and yaw rate. This is not the case for AUVs, as they utilize rudders for controlling yaw. As seen in section 2.1.3, the yaw torque is quadratic in the surge speed. To fulfill assumption 3.2 the search space is required to only contain feasible velocities, which does not saturate the actuators. Hence, the possible accelerations must be dependent on the current velocity.

Little effort has been done to address this problem. [Loe, 2008] modeled the vehicle acceleration limits as $\dot{u}_{max} = \dot{u}_{max}(u)$, $\dot{u}_{min} = \dot{u}_{min}(u)$ and $\dot{r}_{min} = \dot{r}_{max} = \dot{r}_{max}(r)$. This is an improvement to considering the acceleration limits as constants, but does not consider the coupling from surge speed to yaw torque. To address this, a new approach is proposed.

As in the original method, the search space consists of three parts, namely the Dynamic velocity window V_d, the Possible velocities V_s and the Admissible velocities V_a. Each of them are described below.

Dynamic velocity window

By including the actuator dynamics into the dynamics (3.9b), the actuator constraints can be included in the search space derivation. The actuator model can be included in (3.9b) by expanding the control input as:

$$\bar{\boldsymbol{B}}\boldsymbol{f} = \boldsymbol{H}^T \boldsymbol{\tau}(\boldsymbol{\nu}_r, \boldsymbol{\delta}, n_p) \tag{3.19}$$

Where $\boldsymbol{\tau}(\boldsymbol{\nu}_r, \boldsymbol{\delta}, n_p)$ is a vector of generalized forces applied to the CO in 3DOF, given the AUV velocity, rudder deflections and propeller speed. Note that it is transformed to the pivot point using the transformation matrix $\boldsymbol{H}$, given in (3.6). As seen in section 2.1.3, only the top and bottom rudders affect the 3DOF dynamics. Further, remark 3.1 suggests that the top and bottom rudders should be equally utilized. Therefore, a new rudder command signal is defined as $\delta_{yaw} = \delta_{top} + \delta_{bottom}$, where $\delta_{top} = \delta_{bottom}$ is required. (3.19) can then be written as:

$$\begin{aligned} \bar{\boldsymbol{B}}\boldsymbol{f} &= \boldsymbol{H}^T \left(\boldsymbol{\tau}_{propeller}(\bar{\boldsymbol{\nu}}_r, n_p) + \boldsymbol{\tau}_{rudder}(\bar{\boldsymbol{\nu}}_r, \delta_{yaw})\right) \\ &\triangleq \bar{\boldsymbol{\tau}}(\bar{\boldsymbol{\nu}}_r, \delta_{yaw}, n_p) \end{aligned} \tag{3.20}$$

Where $\boldsymbol{\tau}_{propeller}(\bar{\boldsymbol{\nu}}_r, n_p)$ and $\boldsymbol{\tau}_{rudder}(\bar{\boldsymbol{\nu}}_r, \delta_{yaw})$ are vectors containing forces in 3DOF generated by the propeller and rudders respectively, at the CO. They can be expressed as (see section 2.1.3):

$$\boldsymbol{\tau}_{propeller}(\bar{\boldsymbol{\nu}}_r, n_p) = \begin{bmatrix} T_{nn}|n_p|n_p + T_{un}u_r n_p \\ 0 \\ 0 \end{bmatrix} \tag{3.21a}$$

$$\boldsymbol{\tau}_{rudder}(\bar{\boldsymbol{\nu}}_r, \delta_{yaw}) = \begin{bmatrix} 0 \\ Y_{\delta u^2} \\ -Y_{\delta u^2} l_x \end{bmatrix} \delta_{yaw} u_r^2 \tag{3.21b}$$

Note that $\boldsymbol{\nu}_r$ in (3.19) is replaced with $\bar{\boldsymbol{\nu}}_r$. Since only the relative surge speed u_r is used in the expressions, $\boldsymbol{\nu}_r$ and $\bar{\boldsymbol{\nu}}_r$ is equal as arguments to $\boldsymbol{\tau}, \boldsymbol{\tau}_{propeller}$ and $\boldsymbol{\tau}_{rudder}$.

By inserting (3.20) into (3.9b), the AUV dynamics can be expressed as:

$$\bar{\boldsymbol{M}}\dot{\bar{\boldsymbol{\nu}}}_r + \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r + \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r = \bar{\boldsymbol{\tau}}(\bar{\boldsymbol{\nu}}_r, \delta_{yaw}, n) \tag{3.22}$$

Solving (3.22) for $\dot{\bar{\boldsymbol{\nu}}}_r$:

$$\dot{\bar{\boldsymbol{\nu}}}_r = \bar{\boldsymbol{M}}^{-1}\left(\bar{\boldsymbol{\tau}}(\bar{\boldsymbol{\nu}}_r, \delta_{yaw}, n) - \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r\right) \tag{3.23}$$

The acceleration limits can be computed through evaluating the dynamic equations of the system. From (2.10c), the rudder deflection rate of change is limited by $|\dot{\delta}_{top}|, |\dot{\delta}_{bottom}| \leq \dot{\delta}_{max}$. By defining a time allowed for changing the rudder deflection angle, T_{rudd}, limits on the rudder position can be found. The propeller speed rate of change is not limited, but this may also be included. Including the propeller speed and rudder deflection angle limits from (2.10a) and (2.10b), the rudder deflection angle and propeller speed must be in the following sets:

$$\begin{aligned} \delta_{yaw} &\in \operatorname{sat}\left(\left[\delta^*_{yaw} + T_{rudd}2\dot{\delta}_{max}, \delta^*_{yaw} + T_{rudd}2\dot{\delta}_{max}\right], 2\delta_{max}\right) \\ n_p &\in [n_{p_{min}}, n_{p_{max}}] \end{aligned} \tag{3.24}$$

Where $\delta^*_{yaw} = \delta_{yaw}(t_0)$. Again, note that $\delta_{yaw} = \delta_{top} + \delta_{bottom}$, so that the constraints $|\dot{\delta}_{top}|, |\dot{\delta}_{bottom}| \leq \dot{\delta}_{max}$ and $|\delta_{top}|, |\delta_{bottom}| \leq \delta_{max}$ implies $|\dot{\delta}_{yaw}| \leq 2\dot{\delta}_{max}$ and $|\delta_{yaw}| \leq 2\delta_{max}$.

The possible AUV accelerations can be found by evaluating (3.23) for the current velocity $\bar{\boldsymbol{\nu}}^*_r = \bar{\boldsymbol{\nu}}(t_0)$ and rudder deflections and propeller speed as defined in (3.24). Since the propeller only creates surge force, and the rudders yaw torque, the acceleration limits at the current time step can be computed as:

$$\begin{aligned} \dot{\bar{\boldsymbol{\nu}}}_{r_{min}} &= \bar{\boldsymbol{M}}^{-1}\left(\bar{\boldsymbol{\tau}}(\bar{\boldsymbol{\nu}}^*_r, \max(\delta_{yaw}), \min(n_p)) - \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}^*_r)\bar{\boldsymbol{\nu}}^*_r - \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}^*_r)\bar{\boldsymbol{\nu}}^*_r\right) \\ \dot{\bar{\boldsymbol{\nu}}}_{r_{max}} &= \bar{\boldsymbol{M}}^{-1}\left(\bar{\boldsymbol{\tau}}(\bar{\boldsymbol{\nu}}^*_r, \min(\delta_{yaw}), \max(n_p)) - \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}^*_r)\bar{\boldsymbol{\nu}}^*_r - \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}^*_r)\bar{\boldsymbol{\nu}}^*_r\right) \end{aligned} \tag{3.25}$$

Where $\dot{\bar{\boldsymbol{\nu}}}_{r_{min}} = \begin{bmatrix}\dot{u}_{r_{min}} & \dot{v}_{r_{min}} & \dot{r}_{min}\end{bmatrix}^T$ and $\dot{\bar{\boldsymbol{\nu}}}_{r_{max}} = \begin{bmatrix}\dot{u}_{r_{max}} & \dot{v}_{r_{max}} & \dot{r}_{max}\end{bmatrix}^T$. It is worth noticing that a positive rudder deflection results in negative yaw moment.

From (3.25), the minimum and maximum surge and yaw accelerations can be extracted. The dynamic velocity window is then defined using these acceleration limits:

$$\begin{aligned} V_d = \Big\{(u_r, r) \in \mathbb{R} \times \mathbb{R} \Big| \, & u_r \in [u^*_r + \dot{u}_{r_{min}}T, u^* + \dot{u}_{r_{max}}T] \\ & \wedge r \in [r^* + \dot{r}_{min}T, r^* + \dot{r}_{max}T]\Big\} \end{aligned} \tag{3.26}$$

Possible velocities

Further, a set of possible velocities is found. The set contains all velocities the AUV can achieve, with respect to the actuator saturation limits. The possible velocities can be found through computing the steady state solution of the dynamics, for different rudder positions and propeller speed:

$$\begin{aligned} \bar{\boldsymbol{M}}\dot{\bar{\boldsymbol{\nu}}}_r &= \bar{\boldsymbol{\tau}}(\bar{\boldsymbol{\nu}}_r, \delta_{yaw}, n_p) - \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r = \mathbf{0} \\ \text{for } n_p &\in [0, n_{p_{max}}] \\ \text{and } \delta_{yaw} &\in [-2\delta_{max}, 2\delta_{max}] \end{aligned} \tag{3.27}$$

The AUV model is only valid for positive surge velocities, therefore only positive propeller speeds are considered. The set of possible velocities is defined as:

$$V_s = \big\{(u_r, r) \in \mathbb{R} \times \mathbb{R} \big| g(u_r, r) \geq 0\big\} \tag{3.28}$$

Where $g(u_r, r)$ is greater or equal to zero for valid solutions to (3.27), and negative otherwise. The function $g(u_r, r)$ is found through numerically computing the boundaries of the solutions to (3.27). Given m boundaries, defined by the functions $h_a(u_r, r) = 0, a \in \{1, 2, \ldots, m\}$ where $\nabla h_a(u_r, r)$ is required to be pointing inwards to the valid solutions, $g(u_r, r)$ is defined as:

$$g(u_r, r) = \min\left(h_1(u_r, r), h_2(u_r, r), \ldots, h_m(u_r, r)\right) \tag{3.29}$$

Figure 3.5 show a plot of discrete solutions of (3.27), and the computed boundary lines. A 3D plot of the function $g(u_r, r)$ is shown in figure 3.6.

The set of dynamically feasible velocities is then:

$$V_f = V_s \cap V_d \tag{3.30}$$

An illustration of V_s, V_d and V_f, given an initial velocity, is shown in figure 3.7.

Admissible velocities

Further, non-admissible velocities are removed [Eriksen, 2014]. For a velocity pair to be considered admissible, the AUV must be able to stop before it reaches the closest obstacle on the corresponding trajectory. The set of admissible velocities is given as:

$$\begin{aligned} V_a = \Big\{(u_r, r) \in \mathbb{R} \times \mathbb{R} \big| u_r \leq \sqrt{2\rho'(u_r, r)|\dot{u}_{min}|} \\ \wedge |r| \leq \left\{ \begin{array}{ll} \sqrt{2\rho'(u_r, r)|\dot{r}_{max}|} & : r < 0 \\ \sqrt{2\rho'(u_r, r)|\dot{r}_{min}|} & : r \geq 0 \end{array} \right\} \end{aligned} \tag{3.31}$$

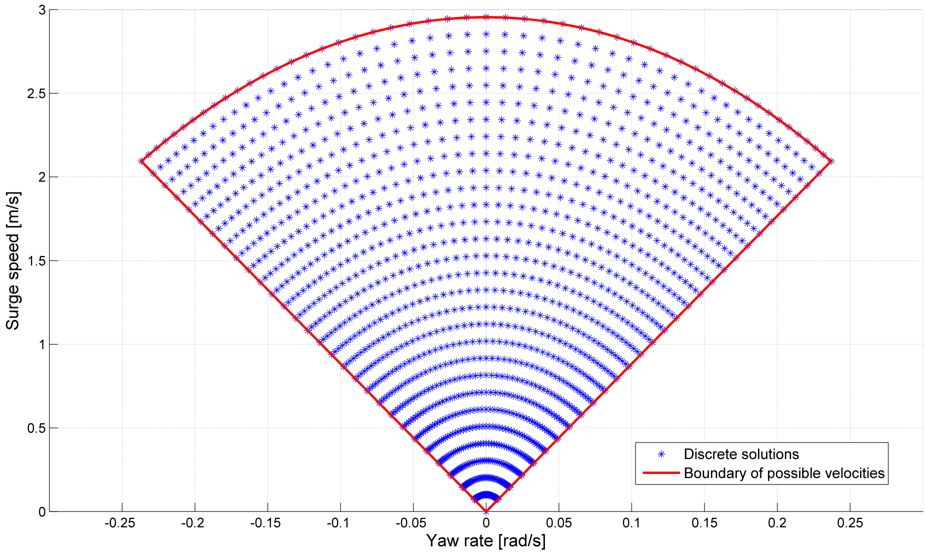


Figure 3.5: Possible combinations of surge speed and yaw rate, with respect to actuator saturation limits

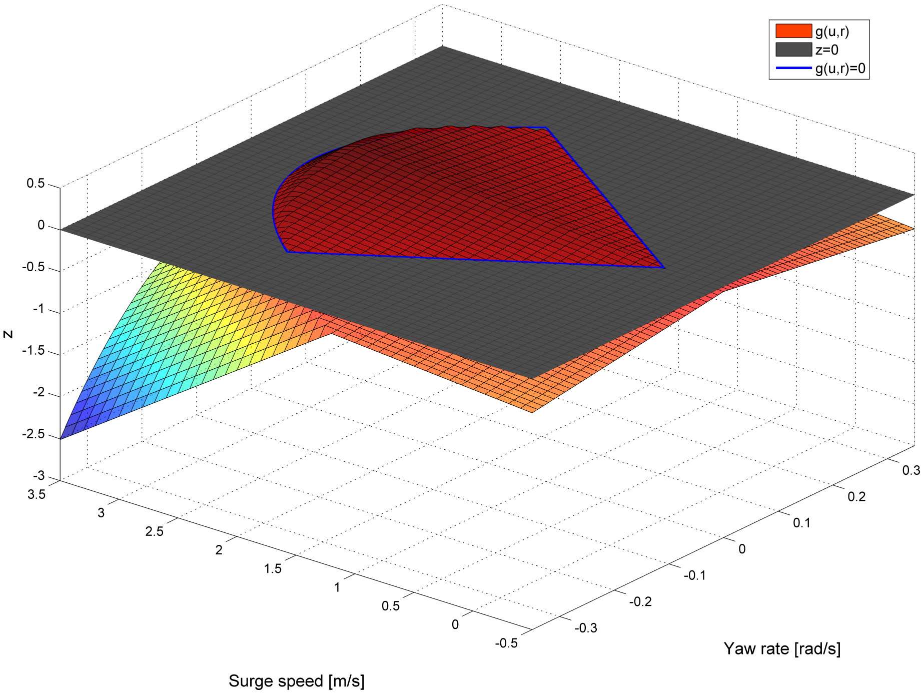


Figure 3.6: Function to find possible velocities

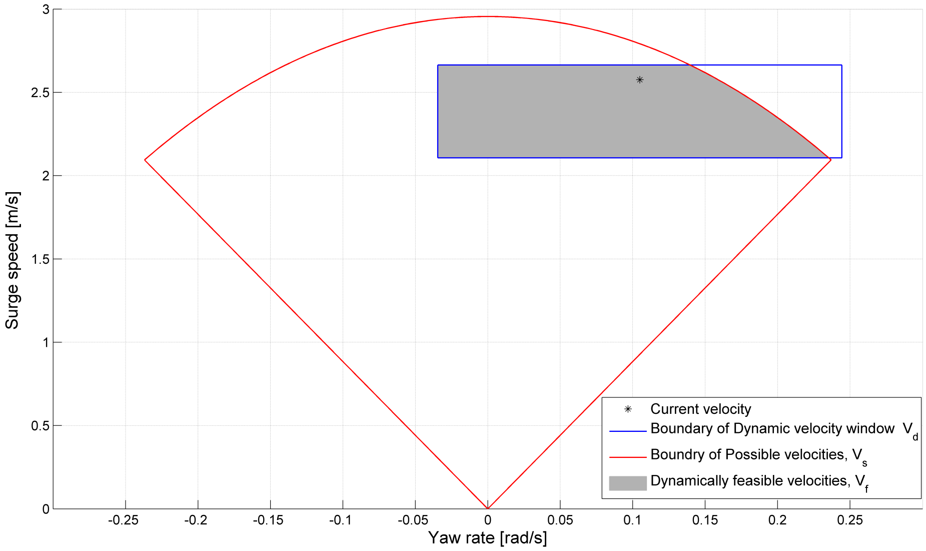


Figure 3.7: The dynamically feasible velocity set, together with the boundaries of the dynamic velocity window and the possible velocity set.

$\rho'(u_r, r)$ expresses the remaining distance the AUV can travel along the resulting trajectory at the next iteration without entering the antitarget region $\mathcal{T}$. It can be computed as:

$$\rho'(u_r, r) = \max(\rho(u_r, r) - \Delta_s, 0) \tag{3.32}$$

Where $\rho(u_r, r)$ expresses the distance the AUV can travel along the resulting trajectory before it enters $\mathcal{T}$ and Δ_s expresses the distance the AUV travels until the next iteration.

Finally, the search space is defined as:

$$V_r = V_f \cap V_a = V_s \cap V_d \cap V_a \tag{3.33}$$

3.4 Trajectory prediction

Most of the literature on obstacle avoidance using the Dynamic window algorithm is applied to robots with first order non-holonomic constraints. This makes it fairly accurate to model the robot trajectories as circular arcs. AUVs have second order non-holonomic constraints, which results in lateral velocities and makes the circular approximations inaccurate. Some alternatives to this already exists:

- [Schröter et al., 2007] used clothoid curves to model the trajectories. This is superior to using circular arcs, but is still an approximation.

- [Loe, 2008] approximated the sway speed using the steady state solution given the current surge speed and yaw rate, and simulated a simplified set of equations of motion to predict the trajectories.
- [Kiss and Tevesz, 2012] used a linear model of the vehicle dynamics to predict the trajectories.

To improve the predictions, it is proposed to use feedback linearization to linearize the surge and yaw dynamics, while leaving the sway motion uncontrolled. The closed loop dynamics is then derived and used for predicting the AUV trajectories. This will include the sway and controller dynamics in the AUV trajectory prediction. The approach is similar as the one presented in [Kiss and Tevesz, 2012], but does not require a linear model and is hence much more flexible. In contrast to the approach suggested by [Loe, 2008], the actual equations of motion is used, and a part of the system is solved analytically. This makes the prediction less computationally expensive.

Since the system is non-holonomic, it is not possible to fully linearize the system by feedback. Instead, the surge and yaw dynamics is linearized by partial feedback linearization. (3.9b) can be rewritten as:

$$\bar{\boldsymbol{M}}\dot{\bar{\boldsymbol{\nu}}}_r + \bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r) = \bar{\boldsymbol{B}}\boldsymbol{f} \tag{3.34}$$

Where $\bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r) = \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r + \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r$ contains the nonlinear parts of the equation. Since the mass matrix always is positive definite [Fossen, 2011] and hence of full rank, (3.34) can be rewritten as:

$$\begin{aligned}\dot{\bar{\boldsymbol{\nu}}}_r &= \bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} - \bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r)\\ &= \bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} - \bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\end{aligned} \tag{3.35}$$

Where $\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r) = \bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r)$.

To formulate the control law, the system is divided into two parts. This is done by using the matrices $\boldsymbol{\Gamma}_1$ and $\boldsymbol{\Gamma}_2$, which is defined as follows:

$$\begin{aligned}\boldsymbol{\Gamma}_1 &\triangleq \begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 1\end{bmatrix}\\ \boldsymbol{\Gamma}_2 &\triangleq \begin{bmatrix}0 & 1 & 0\end{bmatrix}\end{aligned} \tag{3.36}$$

The matrices satisfies $\boldsymbol{\Gamma}_1^T\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2 = \mathbf{I}$. Hence, the system equation (3.35) can be written as:

$$\begin{aligned}\dot{\bar{\boldsymbol{\nu}}}_r &= \left(\boldsymbol{\Gamma}_1^T\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\right)\left(\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} - \bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\right)\\ &= \boldsymbol{\Gamma}_1^T\left(\boldsymbol{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} - \boldsymbol{\Gamma}_1\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\right) + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} - \bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\right)\\ &= \boldsymbol{\Gamma}_1^T\left(\boldsymbol{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\boldsymbol{f} - \boldsymbol{\Gamma}_1\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\right) - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\end{aligned} \tag{3.37}$$

The last equality is due to the property $\mathbf{\Gamma}_2\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}} = \mathbf{0}_{1\times 2}$. The system is now divided into two parts. $\mathbf{\Gamma}_1^T\mathbf{\Gamma}_1$ map dynamics to surge and yaw, while $\mathbf{\Gamma}_2^T\mathbf{\Gamma}_2$ map dynamics to sway. To linearize the surge and yaw dynamics, the control law can be selected as:

$$\boldsymbol{f} = \left(\mathbf{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\right)^{-1}\left(\mathbf{\Gamma}_1\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r) + \boldsymbol{a}_1^b\right) \tag{3.38}$$

Where $\boldsymbol{a}_1^b = \begin{bmatrix}\dot{u}_{rd} & \dot{r}_d\end{bmatrix}^T$ is the desired surge and yaw accelerations.

Remark 3.3. *The matrix* $\left(\mathbf{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\right)^{-1}$ *always exists. By inserting the expression* $\bar{\boldsymbol{M}} = \boldsymbol{H}^T\boldsymbol{M}\boldsymbol{H}$, *and using* (3.10) *and* (3.6), *it is easy to show that:*

$$\begin{aligned}\mathbf{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}} &= \mathbf{\Gamma}_1\left(\boldsymbol{H}^T\boldsymbol{M}\boldsymbol{H}\right)^{-1}\boldsymbol{H}^T\boldsymbol{B}\\ &= \mathbf{\Gamma}_1\boldsymbol{H}^{-1}\boldsymbol{M}^{-1}\boldsymbol{B}\\ &= \begin{bmatrix}\frac{b_{11}}{m_{11}} & 0\\ 0 & \frac{m_{22}b_{32}-m_{23}b_{22}}{m_{22}m_{33}-m_{23}^2}\end{bmatrix}\end{aligned} \tag{3.39}$$

If ϵ *in* (3.4) *is well defined then* $m_{22}b_{32} - m_{23}b_{22} \neq 0$. *In addition,* $m_{22}m_{33} - m_{23}^2 > 0$ *since* $\boldsymbol{M}$ *is positive definite. Hence, since the system can be shown to be controllable in both surge and yaw, the matrix* $\mathbf{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}$ *is of full rank and thus invertible.*

Inserting (3.38) into (3.37) gives:

$$\begin{aligned}\dot{\bar{\boldsymbol{\nu}}}_r &= \mathbf{\Gamma}_1^T\left(\mathbf{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\left(\mathbf{\Gamma}_1\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{B}}\right)^{-1}\left(\mathbf{\Gamma}_1\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r) + \boldsymbol{a}_1^b\right) - \mathbf{\Gamma}_1\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\right)\\ &\qquad - \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\\ &= \mathbf{\Gamma}_1^T\boldsymbol{a}_1^b - \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\end{aligned} \tag{3.40}$$

The desired acceleration is selected as:

$$\begin{aligned}\boldsymbol{a}_1^b &= \dot{\bar{\boldsymbol{\nu}}}_{1r_d} - \boldsymbol{K}_p\left(\bar{\boldsymbol{\nu}}_{1r} - \bar{\boldsymbol{\nu}}_{1r_d}\right)\\ &= \dot{\bar{\boldsymbol{\nu}}}_{1r_d} - \boldsymbol{K}_p\left(\mathbf{\Gamma}_1\bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{\nu}}_{1r_d}\right)\end{aligned} \tag{3.41}$$

Where $\boldsymbol{K}_p = \begin{bmatrix}k_u & 0\\ 0 & k_r\end{bmatrix} > 0$ is a gain matrix, $\bar{\boldsymbol{\nu}}_{1r} = \begin{bmatrix}u_r & r\end{bmatrix}^T$ and $\bar{\boldsymbol{\nu}}_{1r_d} = \begin{bmatrix}u_{rd} & r_d\end{bmatrix}^T$.

Inserting the desired acceleration into the dynamics, and defining $\tilde{\boldsymbol{\nu}}_r = \bar{\boldsymbol{\nu}}_r - \mathbf{\Gamma}_1^T\bar{\boldsymbol{\nu}}_{1r_d} = \begin{bmatrix}\tilde{u}_r & \bar{v}_r & \tilde{r}\end{bmatrix}^T$:

$$\begin{aligned}\dot{\bar{\boldsymbol{\nu}}}_r &= \mathbf{\Gamma}_1^T\left(\dot{\bar{\boldsymbol{\nu}}}_{1r_d} - \boldsymbol{K}_p\left(\mathbf{\Gamma}_1\bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{\nu}}_{1r_d}\right)\right) - \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\\ \dot{\bar{\boldsymbol{\nu}}}_r - \mathbf{\Gamma}_1^T\dot{\bar{\boldsymbol{\nu}}}_{1r_d} &= \mathbf{\Gamma}_1^T\left(-\boldsymbol{K}_p\left(\mathbf{\Gamma}_1\bar{\boldsymbol{\nu}}_r - \mathbf{\Gamma}_1\mathbf{\Gamma}_1^T\bar{\boldsymbol{\nu}}_{1r_d}\right)\right) - \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\\ \dot{\bar{\boldsymbol{\nu}}}_r - \mathbf{\Gamma}_1^T\dot{\bar{\boldsymbol{\nu}}}_{1r_d} &= -\mathbf{\Gamma}_1^T\boldsymbol{K}_p\mathbf{\Gamma}_1\left(\bar{\boldsymbol{\nu}}_r - \mathbf{\Gamma}_1^T\bar{\boldsymbol{\nu}}_{1r_d}\right) - \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\\ \dot{\tilde{\boldsymbol{\nu}}}_r &= -\mathbf{\Gamma}_1^T\boldsymbol{K}_p\mathbf{\Gamma}_1\tilde{\boldsymbol{\nu}}_r - \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)\end{aligned} \tag{3.42}$$

Note that the term $\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)$ takes $\bar{\boldsymbol{\nu}}_r = \tilde{\boldsymbol{\nu}}_r + \boldsymbol{\Gamma}_1^T \bar{\boldsymbol{\nu}}_{1r_d}$ as its argument.

(3.42) is linear in all terms, except the last. It is important to notice that the last term only influences the sway motion, as it is mapped through $\boldsymbol{\Gamma}_2^T \boldsymbol{\Gamma}_2$. Hence, the equation is linear in both surge and yaw rate. In component form (3.42) reads:

$$\begin{aligned} \dot{\tilde{u}}_r &= -k_{11}\tilde{u}_r \\ \dot{\tilde{v}}_r &= -\bar{n}'_2(\bar{\boldsymbol{\nu}}_r) \\ \dot{\tilde{r}} &= -k_{22}\tilde{r} \end{aligned} \tag{3.43}$$

Where $\bar{n}'_2(\bar{\boldsymbol{\nu}}_r)$ is the contribution from the Coriolis-centripetal and damping matrices in sway, $\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r) = \begin{bmatrix} \bar{n}'_1(\bar{\boldsymbol{\nu}}_r) & \bar{n}'_2(\bar{\boldsymbol{\nu}}_r) & \bar{n}'_3(\bar{\boldsymbol{\nu}}_r) \end{bmatrix}^T$. It is given as:

$$\begin{aligned} \bar{n}'_2(\bar{\boldsymbol{\nu}}_r) = \frac{1}{\bar{m}_{22}\bar{m}_{33} - \bar{m}_{23}^2} \Big(& \left(\bar{m}_{33}\bar{d}_{22} - \bar{m}_{23}\bar{d}_{32}\right) \bar{v}_r \\ & - \bar{m}_{23}\left(\bar{m}_{22} - \bar{m}_{11}\right) u_r \bar{v}_r + \left(\bar{m}_{33}\bar{m}_{11} - \bar{m}_{23}^2\right) u_r r \\ & + \left(\bar{m}_{33}\bar{d}_{23} - \bar{m}_{23}\bar{d}_{33}\right) r \Big) \end{aligned} \tag{3.44}$$

Where $\bar{m}_a,\ a \in \{11, 22, 23, 33\}$ and $\bar{d}_a,\ a \in \{11, 22, 23, 32, 33\}$ are coefficients of the matrices in (3.9b):

$$\begin{aligned} \bar{\boldsymbol{M}} &= \begin{bmatrix} \bar{m}_{11} & 0 & 0 \\ 0 & \bar{m}_{22} & \bar{m}_{23} \\ 0 & \bar{m}_{23} & \bar{m}_{33} \end{bmatrix} \\ \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r) &= \begin{bmatrix} \bar{d}_{11}(u_r) & 0 & 0 \\ 0 & \bar{d}_{22} & \bar{d}_{23} \\ 0 & \bar{d}_{32} & \bar{d}_{33} \end{bmatrix} \end{aligned} \tag{3.45}$$

A derivation of (3.44) is shown in appendix 3.A.

By approximating the last term of (3.42) using a first order Taylor series, the dynamics of the AUV is linear in all terms:

$$\begin{aligned} \bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r) &\approx \bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r^*) + \underbrace{\left.\frac{\mathrm{d}\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)}{\mathrm{d}\bar{\boldsymbol{\nu}}_r}\right|_{\bar{\boldsymbol{\nu}}_r = \bar{\boldsymbol{\nu}}_r^*}}_{\boldsymbol{N}} (\bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{\nu}}_r^*) \\ &= \bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r^*) + \boldsymbol{N}\bar{\boldsymbol{\nu}}_r - \boldsymbol{N}\bar{\boldsymbol{\nu}}_r^* \\ &= \boldsymbol{N}\bar{\boldsymbol{\nu}}_r + \boldsymbol{b}(\bar{\boldsymbol{\nu}}_r^*) \end{aligned} \tag{3.46}$$

Where $\bar{\boldsymbol{\nu}}_r^* = \bar{\boldsymbol{\nu}}_r(t_0)$ is a linearization point and $\boldsymbol{b}(\bar{\boldsymbol{\nu}}_r^*) = \bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r^*) - \boldsymbol{N}\bar{\boldsymbol{\nu}}_r^*$ is a constant term. The matrix $\boldsymbol{N}$ is given as:

$$\begin{aligned}\boldsymbol{N} &= \left.\frac{\mathrm{d}\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)}{\mathrm{d}\bar{\boldsymbol{\nu}}_r}\right|_{\bar{\boldsymbol{\nu}}_r=\bar{\boldsymbol{\nu}}_r^*} \\ &= \begin{bmatrix} \frac{\bar{d}_{11}(u_r^*)+\left.\frac{\mathrm{d}\bar{d}_{11}(u_r)}{\mathrm{d}u_r}\right|_{\bar{\boldsymbol{\nu}}_r=\bar{\boldsymbol{\nu}}_r^*} u_r^*}{\bar{m}_{11}} & -\frac{\bar{m}_{22}r^*}{\bar{m}_{11}} & \frac{-\bar{m}_{22}\bar{v}_r^*-2\bar{m}_{23}r^*}{\bar{m}_{11}} \\ \frac{-\bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})\bar{v}_r^*+(\bar{m}_{33}\bar{m}_{11}-\bar{m}_{23}^2)r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \frac{\bar{m}_{33}\bar{d}_{22}-\bar{m}_{23}\bar{d}_{32}-\bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})u_r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \frac{(\bar{m}_{33}\bar{m}_{11}-\bar{m}_{23}^2)u_r^*+\bar{m}_{33}\bar{d}_{23}-\bar{m}_{23}\bar{d}_{33}}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \\ \frac{-\bar{m}_{22}(\bar{m}_{11}-\bar{m}_{22})\bar{v}_r^*+\bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \frac{\bar{m}_{22}\bar{d}_{32}-\bar{m}_{23}\bar{d}_{22}-\bar{m}_{22}(\bar{m}_{11}-\bar{m}_{22})u_r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \frac{\bar{m}_{22}\bar{d}_{33}-\bar{m}_{23}\bar{d}_{23}+\bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})u_r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \end{bmatrix}\end{aligned} \tag{3.47}$$

A derivation of (3.47) is shown in appendix 3.A.

Inserting (3.46) into (3.42) results in:

$$\begin{aligned}\dot{\tilde{\boldsymbol{\nu}}}_r &= -\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1\tilde{\boldsymbol{\nu}}_r - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{N}\bar{\boldsymbol{\nu}}_r + \boldsymbol{b}(\bar{\boldsymbol{\nu}}_r^*)\right) \\ &= -\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1\tilde{\boldsymbol{\nu}}_r - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{N}\left(\tilde{\boldsymbol{\nu}}_r + \boldsymbol{\Gamma}_1^T\bar{\boldsymbol{\nu}}_{1r_d}\right) - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{b}(\bar{\boldsymbol{\nu}}_r^*) \\ &= -\left(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{N}\right)\tilde{\boldsymbol{\nu}}_r - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{N}\boldsymbol{\Gamma}_1^T\bar{\boldsymbol{\nu}}_{1r_d} - \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{b}(\bar{\boldsymbol{\nu}}_r^*)\end{aligned} \tag{3.48}$$

(3.48) can be written as:

$$\dot{\tilde{\boldsymbol{\nu}}}_r = \boldsymbol{A}\tilde{\boldsymbol{\nu}}_r + \boldsymbol{\beta}\bar{\boldsymbol{\nu}}_{1r_d} + \boldsymbol{G} \tag{3.49}$$

Where:

$$\begin{aligned}\boldsymbol{A} &= -\left(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{N}\right) \\ \boldsymbol{\beta} &= -\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{N}\boldsymbol{\Gamma}_1^T \\ \boldsymbol{G} &= -\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{b}(\bar{\boldsymbol{\nu}}_r^*)\end{aligned} \tag{3.50}$$

This is not a true linear system, due to the constant $\boldsymbol{G}$. Hence $\tilde{\boldsymbol{\nu}}_r = \boldsymbol{0}$ and $\bar{\boldsymbol{\nu}}_{1r_d} \equiv \boldsymbol{0}$ does not imply $\dot{\tilde{\boldsymbol{\nu}}}_r = \boldsymbol{0}$. However, by considering $\boldsymbol{G}$ as an input to the system, (3.49) can be solved as:

$$\begin{aligned}\tilde{\boldsymbol{\nu}}_r(t) &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\tilde{\boldsymbol{\nu}}_r(t_0) + \int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\sigma)}\boldsymbol{\beta}\bar{\boldsymbol{\nu}}_{1r_d}(\sigma)\mathrm{d}\sigma + \int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\sigma)}\boldsymbol{G}\mathrm{d}\sigma \\ &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\tilde{\boldsymbol{\nu}}_r(t_0) + \int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\sigma)}\left(\boldsymbol{\beta}\bar{\boldsymbol{\nu}}_{1r_d}(\sigma) + \boldsymbol{G}\right)\mathrm{d}\sigma\end{aligned} \tag{3.51}$$

Since the desired surge speed and yaw rate is constant for each trajectory, $\bar{\boldsymbol{\nu}}_{1r_d}$ is constant for each trajectory. Also by letting $t_0 = 0$ s, (3.51) can be solved as: [Hespanha, 2009]

$$\tilde{\boldsymbol{\nu}}_r(t) = \mathrm{e}^{\boldsymbol{A}t}\tilde{\boldsymbol{\nu}}_r(0) - \boldsymbol{A}^{-1}\left(\boldsymbol{I} - \mathrm{e}^{\boldsymbol{A}t}\right)(\beta\bar{\boldsymbol{\nu}}_{1r_d} + \boldsymbol{G}) \tag{3.52}$$

The solution to the dynamics can then be used to compute the AUV position for a selected velocity pair. The system is simulated in discrete time using a standard simulation algorithm.

Assumption 3.3. *The ocean current, $\boldsymbol{V}_c$, is assumed to be known to the Dynamic Window algorithm.*

Remark 3.4. *The ocean current can be estimated in different ways, for example by using the steady state solution of the ILOS guidance law [Caharija, 2014].*

From (3.9a), the kinematics are modeled as:

$$\dot{\bar{\boldsymbol{\eta}}} = \boldsymbol{R}(\bar{\boldsymbol{\eta}})\bar{\boldsymbol{\nu}}_r + \begin{bmatrix}\boldsymbol{V}_c\\ 0\end{bmatrix} \tag{3.53}$$

$\bar{\boldsymbol{\nu}}_r$ can be computed from (3.52), while assumption 3.3 guarantees knowledge of $\boldsymbol{V}_c$. Hence, $\dot{\bar{\boldsymbol{\eta}}}$ can be computed, and the system can be simulated. The modified Euler method is used for simulating the kinematics. For a system $\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y}, t)$, the method is given as: [Egeland and Gravdahl, 2003]

$$\begin{aligned}\boldsymbol{k}_1 &= \boldsymbol{f}\left(\boldsymbol{y}(t_n), t_n\right)\\ \boldsymbol{k}_2 &= \boldsymbol{f}\left(\boldsymbol{y}(t_n) + \frac{h}{2}\boldsymbol{k}_1, t_n + \frac{h}{2}\right)\\ \boldsymbol{y}(t_{n+1}) &= \boldsymbol{y}(t_n) + h\boldsymbol{k}_2\end{aligned} \tag{3.54}$$

Where h is the discretization time step.

Hence, the system $\dot{\bar{\boldsymbol{\eta}}} = \boldsymbol{f}(\bar{\boldsymbol{\eta}}, t) = \boldsymbol{R}(\bar{\boldsymbol{\eta}})\bar{\boldsymbol{\nu}}_r(t) + \begin{bmatrix}\boldsymbol{V}_c & 0\end{bmatrix}^T$ can be simulated as:

$$\begin{aligned}\boldsymbol{k}_1 &= \boldsymbol{R}(\bar{\boldsymbol{\eta}}(t_n))\bar{\boldsymbol{\nu}}_r(t_n)\\ \boldsymbol{k}_2 &= \boldsymbol{R}(\bar{\boldsymbol{\eta}}(t_n) + \frac{h}{2}\boldsymbol{k}_1)\bar{\boldsymbol{\nu}}_r(t_n + \frac{h}{2})\\ \bar{\boldsymbol{\eta}}(t_{n+1}) &= \bar{\boldsymbol{\eta}}(t_n) + h\left(\boldsymbol{k}_2 + \begin{bmatrix}\boldsymbol{V}_c\\ 0\end{bmatrix}\right)\end{aligned} \tag{3.55}$$

To illustrate the improvement of predicting the AUV trajectories using a linear approximation, a set of AUV trajectories is computed for a search space consisting of three desired surge velocities and three desired yaw rates. This results in nine velocity pairs. The initial velocity is chosen as $\bar{\boldsymbol{\nu}}_r(0) = \begin{bmatrix}2 & 0 & 0\end{bmatrix}^T$, and the trajectories are predicted for 30 s without ocean current. Figure 3.8 show the actual AUV trajectories together with predicted trajectories using both the circular approximation and the linear approximation. Table 3.1 show the average mean square error and root mean square error of the predicted AUV trajectories using the circular and linear approximations. From table 3.1 and figure 3.8 it is clear that the linear approximation is much more accurate, especially for the first part of trajectory.

Table 3.1 Mean square error (MSE) and Root mean square error (RMSE) of predicted AUV trajectories

Time span	**Circular approximation**		**Linear approximation**	
	$t = [0, 30]$ s	$t = [0, 5]$ s	$t = [0, 30]$ s	$t = [0, 5]$ s
MSE	4.67 m^2	0.100 m^2	0.045 m^2	0.000576 m^2
RMSE	2.16 m	0.316 m	0.213 m	0.0240 m

If it is not desirable to have the AUV model defined in the pivot point, the linear trajectory prediction can done with the model defined in the CO of $\{b\}$ by changing the control law (3.38) to:

$$\boldsymbol{f} = \boldsymbol{\Upsilon} \left(\boldsymbol{\Gamma}_1 \boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{a}_1^b \right) \tag{3.56}$$

Where $\boldsymbol{\Upsilon} = \left(\boldsymbol{\Gamma}_1 \boldsymbol{M}^{-1} \boldsymbol{B} \right)^{-1}$ and $\boldsymbol{n}'(\boldsymbol{\nu}_r) = \boldsymbol{M}^{-1} \left(\boldsymbol{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r \right)$. Further, (3.49) must be changed to:

$$\begin{aligned} \boldsymbol{A} &= -\Big(\boldsymbol{\Gamma}_1^T \boldsymbol{K}_p \boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T \boldsymbol{\Gamma}_2 \left(\boldsymbol{\Upsilon}' \left(\boldsymbol{K}_p \boldsymbol{\Gamma}_1 - \boldsymbol{\Gamma}_1 \boldsymbol{N} \right) + \boldsymbol{N} \right) \Big) \\ \boldsymbol{\beta} &= \boldsymbol{\Gamma}_2^T \boldsymbol{\Gamma}_2 \left(\boldsymbol{\Upsilon}' \boldsymbol{\Gamma}_1 - \mathbf{I} \right) \boldsymbol{N} \boldsymbol{\Gamma}_1^T \\ \boldsymbol{G} &= \boldsymbol{\Gamma}_2^T \boldsymbol{\Gamma}_2 \left(\boldsymbol{\Upsilon}' \boldsymbol{\Gamma}_1 - \mathbf{I} \right) \boldsymbol{b}(\boldsymbol{\nu}_r^*) \end{aligned} \tag{3.57}$$

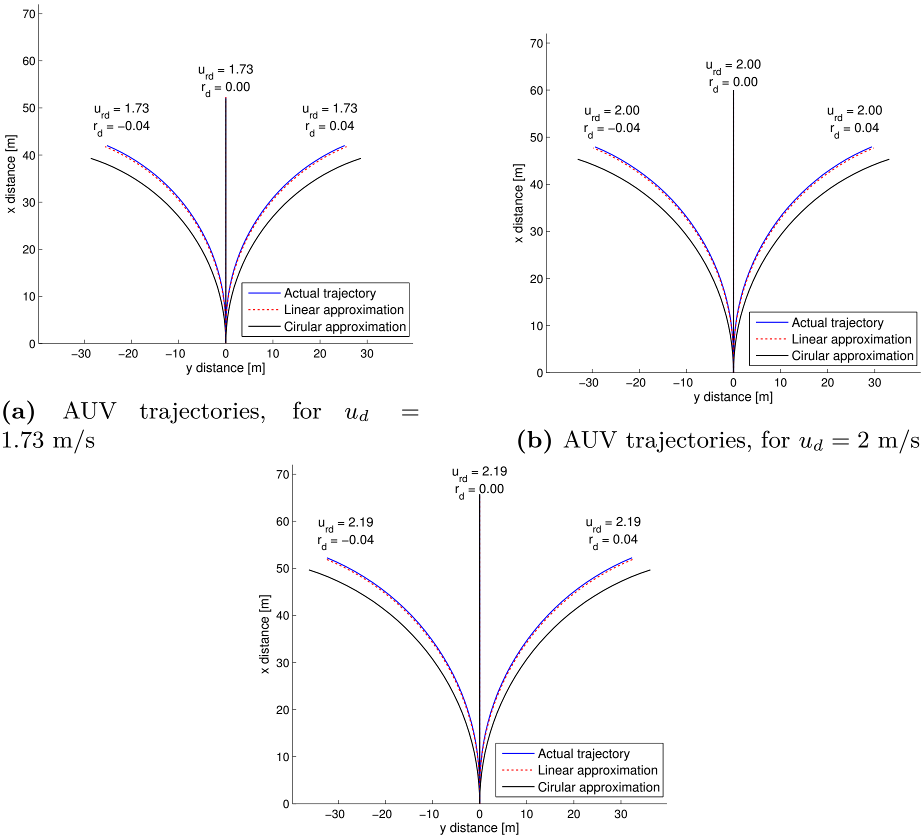
Where $\boldsymbol{\Upsilon}' = \boldsymbol{M}^{-1} \boldsymbol{B} \boldsymbol{\Upsilon}$. Since the trajectory prediction is done in the CO of $\{b\}$ (3.52) and (3.55) are changed to:

$$\tilde{\boldsymbol{\nu}}_r(t) = \mathrm{e}^{\boldsymbol{A}t} \tilde{\boldsymbol{\nu}}_r(0) - \boldsymbol{A}^{-1} \left(\boldsymbol{I} - \mathrm{e}^{\boldsymbol{A}t} \right) \left(\boldsymbol{\beta} \bar{\boldsymbol{\nu}}_{1r_d} + \boldsymbol{G} \right) \tag{3.58}$$

and

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{R}(\boldsymbol{\eta}(t_n)) \boldsymbol{\nu}_r(t_n) \\ \boldsymbol{k}_2 &= \boldsymbol{R}(\boldsymbol{\eta}(t_n) + \frac{h}{2} \boldsymbol{k}_1) \boldsymbol{\nu}_r(t_n + \frac{h}{2}) \\ \boldsymbol{\eta}(t_{n+1}) &= \boldsymbol{\eta}(t_n) + h \left(\boldsymbol{k}_2 + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \right) \end{aligned} \tag{3.59}$$

Where $\tilde{\boldsymbol{\nu}}_r$ now is defined as $\tilde{\boldsymbol{\nu}}_r = \boldsymbol{\nu}_r - \boldsymbol{\Gamma}_1^T \boldsymbol{\nu}_{1r_d}$. The details are shown in appendix 3.B.



(a) AUV trajectories, for u_d = 1.73 m/s

(b) AUV trajectories, for $u_d = 2$ m/s

(c) AUV trajectories, for u_d = 2.19 m/s

Figure 3.8: Actual and approximated AUV trajectories, given initial velocity $\bar{\boldsymbol{\nu}}_r(t_0) = [2 \quad 0 \quad 0]^T$

3.5 Objective function

To make the algorithm as general as possible, and to support a layered implementation, the objective function is modified to take a desired surge speed and yaw rate as inputs. This is inspired by [Berti et al., 2008] and [Inñigo Blasco et al., 2014]. Hence, the algorithm works entirely in the velocity space by taking speed and rate as both inputs and outputs.

In [Eriksen, 2014] the velocity term favored high linear velocities, using a linear function in surge speed. The term is modified to be focused around a desired surge speed U_{rd} taken as input to the algorithm:

$$\text{velocity}(u_r, r, U_{rd}) = 1 - \frac{|U_{rd} - u|}{\max\limits_{u \in V_r}(|U_{rd} - u|)} \tag{3.60}$$

The original DWA, and the implementation in [Eriksen, 2014] used a heading term which took a desired heading as an input. The term predicted the resulting heading of a velocity pair, and compared the predicted and desired heading. To improve the generality and flexibility, this term is replaced with a *yawrate* term which is focused around a desired yaw rate r'_d taken as input to the algorithm:

$$\text{yawrate}(u_r, r, r'_d) = 1 - \frac{|r'_d - r|}{\max\limits_{r \in V_r}(|r'_d - r|)} \tag{3.61}$$

Further, the *dist* term is scaled by the trajectory velocity. This results in a term which approximates the time until collision, which is a more intuitive measure than the distance until collision. This approach is inspired by [Seder et al., 2005]:

$$\text{dist}(u_r, r) = \frac{\bar{\rho}(u_r, r)}{\frac{1}{T}\int_0^T \|\boldsymbol{\chi}(u_r, r, t)\|_2 \, \mathrm{d}t} \tag{3.62}$$

Where $\bar{\rho}(u_r, r)$ is the distance the AUV can travel along the trajectory specified by the velocity pair (u_r, r) until it enters Ω. $\boldsymbol{\chi}(u_r, r, t)$ is the AUV velocity, including the ocean current, along the trajectory specified by the velocity pair (u_r, r). This can be computed using the prediction in (3.52) and (3.55):

$$\begin{aligned}\boldsymbol{\chi}(u_d, r_d, t) = \begin{bmatrix} u(t) \\ \bar{v}(t) \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \bar{\boldsymbol{\nu}}(t | u_{rd}, r_d) \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \Big(\bar{\boldsymbol{\nu}}_r(t | u_{rd}, r_d) \\ &\qquad + \boldsymbol{R}(\bar{\boldsymbol{\eta}}(t | u_{rd}, r_d))^T \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \Big)\end{aligned} \tag{3.63}$$

Where the notation $\bar{\boldsymbol{\nu}}_r(t|u_{rd}, r_d)$ and $\bar{\boldsymbol{\eta}}(t|u_{rd}, r_d)$ denotes the solutions to (3.52) and (3.55) given $\bar{\boldsymbol{\nu}}_{1r_d} = \begin{bmatrix} u_{rd} & r_d \end{bmatrix}$. It is worth noting that the ocean current is included in (3.63). This is necessary since $\bar{\rho}(u, r)$ is computed as the absolute distance the AUV travels, from the predicted trajectories in (3.55). Also note that if no obstacles are present, all the trajectories will have an equal distance measure. The low pass filter σ in the original objective function (2.22) is omitted, since no measurement noise is included. The objective function is then given as:

$$G(u_r, r) = \alpha \cdot \text{yawrate}(u_r, r, r_d') + \beta \cdot \text{dist}(u_r, r) + \gamma \cdot \text{velocity}(u_r, r, U_{rd}) \quad (3.64)$$

It is worth noting that the yawrate(u_r, r, r_d') and velocity(u_r, r, U_{rd}) terms are dimensionless, and that dist(u_r, r) is given in seconds. The scaling constants α and γ are therefore dimensionless, while β is specified in s^{-1}. This results in a dimensionless objective function.

3.A Appendix: Functional expressions

The nonlinear dynamics $\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)$ is given as:

$$
\begin{aligned}
\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r) &= \bar{\boldsymbol{M}}^{-1}\left(\bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r) + \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\right)\bar{\boldsymbol{\nu}}_r \\
&= \begin{bmatrix} \bar{m}_{11} & 0 & 0 \\ 0 & \bar{m}_{22} & \bar{m}_{23} \\ 0 & \bar{m}_{23} & \bar{m}_{33} \end{bmatrix}^{-1} \Bigg(\\
&\qquad \begin{bmatrix} 0 & 0 & -\bar{m}_{22}\bar{v}_r - \bar{m}_{23}r \\ 0 & 0 & \bar{m}_{11}u_r \\ \bar{m}_{22}\bar{v}_r + \bar{m}_{23}r & -\bar{m}_{11}u_r & 0 \end{bmatrix} \\
&\qquad\qquad + \begin{bmatrix} \bar{d}_{11}(u_r) & 0 & 0 \\ 0 & \bar{d}_{22} & \bar{d}_{23} \\ 0 & \bar{d}_{32} & \bar{d}_{33} \end{bmatrix} \Bigg) \begin{bmatrix} u_r \\ \bar{v}_r \\ r \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{\bar{m}_{11}} & 0 & 0 \\ 0 & \frac{\bar{m}_{33}}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \frac{-\bar{m}_{23}}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \\ 0 & \frac{-\bar{m}_{23}}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \frac{\bar{m}_{22}}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \end{bmatrix} \\
&\qquad \begin{bmatrix} \bar{d}_{11}(u_r)u_r - \bar{m}_{22}\bar{v}_r r - \bar{m}_{23}r^2 \\ \bar{d}_{22}\bar{v}_r + \bar{d}_{23}r + \bar{m}_{11}u_r r \\ (\bar{m}_{22} - \bar{m}_{11})\bar{u}_r v_r + \bar{m}_{23}ru_r + \bar{d}_{32}\bar{v}_r + \bar{d}_{33}r \end{bmatrix} \\
&= \begin{bmatrix} \frac{\bar{d}_{11}(u_r)u_r - \bar{m}_{22}\bar{v}_r r - \bar{m}_{23}r^2}{\bar{m}_{11}} \\
\frac{1}{\bar{m}_{22}\bar{m}_{33} - \bar{m}_{23}^2}\Big(\left(\bar{m}_{33}\bar{d}_{22} - \bar{m}_{23}\bar{d}_{32}\right)\bar{v}_r \\
\quad - \bar{m}_{23}\left(\bar{m}_{22} - \bar{m}_{11}\right)u_r\bar{v}_r + \left(\bar{m}_{33}\bar{m}_{11} - \bar{m}_{23}^2\right)u_r r \\
\qquad + \left(\bar{m}_{33}\bar{d}_{23} - \bar{m}_{23}\bar{d}_{33}\right)r\Big) \\
\frac{1}{\bar{m}_{22}\bar{m}_{33} - \bar{m}_{23}^2}\Big(\left(\bar{m}_{22}\bar{d}_{33} - \bar{m}_{23}\bar{d}_{23}\right)r \\
\quad - \bar{m}_{22}\left(\bar{m}_{11} - \bar{m}_{22}\right)u_r\bar{v}_r + \bar{m}_{23}\left(\bar{m}_{22} - \bar{m}_{11}\right)u_r r \\
\qquad + \left(\bar{m}_{22}\bar{d}_{32} - \bar{m}_{23}\bar{d}_{22}\right)\bar{v}_r\Big) \end{bmatrix}
\end{aligned}
\tag{3.65}
$$

The Jacobian matrix N is given as:

$$\boldsymbol{N} = \left.\frac{\mathrm{d}\bar{\boldsymbol{n}}'(\bar{\boldsymbol{\nu}}_r)}{\mathrm{d}\bar{\boldsymbol{\nu}}_r}\right|_{\bar{\boldsymbol{\nu}}_r=\bar{\boldsymbol{\nu}}_r^*} = \left.\begin{bmatrix} \frac{\partial \bar{n}_1'(\bar{\boldsymbol{\nu}}_r)}{\partial u_r} & \frac{\partial \bar{n}_1'(\bar{\boldsymbol{\nu}}_r)}{\partial \bar{v}_r} & \frac{\partial \bar{n}_1'(\bar{\boldsymbol{\nu}}_r)}{\partial r} \\ \frac{\partial \bar{n}_2'(\bar{\boldsymbol{\nu}}_r)}{\partial u_r} & \frac{\partial \bar{n}_2'(\bar{\boldsymbol{\nu}}_r)}{\partial \bar{v}_r} & \frac{\partial \bar{n}_2'(\bar{\boldsymbol{\nu}}_r)}{\partial r} \\ \frac{\partial \bar{n}_3'(\bar{\boldsymbol{\nu}}_r)}{\partial u_r} & \frac{\partial \bar{n}_3'(\bar{\boldsymbol{\nu}}_r)}{\partial \bar{v}_r} & \frac{\partial \bar{n}_3'(\bar{\boldsymbol{\nu}}_r)}{\partial r} \end{bmatrix}\right|_{\bar{\boldsymbol{\nu}}_r=\bar{\boldsymbol{\nu}}_r^*}$$

$$= \left[\begin{matrix} \dfrac{\bar{d}_{11}(u_r^*) + \left.\frac{\mathrm{d}\bar{d}_{11}(u_r)}{\mathrm{d}u_r}\right|_{\bar{\boldsymbol{\nu}}_r=\bar{\boldsymbol{\nu}}_r^*} u_r^*}{\bar{m}_{11}} \\ \dfrac{-\bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})\bar{v}_r^* + (\bar{m}_{33}\bar{m}_{11}-\bar{m}_{23}^2)r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \\ \dfrac{-\bar{m}_{22}(\bar{m}_{11}-\bar{m}_{22})\bar{v}_r^* + \bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \end{matrix}\right. \tag{3.66}$$

$$\left.\begin{matrix} -\dfrac{\bar{m}_{22}r^*}{\bar{m}_{11}} & \dfrac{-\bar{m}_{22}\bar{v}_r^* - 2\bar{m}_{23}r^*}{\bar{m}_{11}} \\ \dfrac{\bar{m}_{33}\bar{d}_{22} - \bar{m}_{23}\bar{d}_{32} - \bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})u_r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \dfrac{(\bar{m}_{33}\bar{m}_{11}-\bar{m}_{23}^2)u_r^* + \bar{m}_{33}\bar{d}_{23} - \bar{m}_{23}\bar{d}_{33}}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \\ \dfrac{\bar{m}_{22}\bar{d}_{32} - \bar{m}_{23}\bar{d}_{22} - \bar{m}_{22}(\bar{m}_{11}-\bar{m}_{22})u_r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} & \dfrac{\bar{m}_{22}\bar{d}_{33} - \bar{m}_{23}\bar{d}_{23} + \bar{m}_{23}(\bar{m}_{22}-\bar{m}_{11})u_r^*}{\bar{m}_{22}\bar{m}_{33}-\bar{m}_{23}^2} \end{matrix}\right]$$

3.B Appendix: Trajectory prediction without pivot point transformation

Given a model in 3 DOF, as (3.1a) and (3.1b):

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\boldsymbol{\eta})\boldsymbol{\nu}_r + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \tag{3.67a}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}}_r + \boldsymbol{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r = \boldsymbol{B}\boldsymbol{f} \tag{3.67b}$$

Since the system is non-holonomic, it is not possible to fully linearize the system by feedback. Instead, the surge and yaw dynamics is linearized by partial feedback linearization. (3.67b) can be rewritten as:

$$\dot{\boldsymbol{\nu}}_r = \boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} - \boldsymbol{n}'(\boldsymbol{\nu}_r) \tag{3.68}$$

Where $\boldsymbol{n}'(\boldsymbol{\nu}_r) = \boldsymbol{M}^{-1}\left(\boldsymbol{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r\right)$.

Since the control input only contains two independent controls, only two DOF's can be controlled. Surge and yaw is selected to be controlled, and sway is left uncontrolled. The controller is selected to remove the non-linearities in surge and yaw by the use of feedback linearization. To formulate the control law, the system is divided into two parts. This is done by using the matrices $\boldsymbol{\Gamma}_1$ and $\boldsymbol{\Gamma}_2$, which is defined as follows:

$$\begin{aligned} \boldsymbol{\Gamma}_1 &\triangleq \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ \boldsymbol{\Gamma}_2 &\triangleq \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \end{aligned} \tag{3.69}$$

The matrices satisfies $\mathbf{\Gamma}_1^T\mathbf{\Gamma}_1 + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2 = \mathbf{I}$. Hence, the system equation (3.68) can be written as:

$$\begin{aligned}\dot{\boldsymbol{\nu}}_r &= \left(\mathbf{\Gamma}_1^T\mathbf{\Gamma}_1 + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\right)\left(\boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} - \boldsymbol{n}'(\boldsymbol{\nu}_r)\right)\\ &= \mathbf{\Gamma}_1^T\left(\mathbf{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} - \mathbf{\Gamma}_1\boldsymbol{n}'(\boldsymbol{\nu}_r)\right) + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\left(\boldsymbol{M}^{-1}\boldsymbol{B}\boldsymbol{f} - \boldsymbol{n}'(\boldsymbol{\nu}_r)\right)\end{aligned} \tag{3.70}$$

The system is now divided into two parts. $\mathbf{\Gamma}_1^T\mathbf{\Gamma}_1$ map dynamics to surge and yaw, while $\mathbf{\Gamma}_2^T\mathbf{\Gamma}_2$ map dynamics to sway. To linearize the surge and yaw dynamics, the control law can be selected as:

$$\begin{aligned}\boldsymbol{f} &= \left(\mathbf{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}\right)^{-1}\left(\mathbf{\Gamma}_1\boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{a}_1^b\right)\\ &= \mathbf{\Upsilon}\left(\mathbf{\Gamma}_1\boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{a}_1^b\right)\end{aligned} \tag{3.71}$$

Where $\boldsymbol{a}_1^b = \begin{bmatrix}\dot{u}_d & \dot{r}_d\end{bmatrix}^T$ is the desired surge and yaw rate accelerations, and $\mathbf{\Upsilon} = \left(\mathbf{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}\right)^{-1}$. The matrix $\mathbf{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}$ is:

$$\mathbf{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B} = \begin{bmatrix}\frac{b_{11}}{m_{11}} & 0\\ 0 & \frac{m_{22}b_{32} - m_{23}b_{22}}{m_{22}m_{33} - m_{23}^2}\end{bmatrix} \tag{3.72}$$

$\mathbf{\Gamma}_1\boldsymbol{M}^{-1}\boldsymbol{B}$ is of full rank if the system is controllable in both surge and yaw [Caharija, 2014]. This is trivially satisfied and therefore $\mathbf{\Upsilon}$ always exists.

Inserting (3.71) into (3.70) gives:

$$\begin{aligned}\dot{\boldsymbol{\nu}}_r &= \mathbf{\Gamma}_1^T\left(\mathbf{\Gamma}_1\boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{a}_1^b - \mathbf{\Gamma}_1\boldsymbol{n}'(\boldsymbol{\nu}_r)\right)\\ &\qquad + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\left(\boldsymbol{M}^{-1}\beta\mathbf{\Upsilon}\left(\mathbf{\Gamma}_1\boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{a}_1^b\right) - \boldsymbol{n}'(\boldsymbol{\nu}_r)\right)\\ &= \mathbf{\Gamma}_1^T\boldsymbol{a}_1^b + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\left(\mathbf{\Upsilon}'\left(\mathbf{\Gamma}_1\boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{a}_1^b\right) - \boldsymbol{n}'(\boldsymbol{\nu}_r)\right)\\ &= \mathbf{\Gamma}_1^T\boldsymbol{a}_1^b + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\left(\mathbf{\Upsilon}'\boldsymbol{a}_1^b + \left(\mathbf{\Upsilon}'\mathbf{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{n}'(\boldsymbol{\nu}_r)\right)\end{aligned} \tag{3.73}$$

Where $\mathbf{\Upsilon}' = \boldsymbol{M}^{-1}\boldsymbol{B}\mathbf{\Upsilon}$.

The dynamics can furthermore be sorted in terms which affects the dynamics from the acceleration input, $\boldsymbol{a}_1^b$, and those who affects the dynamics internally:

$$\dot{\boldsymbol{\nu}}_r = \left(\mathbf{\Gamma}_1^T + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\mathbf{\Upsilon}'\right)\boldsymbol{a}_1^b + \mathbf{\Gamma}_2^T\mathbf{\Gamma}_2\left(\mathbf{\Upsilon}'\mathbf{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{n}'(\boldsymbol{\nu}_r) \tag{3.74}$$

The desired acceleration is selected as:

$$\begin{aligned}\boldsymbol{a}_1^b &= \dot{\boldsymbol{\nu}}_{1r_d} - \boldsymbol{K}_p\left(\boldsymbol{\nu}_{1r} - \boldsymbol{\nu}_{1r_d}\right)\\ &= \dot{\boldsymbol{\nu}}_{1r_d} - \boldsymbol{K}_p\left(\mathbf{\Gamma}_1\boldsymbol{\nu}_r - \boldsymbol{\nu}_{1r_d}\right)\end{aligned} \tag{3.75}$$

Where $\boldsymbol{K}_p = \begin{bmatrix}k_u & 0\\ 0 & k_r\end{bmatrix}$ is a gain matrix, $\boldsymbol{\nu}_{1r} = \begin{bmatrix}u_r & r\end{bmatrix}^T$ and $\boldsymbol{\nu}_{1r_d} = \begin{bmatrix}u_{rd} & r_d\end{bmatrix}^T$.

Inserting the desired acceleration into the dynamics, and defining $\tilde{\boldsymbol{\nu}}_r = \boldsymbol{\nu}_r - \boldsymbol{\Gamma}_1^T \boldsymbol{\nu}_{1r_d}$:

$$\begin{aligned}
\dot{\boldsymbol{\nu}}_r =& \left(\boldsymbol{\Gamma}_1^T + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\right)\left(\dot{\boldsymbol{\nu}}_{1r_d} - \boldsymbol{K}_p\left(\boldsymbol{\Gamma}_1^T\boldsymbol{\nu}_r - \boldsymbol{\nu}_{1r_d}\right)\right) \\
&+ \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{n}'(\boldsymbol{\nu}_r) \\
\boldsymbol{\nu}_r - \boldsymbol{\Gamma}_1^T\boldsymbol{\nu}_{1r_d} =& -\left(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\boldsymbol{K}_p\boldsymbol{\Gamma}_1\right)\boldsymbol{\nu}_r \\
&+ \left(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\boldsymbol{K}_p\right)\boldsymbol{\Gamma}_1\boldsymbol{\Gamma}_1^T\boldsymbol{\nu}_{1r_d} \\
&+ \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}\right) \\
\tilde{\boldsymbol{\nu}}_r =& -\left(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\boldsymbol{K}_p\boldsymbol{\Gamma}_1\right)\tilde{\boldsymbol{\nu}}_r \\
&+ \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{n}'(\boldsymbol{\nu}_r) + \boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}\right)
\end{aligned} \tag{3.76}$$

(3.76) is linear in all terms, except the last. It is important to notice that the last term only influences the sway motion, as it is mapped through $\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2$. Hence, the equation is linear in both surge and yaw rate. In component form (3.76) reads:

$$\begin{aligned}
\dot{\tilde{u}}_r &= -k_{11}\tilde{u}_r \\
\dot{v}_r &= \frac{m_{33} + m_{23}l_x}{m_{23} + m_{22}l_x}\left(k_{22}r - k_{22}r_d - \dot{r}_d - n_3'(\boldsymbol{\nu}_r)\right) - n_2'(\boldsymbol{\nu}_r) \\
\dot{\tilde{r}} &= -k_{22}\tilde{r}_r
\end{aligned} \tag{3.77}$$

Where n_2' and n_3' is the contribution from the Coriolis-centripetal and damping matrices, in sway and yaw respectively. They are given as:

$$\begin{aligned}
n_2'(\boldsymbol{\nu}_r) =& \frac{1}{m_{22}m_{33} - m_{23}^2}\Big(\left(m_{23}N_v - m_{33}Y_v\right)v_r - m_{23}\left(X_{\dot{u}} - Y_{\dot{v}}\right)u_r v_r \\
&+ \left(m_{33}\left(m - X_{\dot{u}}\right) + m_{23}Y_{\dot{r}}\right)u_r r - \left(m_{33}Y_r - m_{23}N_r\right)r\Big) \\
n_3'(\boldsymbol{\nu}_r) =& \frac{1}{m_{22}m_{33} - m_{23}^2}\Big(\left(m_{23}Y_r - m_{22}N_r\right)r \\
&- \left(m_{23}\left(m - X_{\dot{u}}\right) + m_{22}Y_{\dot{r}}\right)u_r r + m_{22}\left(X_{\dot{u}} - Y_{\dot{v}}\right)u_r v_r \\
&+ \left(m_{23}Y_v - m_{22}N_v\right)v_r\Big)
\end{aligned} \tag{3.78}$$

(3.77) is found by evaluating (3.76) with the matrices in (3.2). Specifically $b_{11} = b_{32} = 1$ and $b_{22} = -\frac{1}{l_x}$ where l_x is the distance from the CO to the rudders along the x-axis. (3.78) is found by evaluating (3.65) with the matrices in (3.2).

By approximating the last term of (3.76) using a first order Taylor series, the dynamics of the AUV is linear in all terms:

$$\begin{aligned}\boldsymbol{n}'(\boldsymbol{\nu}_r) &\approx \boldsymbol{n}'(\boldsymbol{\nu}_r^*) + \underbrace{\left.\frac{\partial \boldsymbol{n}'(\boldsymbol{\nu}_r)}{\partial \boldsymbol{\nu}_r}\right|_{\boldsymbol{\nu}_r=\boldsymbol{\nu}_r^*}}_{\boldsymbol{N}'} (\boldsymbol{\nu}_r - \boldsymbol{\nu}_r^*) \\ &= \boldsymbol{n}'(\boldsymbol{\nu}_r^*) + \boldsymbol{N}'\boldsymbol{\nu}_r - \boldsymbol{N}'\boldsymbol{\nu}_r^* \\ &= \boldsymbol{N}'\boldsymbol{\nu}_r + \boldsymbol{b}'(\boldsymbol{\nu}_r^*) \\ &= \boldsymbol{N}'(\tilde{\boldsymbol{\nu}}_r + \boldsymbol{\Gamma}_1^T\boldsymbol{\nu}_{1r_d}) + \boldsymbol{b}'(\boldsymbol{\nu}_r^*)\end{aligned} \tag{3.79}$$

Where $\boldsymbol{\nu}_r^*$ is a linearization point and $\boldsymbol{b}'(\boldsymbol{\nu}_r^*) = \boldsymbol{n}'(\boldsymbol{\nu}_r^*) - \boldsymbol{N}'\boldsymbol{\nu}_r^*$ is a constant term. The linearization points is selected as the current AUV velocity, $\boldsymbol{\nu}_r^* = \boldsymbol{\nu}_r(t_0)$.

Inserting (3.79) into (3.76) results in:

$$\begin{aligned}\dot{\tilde{\boldsymbol{\nu}}}_r = &-\Big(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\boldsymbol{K}_p\boldsymbol{\Gamma}_1\Big)\tilde{\boldsymbol{\nu}}_r \\ &+ \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\Big(\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\Big(\boldsymbol{N}(\tilde{\boldsymbol{\nu}}_r + \boldsymbol{\Gamma}_1^T\boldsymbol{\nu}_{1r_d}) + \boldsymbol{b}(\boldsymbol{\nu}_r^*)\Big) + \boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}\Big) \\ = &-\Big(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{\Upsilon}'\left(\boldsymbol{K}_p\boldsymbol{\Gamma}_1 - \boldsymbol{\Gamma}_1\boldsymbol{N}\right) + \boldsymbol{N}\right)\Big)\tilde{\boldsymbol{\nu}}_r \\ &+ \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{N}\boldsymbol{\Gamma}_1^T\boldsymbol{\nu}_{1r_d} \\ &+ \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{b}(\boldsymbol{\nu}_r^*) + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}\end{aligned} \tag{3.80}$$

(3.80) can be written as:

$$\dot{\tilde{\boldsymbol{\nu}}}_r = \boldsymbol{A}\tilde{\boldsymbol{\nu}}_r + \boldsymbol{\beta}\boldsymbol{\nu}_{1r_d} + \boldsymbol{G} + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d} \tag{3.81}$$

Where:

$$\begin{aligned}\boldsymbol{A} &= -\Big(\boldsymbol{\Gamma}_1^T\boldsymbol{K}_p\boldsymbol{\Gamma}_1 + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{\Upsilon}'\left(\boldsymbol{K}_p\boldsymbol{\Gamma}_1 - \boldsymbol{\Gamma}_1\boldsymbol{N}\right) + \boldsymbol{N}\right)\Big) \\ \boldsymbol{\beta} &= \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{N}\boldsymbol{\Gamma}_1^T \\ \boldsymbol{G} &= \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\left(\boldsymbol{\Upsilon}'\boldsymbol{\Gamma}_1 - \mathbf{I}\right)\boldsymbol{b}(\boldsymbol{\nu}_r^*)\end{aligned} \tag{3.82}$$

This is not a true linear system, due to the constant $\boldsymbol{G}$ and the term $\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}$. Hence $\tilde{\boldsymbol{\nu}}_r = \mathbf{0}$ and $\boldsymbol{\nu}_{1r_d} \equiv \mathbf{0}$ does not imply $\dot{\tilde{\boldsymbol{\nu}}}_r = \mathbf{0}$. However, by considering $\boldsymbol{G}$ and $\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}$ as inputs to the system, (3.81) can be solved as:

$$\begin{aligned}\tilde{\boldsymbol{\nu}}_r(t) &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\tilde{\boldsymbol{\nu}}_r(t_0) + \int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\sigma)}\boldsymbol{\beta}\boldsymbol{\nu}_{1r_d}(\sigma)\mathrm{d}\sigma \\ &\qquad + \int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\sigma)}\Big(\boldsymbol{G} + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}(\sigma)\Big)\mathrm{d}\sigma \\ &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\tilde{\boldsymbol{\nu}}_r(t_0) + \int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\sigma)}\left(\boldsymbol{\beta}\boldsymbol{\nu}_{1r_d}(\sigma) + \boldsymbol{G}\right)\mathrm{d}\sigma \\ &\qquad + \int_{t_0}^{t}\mathrm{e}^{\boldsymbol{A}(t-\sigma)}\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}(\sigma)\mathrm{d}\sigma\end{aligned} \tag{3.83}$$

It is assumed that the initial reference change take place instantly. This is expressed by modeling $\boldsymbol{\nu}_{1r_d}(t)$ using a left continuous Heaviside step function [Wikipedia, 2015]:

$$H(t) = \begin{cases} 0 & t \leq 0 \\ 1 & t > 0 \end{cases} \tag{3.84}$$

Denoting the previous reference as $\boldsymbol{\nu}_{1r_d}^{-}$ and the change in reference as $\Delta\boldsymbol{\nu}_{1r_d}$, the reference signal can be written as:

$$\boldsymbol{\nu}_{1r_d}(t) = \boldsymbol{\nu}_{1r_d}^{-} + H(t-t_0)\Delta\boldsymbol{\nu}_{1r_d} \tag{3.85}$$

Note that $\boldsymbol{\nu}_{1r_d}(t)$ is constant for $t > t_0$, and that the derivative is:

$$\dot{\boldsymbol{\nu}}_{1r_d}(t) = \delta(t-t_0)\Delta\boldsymbol{\nu}_{1r_d} \tag{3.86}$$

Where $\delta(x)$ is the impulse function.

By using (3.86), the last term in (3.83) can be written as:

$$\begin{aligned} \int_{t_0}^{t} \mathrm{e}^{\boldsymbol{A}(t-\sigma)}\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\dot{\boldsymbol{\nu}}_{1r_d}(\sigma) &= \int_{t_0}^{t} \mathrm{e}^{\boldsymbol{A}(t-\sigma)}\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\delta(\sigma-t_0)\Delta\boldsymbol{\nu}_{1r_d}\mathrm{d}\sigma \\ &= \int_{t_0}^{t} \mathrm{e}^{\boldsymbol{A}(t-\sigma)}\delta(\sigma-t_0)\mathrm{d}\sigma\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\Delta\boldsymbol{\nu}_{1r_d} \\ &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\Delta\boldsymbol{\nu}_{1r_d} \end{aligned} \tag{3.87}$$

Remark 3.5. (3.87) *models an impulse to the system, which is intuitive. When the reference changes as a a step, the controllers will excite the actuators infinitely due to the feed forward of the desired acceleration (which is infinite for a step in the reference). For a practical system, this impulse would be dampened by the natural low pass filtering of the actuators, and the accuracy of the predictions would most likely be more correct by neglecting the term* (3.87).

Further denoting and inserting $\boldsymbol{\nu}_{1r_d}(\sigma) = \boldsymbol{\nu}_{1r_d}, \sigma > t_0$ and (3.87) into (3.83):

$$\begin{aligned} \tilde{\boldsymbol{\nu}}_r(t) &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\tilde{\boldsymbol{\nu}}_r(t_0) + \int_{t_0}^{t} \mathrm{e}^{\boldsymbol{A}(t-\sigma)}\left(\boldsymbol{\beta}\boldsymbol{\nu}_{1r_d} + \boldsymbol{G}\right)\mathrm{d}\sigma \\ &\qquad + \mathrm{e}^{\boldsymbol{A}(t-t_0)}\boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\Delta\boldsymbol{\nu}_{1r_d} \\ &= \mathrm{e}^{\boldsymbol{A}(t-t_0)}\left(\tilde{\boldsymbol{\nu}}_r(t_0) + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\Delta\boldsymbol{\nu}_{1r_d}\right) + \int_{t_0}^{t} \mathrm{e}^{\boldsymbol{A}(t-\sigma)}\left(\boldsymbol{\beta}\boldsymbol{\nu}_{1r_d} + \boldsymbol{G}\right)\mathrm{d}\sigma \end{aligned} \tag{3.88}$$

For simplicity t_0 is selected as $t_0 = 0$ s. (3.88) can be solved as [Hespanha, 2009]:

$$\tilde{\boldsymbol{\nu}}_r(t) = \mathrm{e}^{\boldsymbol{A}t}\left(\tilde{\boldsymbol{\nu}}_r(0) + \boldsymbol{\Gamma}_2^T\boldsymbol{\Gamma}_2\boldsymbol{\Upsilon}'\Delta\boldsymbol{\nu}_{1r_d}\right) - \boldsymbol{A}^{-1}\left(\boldsymbol{I} - \mathrm{e}^{\boldsymbol{A}t}\right)\left(\boldsymbol{\beta}\boldsymbol{\nu}_{1r_d} + \boldsymbol{G}\right) \tag{3.89}$$

As remark 3.5 suggests, the impulse generated by the initial reference change should be neglected due to the low pass nature of the actuators. (3.89) is then reduced to:

$$\tilde{\boldsymbol{\nu}}_r(t) = \mathrm{e}^{\boldsymbol{A}t}\tilde{\boldsymbol{\nu}}_r(0) - \boldsymbol{A}^{-1}\left(\boldsymbol{I} - \mathrm{e}^{\boldsymbol{A}t}\right)\left(\boldsymbol{\beta}\boldsymbol{\nu}_{1r_d} + \boldsymbol{G}\right) \tag{3.90}$$

The solution to the dynamics can then be used to compute the AUV position for a selected velocity pair. The system is simulated in discrete time using a standard simulation algorithm. From (3.67), the kinematics are modeled as:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\boldsymbol{\eta})\boldsymbol{\nu}_r + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \tag{3.91}$$

The rotation matrix $\boldsymbol{R}(\boldsymbol{\eta})$ is given as:

$$\boldsymbol{R}(\boldsymbol{\eta}) = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.92}$$

The modified Euler method is used for simulating the position. For a system $\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y}, t)$, the method is given as: [Egeland and Gravdahl, 2003]

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}\left(\boldsymbol{y}_n, t_n\right) \\ \boldsymbol{k}_2 &= \boldsymbol{f}\left(\boldsymbol{y}_n + \frac{h}{2}\boldsymbol{k}_1, t_n + \frac{h}{2}\right) \\ \boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + h\boldsymbol{k}_2 \end{aligned} \tag{3.93}$$

Where h is the discretization time step.

Hence, the system $\dot{\boldsymbol{\eta}} = \boldsymbol{f}(\boldsymbol{\eta}, t) = \boldsymbol{R}(\boldsymbol{\eta})\boldsymbol{\nu}_r(t) + \begin{bmatrix} \boldsymbol{V}_c & 0 \end{bmatrix}^T$ can be simulated as:

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{R}(\boldsymbol{\eta}(t_n))\boldsymbol{\nu}_r(t_n) \\ \boldsymbol{k}_2 &= \boldsymbol{R}(\boldsymbol{\eta}(t_n) + \frac{h}{2}\boldsymbol{k}_1)\boldsymbol{\nu}_r(t_n + \frac{h}{2}) \\ \boldsymbol{\eta}(t_{n+1}) &= \boldsymbol{\eta}(t_n) + h\left(\boldsymbol{k}_2 + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix}\right) \end{aligned} \tag{3.94}$$

Chapter 4

Simulator development

4.1 Simulator overview

The simulator is implemented in SIMULINK [Mathworks, 2014b] using MATLAB [Mathworks, 2014a] function blocks. This makes the simulator structure easy to understand, due to the block-based SIMULINK language, while still employing all the functionality and flexibility of the MATLAB language. The simulator is based on two different HUGIN simulators, one provided by FFI and one by [Engelhardtsen, 2007].

An overview of the simulator is shown in figure 4.1. The different parts of the simulator implementation is described in the following sections. In particular, the sonar model is covered in section 4.3 and the horizontal subsystem in sections 4.4, 4.5 and 4.6.

The simulator code is not included in the thesis due to confidentiality issues.

4.2 AUV model

The simulator is based on the 6DOF mathematical model of the HUGIN 1000 AUV given in section 2.1. The model is, however, transformed to the pivot point to simplify the interaction with the Dynamic window algorithm. The transformation is similar to the one defined in section 3.1, but needs to be defined in 6DOF instead of 3DOF. For a model in 6DOF, the transformation is defined as [Caharija, 2014, Børhaug et al., 2007]:

$$u_r = u_r, \quad \bar{v}_r = v_r + \epsilon_1 r, \quad \bar{w}_r = w_r + \epsilon_2 q, \quad p = p, \quad q = q, \quad r = r \tag{4.1}$$

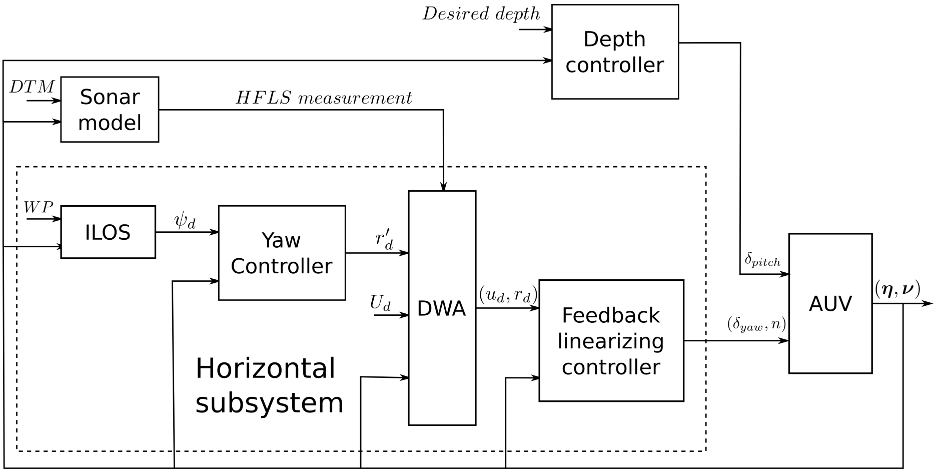


Figure 4.1: Simulator overview

Where ϵ_1 and ϵ_2 is given as:

$$\epsilon_1 \triangleq \frac{m_{66}Y_{\delta u^2} + m_{26}Y_{\delta u^2}l_x}{m_{22}Y_{\delta u^2}l_x + m_{26}Y_{\delta u^2}}, \quad \epsilon_2 \triangleq -\frac{m_{55}Z_{\delta u^2} - m_{35}Z_{\delta u^2}l_x}{m_{33}Z_{\delta u^2}l_x - m_{35}Z_{\delta u^2}} \tag{4.2}$$

It is easy to show that for a slender body AUV with equal pitch and yaw rudders, such as the HUGIN 1000 AUV, $\epsilon_1 = -\epsilon_2 = \epsilon$ [Caharija et al., 2012]. Hence, the new relative velocity vector $\bar{\boldsymbol{\nu}}_r$ can be written as:

$$\bar{\boldsymbol{\nu}}_r = \begin{bmatrix} u_r \\ \bar{v}_r \\ \bar{w}_r \\ p \\ q \\ r \end{bmatrix} \triangleq \boldsymbol{H}^{-1}\boldsymbol{\nu}_r \tag{4.3}$$

Where the transformation matrix is given as:

$$\boldsymbol{H} \triangleq \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -\epsilon \\ 0 & 0 & 1 & \epsilon & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

The transformation corresponds to a translation of the CO along the $\{b\}$ x-axis, of a distance ϵ. As in section 3.1, without loss of generality, the model is transformed to describe the motion of a point located at $\boldsymbol{p}^b = \begin{bmatrix} \epsilon & 0 & 0 \end{bmatrix}^T$. The point $\boldsymbol{p}$ is given in $\{n\}$ as:

$$\bar{N} = N + \epsilon\cos(\psi)\cos(\theta), \quad \bar{E} = E + \epsilon\sin(\psi)\cos(\theta), \quad \bar{D} = D - \epsilon\sin(\theta) \tag{4.5}$$

The new position vector $\bar{\boldsymbol{\eta}}$ is given as:

$$\bar{\boldsymbol{\eta}} = \begin{bmatrix} \bar{N} \\ \bar{E} \\ \bar{D} \\ \phi \\ \theta \\ \psi \end{bmatrix} \tag{4.6}$$

The transformed system in 6DOF is given as:

$$\dot{\bar{\boldsymbol{\eta}}} = \boldsymbol{J}(\bar{\boldsymbol{\eta}})\bar{\boldsymbol{\nu}}_r + \begin{bmatrix} \boldsymbol{V}_c \\ \boldsymbol{0}_{3\times1} \end{bmatrix} \tag{4.7a}$$

$$\bar{\boldsymbol{M}}\dot{\bar{\boldsymbol{\nu}}}_r + \bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r + \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r + \bar{\boldsymbol{g}}(\bar{\boldsymbol{\eta}}) = \boldsymbol{H}^T\left(\boldsymbol{\tau}_{propeller} + \boldsymbol{\tau}_{rudder}\right) \tag{4.7b}$$

Where $\bar{\boldsymbol{M}} = \boldsymbol{H}^T\boldsymbol{M}\boldsymbol{H}$, $\bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r) = \boldsymbol{H}^T\boldsymbol{D}(\bar{\boldsymbol{\nu}}_r)\boldsymbol{H}$ and $\bar{\boldsymbol{g}}(\bar{\boldsymbol{\eta}}) = \boldsymbol{H}^T\boldsymbol{g}(\bar{\boldsymbol{\eta}})$. The Coriolis-centripetal matrix is not transformed, but rather parameterized using $\bar{\boldsymbol{M}}$.

4.3 Sonar modeling

A sonar (Sound Navigation and Ranging) is a device which, by the use of sound waves, measure range to objects in water [National Ocean Service, 2014]. The sonar modeling is based on the work of [Engelhardtsen, 2007].

4.3.1 Sonar configuration

The HUGIN 1000 sonar sensor suite includes the following sonars [Eriksen, 2014, Engelhardtsen, 2007]:

- **Vertical sonar (VS).** Single sonar beam directed straight down from the AUV.
- **Forward tilted sonar (FTS).** Single sonar beam tilted forward.
- **Side scan sonar (SSS).** Sonar pointing to the side of the AUV, one on the starboard side and one on the port side. This sonar has a wide sector, covering 70° in the vertical plane, and 0.5° in the horizontal plane.
- **Horizontal forward looking sonar (HFLS).** An array consisting of 120 sonar beams, pointed straight forward. Each beam covers a sector of 20° in the vertical plane, and 1° in the horizontal plane. The entire array covers a section of 120° in the horizontal plane.

Table 4.1 Position and orientation of sonar sensors

Sonar	**Position in $\{b\}$**	**Orientation in $\{b\}$**
VS	$\begin{bmatrix}-0.500 & -0.080 & 0.215\end{bmatrix}$	$\begin{bmatrix}0 & -90^\circ & 0\end{bmatrix}$
FTS	$\begin{bmatrix}-0.492 & 0.080 & 0.218\end{bmatrix}$	$\begin{bmatrix}0 & -45^\circ & 0\end{bmatrix}$
Starboard SSS	$\begin{bmatrix}-2.212 & 0.282 & 0.232\end{bmatrix}$	$\begin{bmatrix}0 & (-80^\circ \rightarrow -10^\circ) & 90^\circ\end{bmatrix}$
Port SSS	$\begin{bmatrix}-2.212 & -0.282 & 0.232\end{bmatrix}$	$\begin{bmatrix}0 & (-80^\circ \rightarrow -10^\circ) & -90^\circ\end{bmatrix}$
HFLS	$\begin{bmatrix}-0.085 & 0.000 & 0.050\end{bmatrix}$	$\begin{bmatrix}0 & (-10^\circ \rightarrow 10^\circ) & -60^\circ : 1^\circ : 60^\circ\end{bmatrix}$

Only the HFLS sonar is used in the simulator, but the other sonars are also included in the simulator. For completeness, the entire sonar model is covered in this section.

The maximum detection range of the sonars is defined as 50 m. The orientation and position of the sonars are listed in Table 4.1. Figure 4.2 show an illustration of the sonar sensors.

4.3.2 Seabed modeling

The environment of the AUV is modeled using a Digital terrain model (DTM). A DTM contain height data of a terrain, which creates a 3-dimensional model of the environment [Statens Kartverk, 2014]. It is usual to use matrices for storing the DTM. This is done by discretizing the base plane as rows and columns, and storing the height in the correct matrix entry. For the $\{n\}$ frame, the North-axis is used to identify the row number, while the East-axis is used to identify the column number. The Down-axis is stored in the corresponding matrix element. If surface ice is to be included, a separate DTM matrix can be used for modeling the ice depths. However, since all obstacles are assumed to be tall with vertical faces, surface ice is not included in the simulator.

Figure 4.3 show an example DTM matrix, for the $\{n\}$ frame. A visualization of a DTM matrix of the seabed in Jesusbukta in the Breiangen area is shown in Figure 4.4.

4.3.3 Single beam sonars

The VS and FTS are single beam sonar sensors. A sonar measurement is found by searching for an intersection between the sonar beam and an obstacle, up to the maximum sonar range. The search is done by discretizing the sonar beam with a desired resolution, and comparing the height of each point with the DTM matrix for the seabed profile.

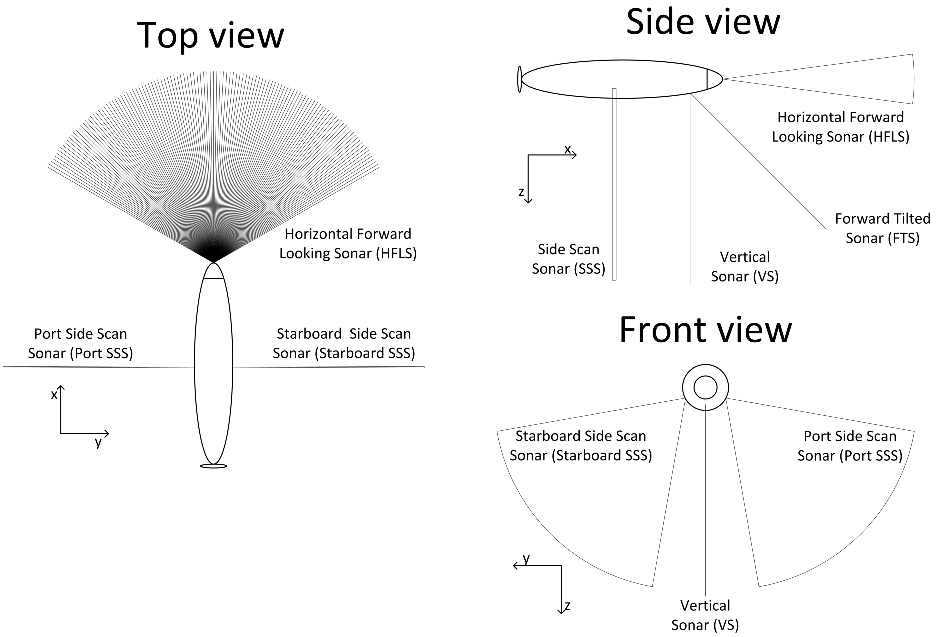
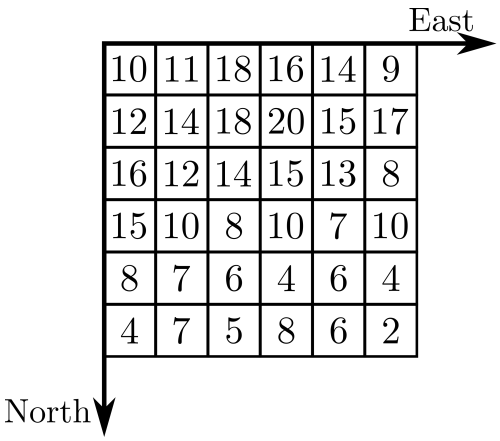

Top view
Horizontal Forward Looking Sonar (HFLS)
Port Side Scan Sonar (Port SSS)
Starboard Side Scan Sonar (Starboard SSS)
x
y
Side view
Horizontal Forward Looking Sonar (HFLS)
x
z
Forward Tilted Sonar (FTS)
Side Scan Sonar (SSS)
Vertical Sonar (VS)
Front view
Starboard Side Scan Sonar (Starboard SSS)
Port Side Scan Sonar (Port SSS)
y
z
Vertical Sonar (VS)

Figure 4.2: HUGIN 1000 sonar sensors



10	11	18	16	14	9
12	14	18	20	15	17
16	12	14	15	13	8
15	10	8	10	7	10
8	7	6	4	6	4
4	7	5	8	6	2

Figure 4.3: Example DTM matrix

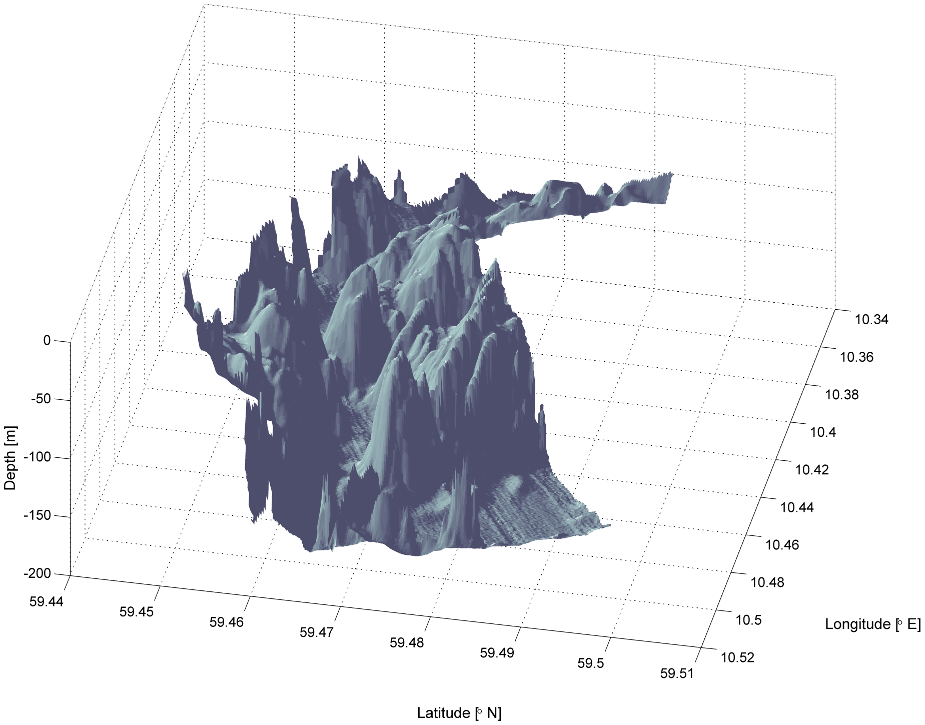


Figure 4.4: Seabed map of Jesusbukta in the Breiangen area

To describe the search, the SONAR frame, $\{s\}$, is introduced. This frame has its origin in the sonar position, and is rotated with respect to $\{b\}$ such that the x-axis is pointing normal to the sonar plane. To fully define the frame, the rotation about the x-axis is zero. For the SSS and HFLS, the orientation of $\{s\}$ is defined such that the x-axis points in the center of the sonar sector.

The search is done by iterating along the sonar beam. In $\{s\}$, each iteration point is given as $\boldsymbol{p}^s_{i/s} = \begin{bmatrix} m_j & 0 & 0 \end{bmatrix}$, where m_j is the iteration variable. m_j is initialized as 0, and incremented with the desired resolution, δ_m, until an intersection with an obstacle is found, or the maximum sonar range is reached. In order to check for an intersection, the iteration point must be transformed to $\{n\}$. The iteration point height is then compared with the height of the sea terrain. If the iteration point is below the sea terrain, an intersection is found.

The transformation from $\{s\}$ to $\{n\}$ is done by first transforming from $\{s\}$ to $\{b\}$, and then from $\{b\}$ to $\{n\}$. The transformations includes both rotation and translation. The transformation from $\{s\}$ to $\{b\}$ is shown in (4.8), where $\boldsymbol{p}^b_{s/b}$ is the position of the sonar and $\boldsymbol{R}^b_s$ is the rotation matrix from $\{b\}$ to $\{s\}$.

$$\boldsymbol{p}^b_{i/b} = \boldsymbol{R}^b_s \boldsymbol{p}^s_{i/s} + \boldsymbol{p}^b_{s/b} \tag{4.8}$$

The transformation can be written in matrix form as:

$$\begin{bmatrix} \boldsymbol{p}^b_{i/b} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^b_s & \boldsymbol{p}^b_{s/b} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}^s_{i/s} \\ 1 \end{bmatrix} \tag{4.9}$$

This is a homogeneous transformation, which combines rotation and transformation in one operation. It can be written as in equation (4.10), where $\boldsymbol{A}^b_s$ is the homogeneous transformation matrix from $\{s\}$ to $\{b\}$.

$$\begin{gathered} \boldsymbol{P}^b_i = \boldsymbol{A}^b_s \boldsymbol{P}^s_i \\ \boldsymbol{P}^b_i = \begin{bmatrix} \boldsymbol{p}^b_{i/b} \\ 1 \end{bmatrix}, \quad \boldsymbol{P}^s_i = \begin{bmatrix} \boldsymbol{P}^s_i \\ 1 \end{bmatrix}, \quad \boldsymbol{A}^b_s = \begin{bmatrix} \boldsymbol{R}^b_s & \boldsymbol{p}^b_{s/b} \\ 0 & 1 \end{bmatrix} \end{gathered} \tag{4.10}$$

Furthermore, the transformation to $\{n\}$ can simply be done by using the homogeneous transformation again:

$$\begin{gathered} \boldsymbol{P}^n_i = \boldsymbol{A}^n_b \boldsymbol{P}^b_i = \boldsymbol{A}^n_b \boldsymbol{A}^b_s \boldsymbol{P}^s_i = \boldsymbol{A}^n_s \boldsymbol{P}^s_i \\ \boldsymbol{P}^n_i = \begin{bmatrix} \boldsymbol{p}^n_{i/n} \\ 1 \end{bmatrix}, \quad \boldsymbol{A}^n_s = \boldsymbol{A}^n_b \boldsymbol{A}^b_s, \quad \boldsymbol{A}^n_b = \begin{bmatrix} \boldsymbol{R}^n_b & \boldsymbol{p}^n_{b/n} \\ 0 & 1 \end{bmatrix} \end{gathered} \tag{4.11}$$

Where $\boldsymbol{R}^n_b$ is the rotation matrix from $\{n\}$ to $\{b\}$ and $\boldsymbol{p}^n_{b/n}$ is the position of the AUV, given in $\{n\}$.

Algorithm 1 show a pseudo code for simulating a single beam sonar.

Algorithm 1 Single beam sonar simulator pseudo code

```
m ← 0
while $r \leq$ max sonar range do
    $\boldsymbol{P}_i^s = \begin{bmatrix} m & 0 & 0 & 1 \end{bmatrix}^T$
    $\boldsymbol{P}_i^n = \boldsymbol{A}_s^n \boldsymbol{P}_i^s$
    Find correct section in the DTM matrices using $x$ and $y$ coordinates of $\boldsymbol{P}_i^n$
    if z coordinate of $\boldsymbol{P}_i^n$ below sea terrain then
        break
    else
        $m = m + \delta_m$
    end if
end while
if $m >$ max range then
    return sonar out of range
else
    return $m$
end if
```

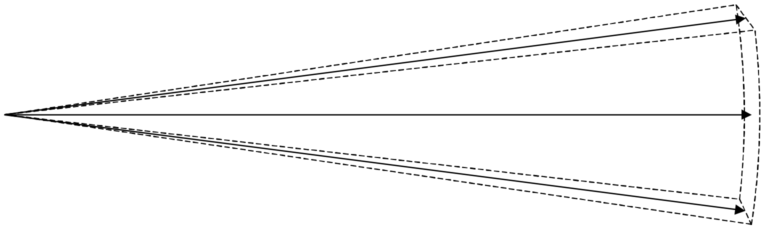


Figure 4.5: Search lines of a single HFLS beam

4.3.4 Horizontal forward looking sonar

The HFLS is an array of 120 beams, each covering a sector of 20° in the vertical plane, and 1° in the horizontal plane. To model the sonar, each beam is treated as a separate sonar. In order to model a beam, the SONAR BEAM, $\{sb\}$ frame is introduced. The origin of the frame is the same as $\{s\}$, and is rotated such that the x-axis points toward the center of the beam sector. Again, the rotation about the x-axis is zero to fully define the frame. Furthermore, each of the 120 sonar beams covers a sector of 20° of the vertical plane. To include this in the sonar model, each beam is modeled by three lines inside the beam sector. The shortest range is then used as the measurement. Figure 4.5 show a single HFLS beam, and the three search lines. To describe the search, the SONAR LINE frame, $\{sl\}$ is used. The origin of this frame is $\{sb\}$, and it is rotated with respect to $\{sb\}$ such that the x-axis aligns with the search line. The rotation about the x-axis is zero.

To find the iteration point in $\{n\}$ coordinates, two additional transformations must be applied. The iteration point, $\boldsymbol{p}^{sl}_{i/sl}$, is now given in $\{sl\}$. This is transformed to $\{n\}$ coordinates using the same method, but with a different homogeneous transformation matrix. The transformation is given as:

$$\boldsymbol{P}^n_i = \boldsymbol{A}^n_{sl}\boldsymbol{P}^{sl}_i$$
$$\boldsymbol{P}^n_i = \begin{bmatrix}\boldsymbol{p}^n_{i/n}\\ 1\end{bmatrix}, \quad \boldsymbol{P}^{sl}_i = \begin{bmatrix}\boldsymbol{p}^{sl}_{i/sl}\\ 1\end{bmatrix}, \quad \boldsymbol{A}^n_{sl} = \boldsymbol{A}^n_b\boldsymbol{A}^b_s\boldsymbol{A}^s_{sb}\boldsymbol{A}^{sb}_{sl} \tag{4.12}$$

The same approach as for the single beam sonars, as described in section 4.3.3, can then be applied for each of the search lines.

Algorithm 2 show a pseudo code for simulating the HFLS.

Algorithm 2 HFLS simulator pseudo code

```
for $j$ = each of the 120 HFLS beams do
    for $k$ = each of the three search lines do
        $m_k \leftarrow 0$
        while $m_k \leq$ max sonar range do
            $\boldsymbol{P}^{sl}_i = \begin{bmatrix}m_k & 0 & 0 & 1\end{bmatrix}^T$
            $\boldsymbol{P}^n_i = \boldsymbol{A}^n_{sl}\boldsymbol{P}^{sl}_i$
            Find correct section in the DTM matrices using $x$ and $y$ coordinates of $\boldsymbol{P}^n_i$
            if z coordinate of $\boldsymbol{P}^n_i$ below sea terrain then
                break
            else
                $m_k = m_k + \delta_m$
            end if
        end while
    end for
    $l_j = \min_k \boldsymbol{m}$
    if $l_j >$ max range then
        $l_j$ = out of range
    end if
end for
return HFLS measurement $\boldsymbol{l}$
```

4.3.5 Side scan sonars

The Port and Starboard SSS are identical, so they are modeled in the same way. The SSS covers a sector of 0.5° in the horizontal plane, and 70° in the vertical plane. The same approach, with splitting the sector into multiple single lines, as for the HFLS is used. 10 lines are used to cover the sector. The transformation in

(4.12) is also used for the SSS. It is worth noticing that the $\{s\}$ and $\{sb\}$ frames are identical for the SSS, hence $A_{sb}^{s} = \mathbb{I}$.

Algorithm 3 show a pseudo code for simulating a SSS.

Algorithm 3 SSS simulator pseudo code

```
for $j$ = each of the ten search lines do
    $m_j \leftarrow 0$
    while $m_j \leq$ max sonar range do
        $\boldsymbol{P}_i^{sl} = \begin{bmatrix} m_j & 0 & 0 & 1 \end{bmatrix}^T$
        $\boldsymbol{P}_i^{n} = \boldsymbol{A}_{sl}^{n} \boldsymbol{P}_i^{sl}$
        Find correct section in the DTM matrices using $x$ and $y$ coordinates of $\boldsymbol{P}_i^{n}$
        if z coordinate of $\boldsymbol{P}_i^{n}$ below sea terrain then
            break
        else
            $m_j = m_j + \delta_m$
        end if
    end while
end for
$l = \min_j \boldsymbol{m}$
if $l >$ max range then
    return SSS out of range
else
    return SSS measurement $l$
end if
```

4.4 Horizontal guidance system

The horizontal guidance system should make the AUV converge to a Line of sight (LOS) vector. To cope with the constant irrotational ocean current, an integral line-of-sight (ILOS) guidance law is used. The ILOS heading is defined as [Børhaug et al., 2008, Caharija, 2014, Fossen, 2011]:

$$\begin{aligned} \psi_{ILOS} &= \alpha - \arctan\left(\frac{e + \sigma e_{int}}{\Delta}\right) \\ \dot{e}_{int} &= \frac{\Delta e}{(e + \sigma e_{int})^2 + \Delta^2} \end{aligned} \tag{4.13}$$

Where α is the path heading, e is the cross track error and e_{int} is an integral state. The tuning constants Δ and σ is the look-ahead-distance and an integral gain respectively. Δ and σ are both given in meters. The variables are illustrated in figure 4.6.

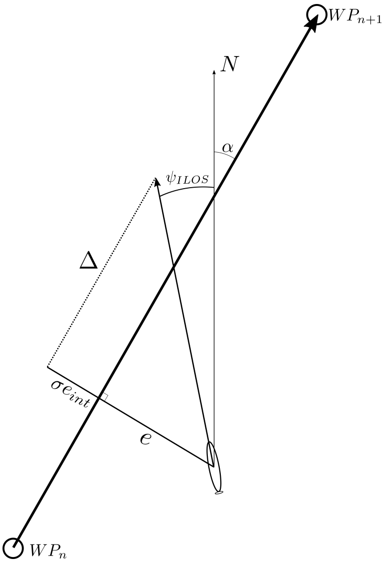


Figure 4.6: ILOS guidance law

The integral state e_{int} allow the heading to be different from the path heading α when the cross track error e is zero. Hence, the AUV can compensate for the ocean current by holding an off-path heading. This is illustrated in figure 4.7. $\dot{e}_{int}$ is forced to zero when the collision avoidance system is active to prevent integrator wind-up.

4.5 Feedback controllers

The simulator contains several controllers:

- Feedback linearizing controller for surge speed and yaw rate
- Yaw controller
- Depth controller

4.5.1 Surge speed and yaw rate controller

Due to the nonholonomic nature of the AUV, only a part of the system can be controlled. Hence, a partial feedback linearizing controller is used for controlling the surge speed and the yaw rate. The controller is derived in section 3.4. From

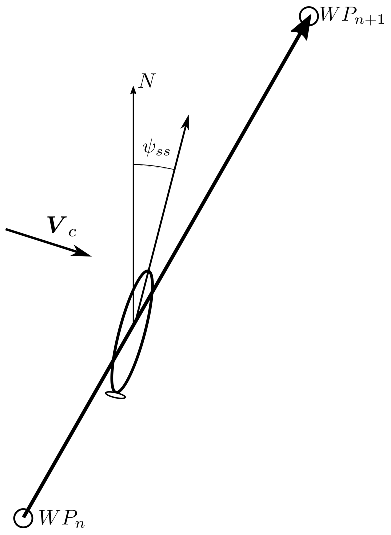


Figure 4.7: Steady state ILOS heading

(3.38) and (3.41) the feedback linearizing controller is formulated as:

$$\begin{aligned}\boldsymbol{f} &= \left(\boldsymbol{\Gamma}_1 \bar{\boldsymbol{M}}^{-1} \bar{\boldsymbol{B}}\right)^{-1} \left(\boldsymbol{\Gamma}_1 \bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r) + \boldsymbol{a}^b\right) \\ &= \left(\boldsymbol{\Gamma}_1 \bar{\boldsymbol{M}}^{-1} \bar{\boldsymbol{B}}\right)^{-1} \left(\boldsymbol{\Gamma}_1 \bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r) + \dot{\bar{\boldsymbol{\nu}}}_{1r_d} - \boldsymbol{K}_p \left(\boldsymbol{\Gamma}_1 \bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{\nu}}_{1r_d}\right)\right)\end{aligned} \tag{4.14}$$

The required propeller speed and rudder deflection angle are computed from $\boldsymbol{f} = \begin{bmatrix} X & N \end{bmatrix}^T$ in (4.14). From section 2.1.3 the surge thrust and yaw torque are given as:

$$X = T_{nn}|n_p|n_p + T_{un}u_r n_p \tag{4.15a}$$

$$N = -Y_{\delta u^2} u_r^2 l_x \delta_{yaw} \tag{4.15b}$$

Solving (4.15a) for the propeller speed n_p gives:

$$n_p = \begin{cases} \frac{-T_{un}u_r + \sqrt{(T_{un}u_r)^2 + 4T_{nn}X}}{2T_{nn}} & : X \geq 0 \\ -\frac{-T_{un}u_r + \sqrt{(T_{un}u_r)^2 - 4T_{nn}X}}{2T_{nn}} & : X < 0 \end{cases} \tag{4.16}$$

Solving (4.15b) for δ_{yaw} gives:

$$\delta_{yaw} = -\frac{N}{Y_{\delta u^2} u_r^2 l_x} \tag{4.17}$$

To avoid division by zero, (4.17) is slightly modified:

$$\delta_{yaw} = -\frac{N}{Y_{\delta u^2} (u_r')^2 l_x}, \quad u_r' = max\left(u_r, \mu\right) \tag{4.18}$$

Where $\mu > 0$ is an arbitrary constant, avoiding division by zero.

It is often useful to analyze the system equations in component form, as in (3.12). To do this, (4.14) can be written as:

$$\boldsymbol{\Gamma}_1 \bar{\boldsymbol{M}}^{-1} \bar{\boldsymbol{B}} \boldsymbol{f} = \boldsymbol{\Gamma}_1 \bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r) + \dot{\bar{\boldsymbol{\nu}}}_{1r_d} - \boldsymbol{K}_p \left(\boldsymbol{\Gamma}_1 \bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{\nu}}_{1r_d} \right) \tag{4.19}$$

By using the transformation (3.11), this can be written as:

$$\boldsymbol{\Gamma}_1 \begin{bmatrix} \tau_u \\ 0 \\ \tau_r \end{bmatrix} = \boldsymbol{\Gamma}_1 \bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r) + \dot{\bar{\boldsymbol{\nu}}}_{1r_d} - \boldsymbol{K}_p \left(\boldsymbol{\Gamma}_1 \bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{\nu}}_{1r_d} \right) \tag{4.20}$$

Inserting the expressions for $\boldsymbol{\Gamma}_1$, $\bar{\boldsymbol{n}}(\bar{\boldsymbol{\nu}}_r)$, $\boldsymbol{K}_p$, $\bar{\boldsymbol{\nu}}_r$ and $\bar{\boldsymbol{\nu}}_{1r_d}$ into (4.20), τ_u and τ_r can be written as:

$$\begin{aligned} \tau_u &= \bar{n}_1(\bar{\boldsymbol{\nu}}_r) - k_u(u_r - u_{rd}) + \dot{u}_{rd} \\ \tau_r &= \bar{n}_3(\bar{\boldsymbol{\nu}}_r) - k_r(r - r_d) + \dot{r}_d \end{aligned} \tag{4.21}$$

Where $k_u, k_r > 0$ since $\boldsymbol{K}_p > 0$. By inserting (4.21) into (3.12), the error dynamics can be written as:

$$\begin{aligned} \dot{\tilde{u}}_r &= -k_u \tilde{u}_r \\ \dot{\tilde{r}} &= -k_r \tilde{r} \end{aligned} \tag{4.22}$$

From (4.22), it is clear that the response should be linear with time constants of $\frac{1}{k_u}$ and $\frac{1}{k_r}$. However, the actuator dynamics are not included in the controller, and introduces a nonlinearity to the closed loop system. This is verified from the step responses in figure 4.8 and 4.9, where the gains are selected as $k_u = k_r = 1\ \text{s}^{-1}$. The peak in the desired propeller speed is due to the term $\dot{u}_{rd}$ in (4.21).

4.5.2 Yaw controller

The yaw controller is selected as a simple proportional controller with a derivative feed forward:

$$r'_d = -k_\psi \left(\psi - \psi_d \right) + \dot{\psi}_d \tag{4.23}$$

Where $k_\psi > 0$ is a gain constant and $\psi_d \triangleq \psi_{ILOS}$. By neglecting the Dynamic window algorithm and selecting $r_d = r'_d$, the closed loop yaw dynamics can be written:

$$\begin{aligned} \dot{\tilde{\psi}} &= \dot{\psi} - \dot{\psi}_d \\ &= r - r'_d - k_\psi(\psi - \psi_d) \\ &= \tilde{r} - k_\psi \tilde{\psi} \end{aligned} \tag{4.24}$$

A step response of the closed loop yaw dynamics is shown in figure 4.10. k_ψ was selected as $k_\psi = 0.5\ \text{s}^{-1}$. It is worth noting that the rudder saturation and rate limitation affects the response. The actuator dynamics are accounted for by limiting the available steering commands through the DWA search space, as described in section 3.3.

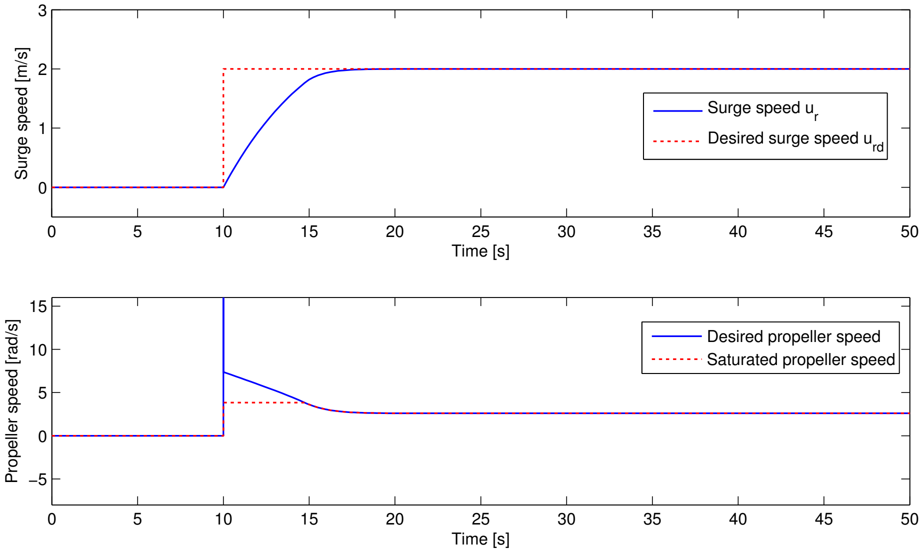


Figure 4.8: Step response of closed loop surge dynamics

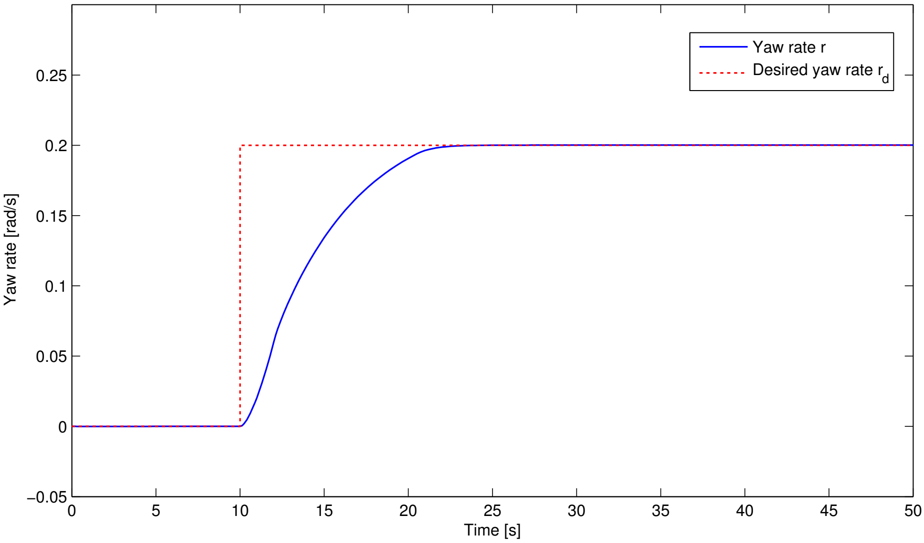


Figure 4.9: Step response of closed loop yaw rate dynamics

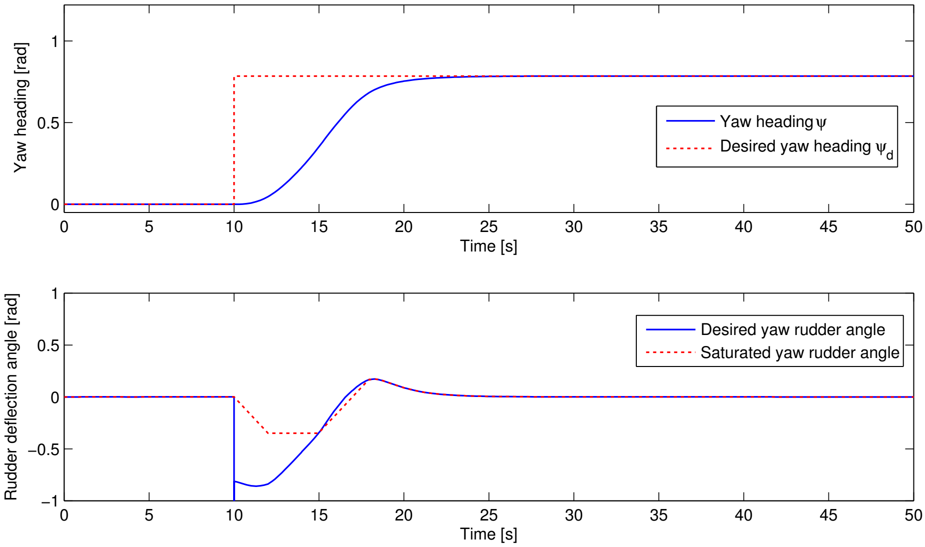


Figure 4.10: Step response of closed loop yaw dynamics

4.5.3 Depth controller

The depth controller is reused from the simulator provided by FFI. It consists of a proportional controller for the depth in cascade with a proportional-derivative controller for the pitch angle. The controller is not considered important for the scope of the thesis, and is hence not described in further detail. The functionality of the controller is verified through a step response, shown in figure 4.11. The surge speed was kept constant at $u_r = 2\ \mathrm{m\,s^{-1}}$. The performance is considered satisfactory, as the controller succeeds in stabilizing the depth of the AUV. The rudder control signal δ_{pitch} seen in figure 4.1 is defined as $\delta_{pitch} = \delta_{port} + \delta_{starboard}$, where $\delta_{port} = \delta_{starboard}$ is required.

4.6 Dynamic window algorithm

The Dynamic window algorithm is implemented based on the theory in section 2.3.1 and the modifications in chapter 3. The different parts of the algorithm is described in the following sections.

4.6.1 Environment representation

As described in section 3.2 a map of the local environment is created to make the AUV circumnavigate obstacles with a desired minimum clearance.

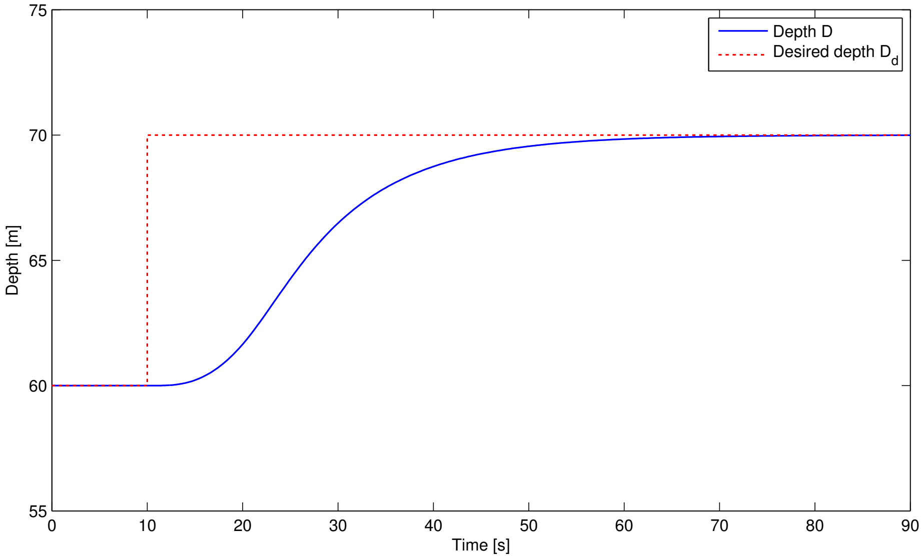


Figure 4.11: Step response of closed loop depth dynamics

The *antitarget* region, $\mathcal{T}$, is found by generating a boundary by interpolating between all the HFLS sonar beams. The boundary is then padded with circles of radius r^*. By interpolating between all sonar points, regardless whether the sonar beam returns a measurement or reaches the maximum range, possible obstacles just outside of the sonar range is taken into account. The interpolated boundary and the corresponding part of $\mathcal{T}$ is illustrated in red in figure 4.12. In addition to this area, the area outside of the sonar coverage is included in $\mathcal{T}$ since no information about this area is available.

The *avoid* region, Ω, is found by estimating the obstacle boundary by interpolating between the sonar beams which return a measurement. The estimated obstacle boundary is then padded with circles of radius $\bar{r}$. Ω is illustrated in figure 4.12 as the blue area.

4.6.2 Search space and predicted trajectories

The search space is found in four steps:

1. Compute the set of *Dynamically feasible velocities*, V_f, and discretize the set into a list of velocity pairs $\boldsymbol{U}$.

2. Predict the trajectories for the velocity pairs in $\boldsymbol{U}$.

3. Compute $\rho(u_r, r)$ for the velocity pairs $(u_r, r) \in \boldsymbol{U}$.

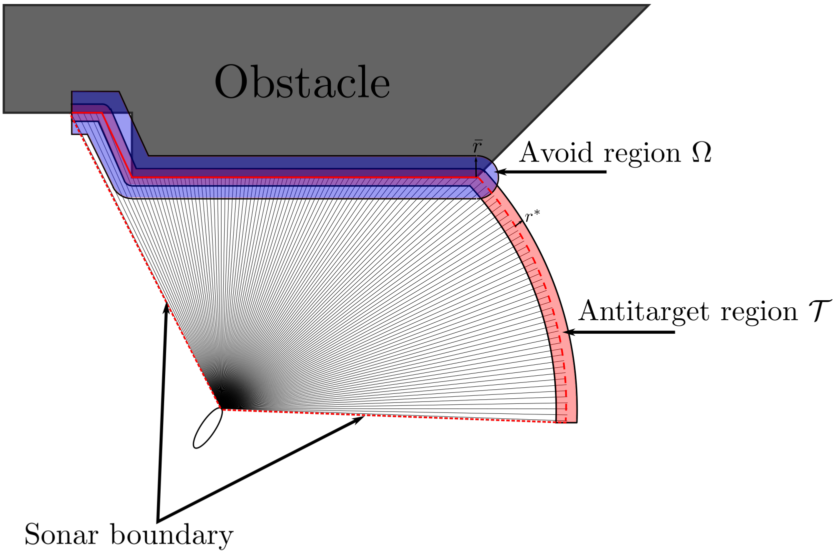


Figure 4.12: Local obstacle map. The estimated obstacle boundary is marked with a red line.

4. Compute the set of *Admissible velocities*, V_a, and compute the final search space as $V_r = V_f \cap V_a$.

V_f is found by employing (3.26) and (3.28). The set is discretized uniformly into the list $\boldsymbol{U}$, by specifying a number of surge velocities and yaw rates. If the desired surge speed and yaw rate (U_{rd}, r'_d) is in V_f, then the velocity pair (U_{rd}, r'_d) is included in $\boldsymbol{U}$. The same is done for the velocity pair $(0, 0)$. Figure 4.13 show an example of the dynamically feasible velocity set, together with the desired velocity (U_{rd}, r'_d) and the discrete set of velocity pairs.

Trajectories for the velocity pairs in $\boldsymbol{U}$ are predicted using the theory in section 3.4. Through solving (3.52), $\tilde{\boldsymbol{\nu}}_r(t)$ is obtained while $\bar{\boldsymbol{\nu}}_r(t)$ is required to simulate the AUV trajectories. Therefore, the relation $\tilde{\boldsymbol{\nu}}_r = \bar{\boldsymbol{\nu}}_r - \boldsymbol{\Gamma}_1^T \bar{\boldsymbol{\nu}}_{1r_d}$ is inserted into (3.52):

$$\bar{\boldsymbol{\nu}}_r(t) = \mathrm{e}^{\boldsymbol{A}t}\left(\bar{\boldsymbol{\nu}}_r(t_0) - \boldsymbol{\Gamma}_1^T \bar{\boldsymbol{\nu}}_{rd_1}\right) - \boldsymbol{A}^{-1}\left(\boldsymbol{I} - \mathrm{e}^{\boldsymbol{A}t}\right)\left(\boldsymbol{B}\bar{\boldsymbol{\nu}}_{1r_d} + \boldsymbol{G}\right) + \boldsymbol{\Gamma}_1^T \bar{\boldsymbol{\nu}}_{1r_d} \tag{4.25}$$

The trajectories are predicted 20 second forward, with a discrete sample time of 1 second. Hence, (4.25) has to be solved 20 times for each velocity pair. It is however possible to use vectorization [Mathworks, 2015] to compute the solutions more efficiently, as MATLAB is optimized for use with matrices and vectors. Given the list of n velocity pairs $\boldsymbol{U}$ as a matrix:

$$\boldsymbol{U} = \begin{bmatrix} \bar{\boldsymbol{\nu}}_{1r_{d_1}} & \bar{\boldsymbol{\nu}}_{1r_{d_2}} & \dots & \bar{\boldsymbol{\nu}}_{1r_{d_n}} \end{bmatrix}, \qquad \boldsymbol{U} \in \mathbb{R}^{2\times n} \tag{4.26}$$

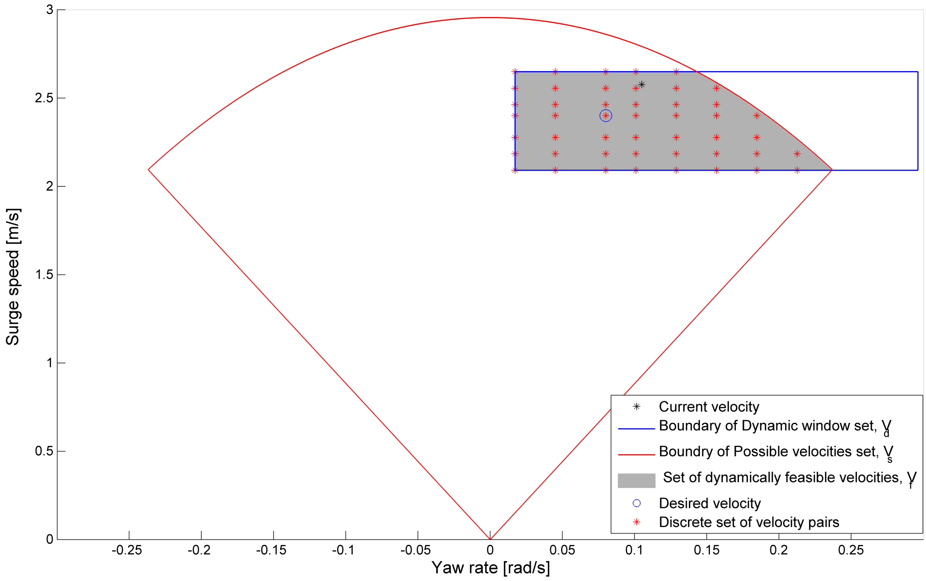


Figure 4.13: A discrete set of dynamically feasible velocities

The trajectories for the n velocity pairs can be solved simultaneously by defining the equation:

$$\boldsymbol{Y}(t) = \mathrm{e}^{\boldsymbol{A}t}\boldsymbol{Y}_0 - \boldsymbol{A}^{-1}\left(\boldsymbol{I} - \mathrm{e}^{\boldsymbol{A}t}\right)\left(\boldsymbol{B}\boldsymbol{U} + \boldsymbol{F}\right) + \boldsymbol{\Gamma}_1^T\boldsymbol{U} \tag{4.27}$$

Where:

$$\begin{aligned}
\boldsymbol{Y}(t) &\triangleq \begin{bmatrix}\bar{\boldsymbol{\nu}}_{r_1}(t) & \bar{\boldsymbol{\nu}}_{r_2}(t) & \dots & \bar{\boldsymbol{\nu}}_{r_n}(t)\end{bmatrix} &, \quad \boldsymbol{Y}(t) \in \mathbb{R}^{3\times n} \quad &(4.28\text{a})\\
\boldsymbol{Y}_0 &\triangleq \begin{bmatrix}\tilde{\boldsymbol{\nu}}_r(t_0) & \tilde{\boldsymbol{\nu}}_r(t_0) & \dots & \tilde{\boldsymbol{\nu}}_r(t_0)\end{bmatrix} - \boldsymbol{\Gamma}_1^T\boldsymbol{U}, &\quad \boldsymbol{Y}_0 \in \mathbb{R}^{3\times n} \quad &(4.28\text{b})\\
\boldsymbol{F} &\triangleq \begin{bmatrix}\boldsymbol{G} & \boldsymbol{G} & \dots & \boldsymbol{G}\end{bmatrix} &, \quad \boldsymbol{F} \in \mathbb{R}^{3\times n} \quad &(4.28\text{c})
\end{aligned}$$

Since the matrices $\boldsymbol{A}$ and $\mathrm{e}^{\boldsymbol{A}t}$ is independent of the velocity pair matrix $\boldsymbol{U}$, (4.27) only has to be solved once per time step for obtaining the solution to the AUV dynamics given the n velocity pairs. The AUV trajectories are further obtained by simulating (3.55). Algorithm 4 show the pseudo code for generating AUV trajectories for n velocity pairs.

The distances ρ and $\bar{\rho}$ are found by iterating along the predicted trajectories, and checking if the AUV enters Ω and $\mathcal{T}$. Algorithm 5 show the pseudo code for computing ρ and $\bar{\rho}$. The distances for a trajectory is illustrated in figure 4.14.

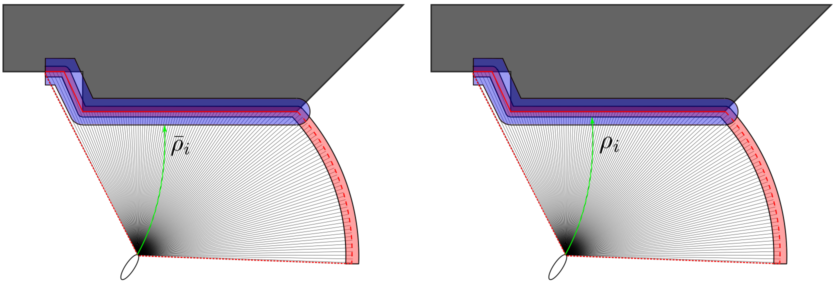
Instead of computing the set of admissible velocities V_a, the condition for a velocity pair to be part of V_a, (3.31), is evaluated for all the velocity pairs $(u_r, r) \in \boldsymbol{U}$. The admissible velocity pairs are then put in the discrete list $\boldsymbol{U}_a$. Finally, the resulting discrete search space is found as $\boldsymbol{U}_r = \boldsymbol{U} \cap \boldsymbol{U}_a$.

Algorithm 4 AUV trajectory prediction

```
$\boldsymbol{U} \leftarrow \begin{bmatrix} \bar{\boldsymbol{\nu}}_{1r_{d_1}} & \bar{\boldsymbol{\nu}}_{1r_{d_2}} & \dots & \bar{\boldsymbol{\nu}}_{1r_{d_n}} \end{bmatrix}$
$\boldsymbol{F} \leftarrow \mathbf{1}_{1\times n} \otimes \boldsymbol{G}$
$\boldsymbol{Y}_0 \leftarrow \mathbf{1}_{1\times n} \otimes \tilde{\boldsymbol{\nu}}_r(t_0) - \boldsymbol{\Gamma}_1^T \boldsymbol{U}$
for $t = 1 : 20$ do
    $\boldsymbol{\phi} \leftarrow \mathrm{e}^{\boldsymbol{A}t}$
    $\boldsymbol{Y}(t) \leftarrow \boldsymbol{\phi}\boldsymbol{Y}_0 - \boldsymbol{A}^{-1}(\boldsymbol{I} - \boldsymbol{\phi})(\boldsymbol{B}\boldsymbol{U} + \boldsymbol{F}) + \boldsymbol{\Gamma}_1^T \boldsymbol{U}$
end for
for all velocity pairs $i$ in $\boldsymbol{U}$ do
    $\bar{\boldsymbol{\nu}}_{1r_d} = \bar{\boldsymbol{\nu}}_{1r_{d_i}}$    ▷ Pick out the dynamics solution for velocity pair $i$
    $\bar{\boldsymbol{\eta}}_i(t) \leftarrow \mathrm{ModifiedEuler}(\bar{\boldsymbol{\nu}}_{1r_d})$    ▷ Simulate (3.55) with the modified Euler method
end for
$\boldsymbol{P} \leftarrow \begin{bmatrix} \bar{\boldsymbol{\eta}}_i(t) & \bar{\boldsymbol{\eta}}_2(t) & \dots & \bar{\boldsymbol{\eta}}_n(t) \end{bmatrix}$
return $\boldsymbol{P}, \boldsymbol{Y}$    ▷ Return the velocity and position prediction for all the velocity pairs
```

Algorithm 5 AUV distance calculation

```
for all velocity pairs $i$ in $\boldsymbol{U}$ do
    $\bar{\rho}_i \leftarrow 0$
    $\rho_i \leftarrow 0$
    for $t = 1 : 20$ do
        $\Delta_s \leftarrow \left\| \begin{bmatrix} \bar{N}(t) - \bar{N}(t-1) \\ \bar{E}(t) - \bar{E}(t-1) \end{bmatrix} \right\|_2$    ▷ Compute the distance the AUV travels in this iteration. $\bar{N}(t)$ and $\bar{E}(t)$ is the predicted AUV positions given as $\bar{\boldsymbol{\eta}}_i(t) = \begin{bmatrix} \bar{N}(t) & \bar{E}(t) & \psi(t) \end{bmatrix}^T$
        if $\bar{\boldsymbol{\eta}}_i(t) \notin \Omega$ then
            $\bar{\rho}_i \leftarrow \bar{\rho}_i + \Delta_s$
        end if
        if $\bar{\boldsymbol{\eta}}_i(t) \notin \mathcal{T}$ then
            $\rho_i \leftarrow \rho_i + \Delta_s$
        else
            break
        end if
    end for
end for
return $\bar{\rho}, \rho$
```



(a) Distance until the AUV enters Ω **(b)** Distance until the AUV enters $\mathcal{T}$

Figure 4.14: AUV distances $\bar{\rho}_i$ and ρ_i given the trajectory for velocity pair i

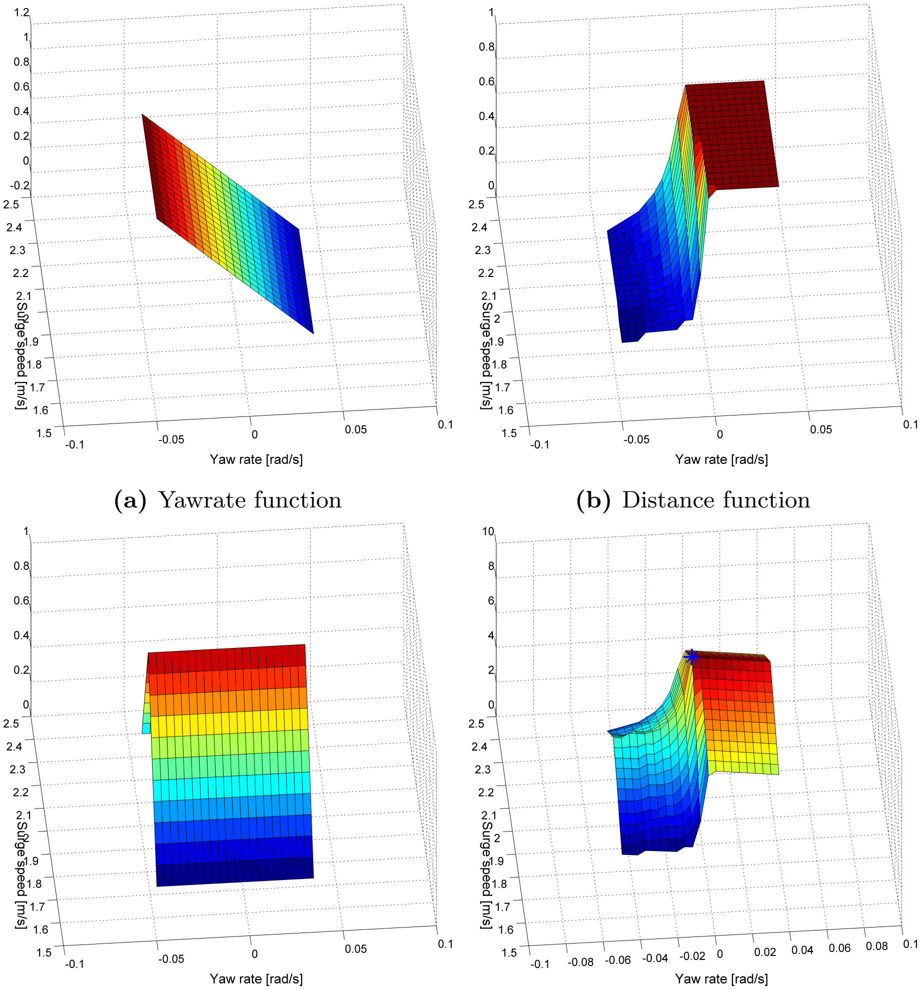
4.6.3 Velocity pair selection

The optimal velocity pair is selected through maximizing the objective function given in (3.64) for the discrete velocities in U_r:

$$(u_{rd}, r_d) = \underset{(u_r, r) \in \boldsymbol{U}_r}{\operatorname{argmax}} G(u_r, r) \tag{4.29}$$

Notice that the maximizing problem (4.29) is solved numerically by computing $G(u_r, r)$ for all the velocity pairs $(u_r, r) \in \boldsymbol{U}_r$. This is an approximation to solving it in the continuous search space V_r. The objective function is computed as described in section 3.5. An example of the objective function parts and the combined objective function is shown in figure 4.15. The scaling parameters are selected as $\alpha = 1$, $\beta = 6$ s^{-1} and $\gamma = 3$. Figure 4.16 show the predicted AUV trajectories corresponding to the objective function in figure 4.15. The trajectory corresponding to the selected velocity pair is shown in yellow. Notice the side-slip of the AUV, which compensates for the ocean current.

The Dynamic window algorithm is summarized in algorithm 6. The most time-consuming part of the algorithm is to predict the AUV trajectories, which takes about 285 ms for a search space consisting of 451 velocity pairs. About 265 ms of this is used for simulating the kinematics, which is not vectorized. Most likely, the computational time can be reduced a great deal by vectorizing this part of algorithm (see line 8 to 11 of algorithm 4).



(a) Yawrate function

(b) Distance function

(c) Velocity function

(d) Combined objective function, with the selected velocity pair marked with a blue asterisk

Figure 4.15: The Dynamic window objective function parts (a)-(c) and the combined objective function (d).

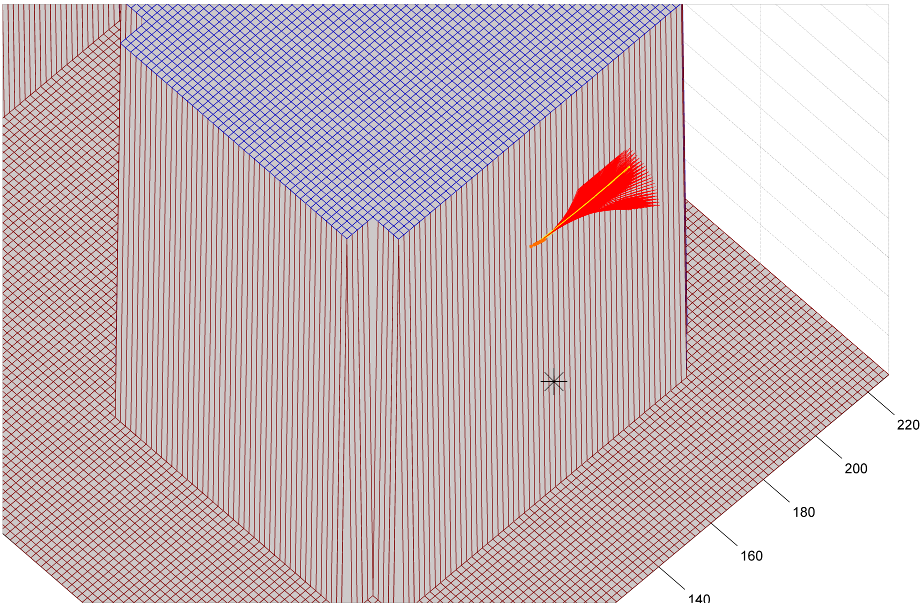


Figure 4.16: Predicted AUV trajectories with the selected trajectory in yellow

Algorithm 6 Dynamic window algorithm

```
1: Compute the set of dynamically feasible velocity pairs $V_f$
2: Discretize $V_f$ into $n$ velocity pairs in $\boldsymbol{U}$
3: Predict the AUV trajectories for the velocity pairs in $\boldsymbol{U}$ using algorithm 4
4: Compute $\bar{\rho}$ and $\rho$ using algorithm 5
5: Compute a discrete set of admissible velocities $\boldsymbol{U}_a$ by evaluating (3.31) for
   the velocity pairs in $\boldsymbol{U}$
6: Find the intersection $\boldsymbol{U}_r = \boldsymbol{U} \cap \boldsymbol{U}_a$
7: Evaluate the objective function (3.64) for the velocity pairs in $\boldsymbol{U}_r$
8: Numerically search for the maximum value of the objective function, and
   apply the corresponding velocity pair in $\boldsymbol{U}_r$ as the algorithm output $(u_{rd}, r_d)$
```

Chapter 5

Stability analysis

The stability analysis follows along the lines of [Caharija, 2014, Theorem 6.1], and is done in 3DOFs under assumption 3.1. It is worth noting that [Caharija, 2014, Theorem 6.1] establishes stability properties for an underactuated surface vessel, not an AUV. However, the model in (3.9a) and (3.9b) complies with the model used in the theorem, hence it is applicable.

Assumption 5.1. *It is assumed that the AUV is operating in an obstacle-free environment.*

Remark 5.1. *Assuming an obstacle-free environment is of course a major assumption, and may seem somewhat inapplicable for proving stability of a system containing a collision avoidance system. It is, however, necessary to prove stability both with and without obstacles. Hence, this is a first step of a complete stability proof.*

5.1 AUV model in component form

From (3.9a) and (3.12), the AUV model is written as:

$$\dot{\bar{\boldsymbol{\eta}}} = \boldsymbol{R}(\bar{\boldsymbol{\eta}})\bar{\boldsymbol{\nu}}_r + \begin{bmatrix} \boldsymbol{V}_c \\ 0 \end{bmatrix} \tag{5.1a}$$

$$\dot{\bar{\boldsymbol{\nu}}}_r = \begin{bmatrix} \tau_u \\ 0 \\ \tau_r \end{bmatrix} - \bar{\boldsymbol{M}}^{-1} \left(\bar{\boldsymbol{C}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r - \bar{\boldsymbol{D}}(\bar{\boldsymbol{\nu}}_r)\bar{\boldsymbol{\nu}}_r \right) \tag{5.1b}$$

Where τ_u and τ_r is given from the transformation (3.11).

Writing (5.1a) and (5.1b) in component form:

$$\begin{aligned}
\dot{\bar{N}} &= u_r \cos\psi - \bar{v}_r \sin\psi + V_{\bar{N}} \\
\dot{\bar{E}} &= u_r \sin\psi + \bar{v}_r \cos\psi + V_{\bar{E}} \\
\dot{\psi} &= r \\
\dot{u}_r &= F_{ur}(u_r, \bar{v}, r) + \tau_u \\
\dot{\bar{v}}_r &= X(u_r)r + Y(u_r)\bar{v}_r \\
\dot{r} &= F_r(u_r, \bar{v}, r) + \tau_r
\end{aligned} \tag{5.2}$$

Where $F_{ur}(u_r, \bar{v}_r, r)$, $F_r(u_r, \bar{v}_r, r)$, $X(u_r)$ and $Y(u_r)$ are given in appendix 5.A.

Assumption 5.2. *$Y(u_r)$ is assumed to satisfy $Y(u_r) \leq -Y^{min} < 0, \quad \forall u_r \in [-V_{max}, U_{rd}]$, where Y^{min} is a positive constant. In addition, $|Y(u_r)|$ is assumed to be strictly increasing for $u_r > 0$.*

Remark 5.2. *Assumption 5.2 is justified by contradiction: $Y(u_r) \geq 0$ would imply an undamped or unstable vehicle in sway, which is not the case in practice [Børhaug et al., 2007, Caharija, 2014].*

5.2 Control objective

To simplify the proof without any loss of generality, the desired path $\mathcal{P}$ is assumed to be aligned with the north-axis of the inertial reference frame. Hence, $\mathcal{P} \triangleq \left\{ (\bar{N}, \bar{E}) \in \mathbb{R} \times \mathbb{R} \middle| \bar{E} = 0 \right\}$. Hence, the cross track error is equal to the east position, $e(t) = \bar{E}(t)$. The control goal is:

$$\lim_{t\to\infty} \bar{E}(t) = 0 \tag{5.3a}$$

$$\lim_{t\to\infty} \psi(t) = \psi_{ss} \tag{5.3b}$$

$$\lim_{t\to\infty} u_r(t) = U_{rd} \tag{5.3c}$$

Assumption 5.3. *The surge thrust capacity of the AUV is assumed to be sufficient such that U_{rd} satisfies $U_{rd} > V_{max}$, where V_{max} is defined in assumption 2.2.*

5.3 Control system

Only the horizontal subsystem of the control system shown in figure 4.1 is needed when the AUV is modeled in 3DOF.

From (4.21), the feedback linearizing controllers are given as:

$$\begin{aligned}
\tau_u &= \bar{n}_1(\bar{\boldsymbol{\nu}}_r) - k_u(u_r - u_{rd}) + \dot{u}_{rd} \\
\tau_r &= \bar{n}_3(\bar{\boldsymbol{\nu}}_r) - k_r(r - r_d) + \dot{r}_d
\end{aligned} \tag{5.4}$$

Noting that $\bar{n}_1(\bar{\boldsymbol{\nu}}_r) = -F_u(u, \bar{v}, r)$ and $\bar{n}_3(\bar{\boldsymbol{\nu}}_r) = -F_r(u, \bar{v}, r)$, the controllers can be written as:

$$\begin{aligned} \tau_u &= -F_u(u, \bar{v}, r) - k_u\,(u - u_d) + \dot{u}_d \\ \tau_r &= -F_r(u, \bar{v}, r) - k_r\,(r - r_d) + \dot{r}_d \end{aligned} \tag{5.5}$$

From (4.23), the desired yaw rate is selected using a proportional controller, with a derivative feed forward:

$$r'_d = -k_\psi\,(\psi - \psi_d) + \dot{\psi}_d \tag{5.6}$$

Where the desired heading is chosen as $\psi_d \triangleq \psi_{ILOS}$. For the desired path $\mathcal{P}$, the ILOS control law (4.13) simplifies to:

$$\begin{aligned} \psi_{ILOS} &= -\arctan\left(\frac{\bar{E} + \sigma \bar{E}_{int}}{\Delta}\right) \\ \dot{\bar{E}}_{int} &= \frac{\Delta \bar{E}}{\left(\bar{E} + \sigma \bar{E}_{int}\right)^2 + \Delta^2} \end{aligned} \tag{5.7}$$

Assumption 5.4. *The sample time of the Dynamic window algorithm is assumed to be sufficiently fast such that the time delay it represents does not significantly affect the closed loop dynamics.*

Assumption 5.5. *The search space, V_r, of the Dynamic window algorithm is assumed to contain the velocity pair (U_d, r'_d).*

Remark 5.3. *The search space of the Dynamic window algorithm is defined as $V_r = V_f \cap V_a$, where V_f are the set of dynamically feasible velocities and V_a are the set of admissible velocities. In an obstacle-free environment, as Assumption 5.1 guarantees, V_a will not impose any limitation to the search space. Hence, $V_r = V_f$, and Assumption 5.5 should therefore not impose any practical limitation.*

From (3.64), the objective function of the Dynamic window algorithm is:

$$G(u_r, r) = \alpha \cdot \text{yawrate}(u_r, r, r'_d) + \beta \cdot \text{dist}(u_r, r) + \gamma \cdot \text{velocity}(u_r, r, U_{rd}) \tag{5.8}$$

The functions yawrate(u_r, r, r'_d) and velocity(u_r, r, U_{rd}) have global maximums at the points $r = r'_d$ and $u_r = U_{rd}$ respectively. In an obstacle-free environment, the dist(u_r, r) term is equal for all velocities in V_r. Hence, when Assumption 5.1 is satisfied and since $\alpha, \beta > 0$, the function $G(u_r, r)$ have a global maximum at $(u_r, r) = (U_{rd}, r'_d)$.

Applying assumptions 5.1, 5.4 and 5.5 it follows that:

$$(u_{rd}, r_d) = \underset{(u_r, r)}{\text{argmax}}\; G(u_r, r) = (U_{rd}, r'_d) \tag{5.9}$$

5.4 Stability of the closed loop system

Theorem 5.1. *Given an AUV described by the dynamical system* (5.2)*. If assumptions 2.2, 5.1, 5.2, 5.3, 5.4 and 5.5 hold and, if the look-ahead distance* Δ *and the integral gain constant* σ *satisfy:*

$$\Delta > \frac{|X(U_{rd})|}{|Y(U_{rd}|}\left(\frac{5}{4}\frac{U_{rd}+V_{max}+\sigma}{U_{rd}-V_{max}-\sigma}+1\right) \tag{5.10}$$

$$0 < \sigma < U_{rd} - V_{max} \tag{5.11}$$

then the control system in section 5.3 guarantee achievement of the control objectives in (5.3) *with:*

$$\psi_{ss} = -\arctan\left(\frac{V_{\bar{E}}}{\sqrt{U_{rd}^2 - V_{\bar{E}}^2}}\right) \tag{5.12}$$

5.4.1 Proof of theorem 5.1

First, the controller dynamics are derived. Assumptions 5.1, 5.4 and 5.5 allows $u_{rd} = U_{rd}$ and $r_d = r'_d$, as in (5.9). Given the tracking errors:

$$\begin{aligned} \tilde{u}_r &= u_r - u_{rd} \\ &= u_r - U_{rd} \\ \tilde{r} &= r - r_d \\ \tilde{\psi} &= \psi - \psi_d \end{aligned} \tag{5.13}$$

and the controllers in (5.5) and (5.6), the surge and yaw error dynamics can be described as:

$$\dot{\tilde{u}}_r = -k_u\tilde{u}_r \tag{5.14a}$$

$$\dot{\tilde{r}} = -k_r\tilde{r} \tag{5.14b}$$

$$\begin{aligned} \dot{\tilde{\psi}} &= \dot{\psi} - \dot{\psi}_d \\ &= r - r'_d - k_\psi(\psi - \psi_d) \\ &= \tilde{r} - k_\psi\tilde{\psi} \end{aligned} \tag{5.14c}$$

The last equality is found by substituting $r'_d = r_d$ into (5.14c). The resulting system of (5.14a), (5.14b) and (5.14c) can be written as a linear state space system, with a state vector $\boldsymbol{\xi} = \begin{bmatrix}\tilde{u}_r & \tilde{\psi} & \tilde{r}\end{bmatrix}^T$:

$$\dot{\boldsymbol{\xi}} = \begin{bmatrix} -k_u & 0 & 0 \\ 0 & -k_\psi & 1 \\ 0 & 0 & -k_r \end{bmatrix} \begin{bmatrix} \tilde{u}_r \\ \tilde{\psi} \\ \tilde{r} \end{bmatrix} \triangleq \boldsymbol{\Lambda\xi} \tag{5.15}$$

(5.15) is a LTI system and all the controller gains are strictly positive, $k_u, k_r, k_\psi > 0$. Hence, the matrix $\mathbf{\Lambda}$ is Hurwitz, and the origin $\boldsymbol{\xi} = \mathbf{0}$ is UGES. It is worth noting that $\mathbf{\Lambda}$ have a different structure than the one of [Caharija, 2014], due to the difference in the control structure. However, the system holds the same stability properties. It follows that the control objective (5.3c) is achieved with exponential convergence properties.

The dynamics of the cross track error, $\bar{E}$, and the relative sway speed, $\bar{v}_r$, are given in (5.2). Inserting $u_r = \tilde{u} + U_{rd}$ and $\psi = \tilde{\psi} + \psi_d$, the system is given as:

$$\begin{aligned} \dot{\bar{E}}_{int} &= \frac{\Delta \bar{E}}{\left(\bar{E} + \sigma \bar{E}_{int}\right)^2 + \Delta^2} \\ \dot{\bar{E}} &= (\tilde{u}_r + U_{rd}) \sin\left(\psi_d + \tilde{\psi}\right) + \bar{v} \cos\left(\psi_d + \tilde{\psi}\right) \\ \dot{\bar{v}}_r &= X(\tilde{u}_r + U_{rd})(\tilde{r} + r_d) + Y(\tilde{u} + U_{rd})\bar{v}_r \end{aligned} \tag{5.16}$$

The equilibrium of the system (5.16) is given as:

$$\bar{E}_{int}^{eq} = \frac{\Delta}{\sigma} \frac{V_{\bar{E}}}{\sqrt{U_{rd}^2 - V_{\bar{E}}^2}}, \quad \bar{E}^{eq} = 0, \quad \bar{v}_r^{eq} = 0 \tag{5.17}$$

The equilibrium is moved to the origin by defining the new variables $e_1 \triangleq \bar{E}_{int} - \bar{E}_{int}^{eq}$ and $e_2 = \bar{E} + \sigma e_1$. By computing the dynamics of the transformed system, substituting $\psi_d \triangleq \psi_{ILOS}$, where ψ_{ILOS} is given in (5.7), and factoring for the signals in $\boldsymbol{\xi}$ the closed loop dynamics can be written as:

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{\bar{v}}_r \end{bmatrix} = \boldsymbol{A}(e_2) \begin{bmatrix} e_1 \\ e_2 \\ \bar{v}_r \end{bmatrix} + \boldsymbol{B}(e_2) + \boldsymbol{H}(\bar{E}, \bar{E}_{int}, \psi_d, \bar{v}_r, \boldsymbol{\xi})\boldsymbol{\xi} \tag{5.18a}$$

$$\dot{\boldsymbol{\xi}} = \mathbf{\Lambda}\boldsymbol{\xi} \tag{5.18b}$$

Note that (5.18) is a cascaded system, where the linear UGES system (5.18b) perturbs the nonlinear dynamics (5.18a) through an interconnection term. The interconnection term $\boldsymbol{H}(\bar{E}, \bar{E}_{int}, \psi_d, \bar{v}_r, \boldsymbol{\xi})$ contains all the terms which disappear when $\boldsymbol{\xi} = \mathbf{0}$. $\boldsymbol{A}(e_2)$, $\boldsymbol{B}(e_2)$ and $\boldsymbol{H}(\bar{E}, \bar{E}_{int}, \psi_d, \bar{v}_r, \boldsymbol{\xi})$ are given as:

$$\boldsymbol{A}(e_2) = \begin{bmatrix} -\frac{\sigma\Delta}{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2} & \frac{\Delta}{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2} & 0 \\ -\frac{\sigma^2\Delta}{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2} & -\frac{U_{rd}}{\sqrt{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2}} + \frac{\sigma\Delta}{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2} & \frac{\Delta}{\sqrt{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2}} \\ \frac{\sigma^2\Delta^2 X(U_{rd})}{\left(\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2\right)^2} & \frac{U_{rd}\Delta X(U_{rd})}{\left(\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2\right)^{\frac{3}{2}}} - \frac{\sigma\Delta^2 X(U_{rd})}{\left(\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2\right)^2} & Y(U_{rd}) - \frac{\Delta^2 X(U_{rd})}{\left(\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2\right)^{\frac{3}{2}}} \end{bmatrix} \tag{5.19}$$

$$\boldsymbol{B}(e_2) = \begin{bmatrix} 0 \\ V_{\bar{E}} f(e_2) \\ -\frac{\Delta X(U_{rd})}{\left(e_2+\sigma \bar{E}_{int}^{eq}\right)^2+\Delta^2} V_{\bar{E}} f(e_2) \end{bmatrix} \tag{5.20}$$

$$\boldsymbol{H}(\bar{E}, \bar{E}_{int}, \psi_d, \bar{v}_r, \boldsymbol{\xi}) = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ -\frac{\Delta X(\tilde{u}_r+U_{rd})}{\left(e_2+\sigma \bar{E}_{int}^{eq}\right)^2+\Delta^2} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{h}_{\bar{E}}^T(\psi_d, \bar{v}_r, \boldsymbol{\xi}) \\ \boldsymbol{h}_{\bar{v}_r}^T(\bar{E}, \bar{E}_{int}, \bar{v}_r, \boldsymbol{\xi}) \end{bmatrix} \tag{5.21}$$

The function $f(e_2)$ and the vectors $\boldsymbol{h}_{\bar{E}}$ and $\boldsymbol{h}_{\bar{v}_r}$ are given as:

$$f(e_2) = 1 - \frac{\sqrt{\left(\sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2}}{\sqrt{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2}} \tag{5.22}$$

$$\boldsymbol{h}_{\bar{E}}(\psi_d, \bar{v}_r, \boldsymbol{\xi}) = \begin{bmatrix} \sin(\tilde{\psi} + \psi_d) \\ U_{rd}\left(\frac{\sin\tilde{\psi}}{\tilde{\psi}}\cos\psi_d + \frac{\cos\tilde{\psi} - 1}{\tilde{\psi}}\sin\psi_d\right) \\ \qquad + \bar{v}_r\left(\frac{\cos\tilde{\psi} - 1}{\tilde{\psi}}\cos\psi_d - \frac{\sin\tilde{\psi}}{\tilde{\psi}}\sin\psi_d\right) \\ 0 \end{bmatrix} \tag{5.23}$$

$$\boldsymbol{h}_{\bar{v}_r}(\bar{E}, \bar{E}_{int}, \bar{v}_r, \boldsymbol{\xi}) = \begin{bmatrix} \frac{X(\tilde{u}_r+U_{rd}) - X(U_{rd})}{\tilde{u}_r}\gamma(\bar{E}, \bar{E}_{int}, \bar{v}_r) \\ -k_{\psi} X(\tilde{u}_r + U_{rd}) \\ X(\tilde{u}_r + U_{rd}) \end{bmatrix} \tag{5.24}$$

Where the function $\gamma(\bar{E}, \bar{E}_{int}, \bar{v}_r)$ is given as:

$$\begin{aligned} \gamma(\bar{E}, \bar{E}_{int}, \bar{v}_r) = {} & \frac{U_{rd}\Delta\left(\bar{E} + \sigma\bar{E}_{int}\right)}{\left(\left(\bar{E} + \sigma\bar{E}_{int}\right)^2 + \Delta^2\right)^{3/2}} - \frac{\Delta^2}{\left(\left(\bar{E} + \sigma\bar{E}_{int}\right)^2 + \Delta^2\right)^{3/2}}\bar{v}_r \\ & - \frac{\sigma\Delta^2}{\left(\left(\bar{E} + \sigma\bar{E}_{int}\right)^2 + \Delta^2\right)^2}\bar{E} - \frac{\Delta V_{\bar{E}}}{\left(\bar{E} + \sigma\bar{E}_{int}\right)^2 + \Delta^2} \end{aligned} \tag{5.25}$$

It is worth noting that the expression for $\boldsymbol{h}_{\bar{v}_r}(\bar{E}, \bar{E}_{int}, \bar{v}_r, \boldsymbol{\xi})$ differs from the one of [Caharija, 2014], due to the difference in the control structure. This does, however, not change the required properties of $\boldsymbol{H}(\bar{E}, \bar{E}_{int}, \psi_d, \bar{v}_r, \boldsymbol{\xi})$. Also note that the following bound holds for $f(e_2)$:

$$|f(e_2)| \leq \frac{|e_2|}{\sqrt{\left(e_2 + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2}} \tag{5.26}$$

To establish the stability properties of the cascaded system (5.18), the nominal system defined on the manifold $\boldsymbol{\xi} = \mathbf{0}$ is first analyzed:

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{\bar{v}}_r \end{bmatrix} = \boldsymbol{A}(e_2) \begin{bmatrix} e_1 \\ e_2 \\ \bar{v}_r \end{bmatrix} + \boldsymbol{B}(e_2) \tag{5.27}$$

Lemma 5.1 states the stability properties of the nominal system (5.27)

Lemma 5.1. *Under the conditions of Theorem 5.1, the origin* $\begin{bmatrix} e_1 & e_2 & \bar{v}_r \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$ *of the system* (5.27) *is UGAS and ULES.*

Proof. Lemma 5.1 is proven through Lyapunov analysis using the Lyapunov function candidate (LFC):

$$V \triangleq \frac{1}{2}\sigma^2 e_1^2 + \frac{1}{2}e_2^2 + \frac{1}{2}\mu\bar{v}_r^2, \quad \mu > 0 \tag{5.28}$$

Under the conditions of Theorem 5.1, and the inequality (5.26), negative definitiveness of $\dot{V}$ is shown. Further, positive definitiveness of V is guaranteed. And since V is radially unbounded, (5.27) is UGAS. Moreover, in a neighborhood of the origin, $\dot{V}$ can be upper bounded by a negative quadratic function in e_1, e_2 and $\bar{v}_r$. This, together with the fact that V is quadratic in e_1, e_2 and $\bar{v}_r$ states that (5.27) also is ULES.

The details of the proof is shown in Appendix 5.B. □

Further, the stability of the cascaded system (5.18) is analyzed. The perturbing system, $\boldsymbol{\xi}$, is UGES, and the interconnection term $\boldsymbol{H}(\bar{E}, \bar{E}_{int}, \psi_d, \bar{v}_r, \boldsymbol{\xi})$ can be shown to satisfy:

$$\left\| \boldsymbol{H}(\bar{E}, \bar{E}_{int}, \psi_d, \bar{v}_r, \boldsymbol{\xi}) \right\| \leq \theta_1(\|\boldsymbol{\xi}\|)(|\bar{E}| + |\bar{E}_{int}| + |\bar{v}_r|) + \theta_2(\|\boldsymbol{\xi}\|) \tag{5.29}$$

where $\theta_1(\cdot)$ and $\theta_2(\cdot)$ are some continuous non-negative functions.

Applying Theorem A.1, we can conclude that the cascaded system (5.18) is UGAS. Further, applying Lemma A.1, we can conclude that the cascaded system (5.18) also is ULES. Hence, under the conditions of Theorem 5.1, the origin $\begin{bmatrix} e_1 & e_2 & \bar{v}_r & \boldsymbol{\xi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & \mathbf{0} \end{bmatrix}$ of the system (5.18) is UGAS and ULES. The control objectives (5.3a) and (5.3b) are achieved with exponential convergence properties, with ψ_{ss} given in (5.12).

5.A Appendix: Functional expressions

$$F_{ur}(u_r, \bar{v}_r, r) \triangleq -\frac{\bar{d}_{11}(u_r)u_r - \bar{m}_{22}\bar{v}_r r - \bar{m}_{23}r^2}{\bar{m}_{11}} \tag{5.30}$$

$$F_r(u_r, \bar{v}_r, r) \triangleq -\frac{1}{\bar{m}_{22}\bar{m}_{33} - \bar{m}_{23}^2}\big((\bar{m}_{22}\bar{d}_{33} - \bar{m}_{23}\bar{d}_{23})r - \bar{m}_{22}(\bar{m}_{11} - \bar{m}_{22})u_r\bar{v}_r \\ + \bar{m}_{23}(\bar{m}_{22} - \bar{m}_{11})u_r r + (\bar{m}_{22}\bar{d}_{32} - \bar{m}_{23}\bar{d}_{22})\bar{v}_r\big) \quad (5.31)$$

$$X(u_r) \triangleq -\frac{\left(\bar{m}_{33}\bar{m}_{11} - \bar{m}_{23}^2\right)u_r + \bar{m}_{33}\bar{d}_{23} - \bar{m}_{23}\bar{d}_{33}}{\bar{m}_{22}\bar{m}_{33} - \bar{m}_{23}^2} \quad (5.32)$$

$$Y(u_r) \triangleq -\frac{\bar{m}_{33}\bar{d}_{22} - \bar{m}_{23}\bar{d}_{32} - \bar{m}_{23}\left(\bar{m}_{22} - \bar{m}_{11}\right)u_r}{\bar{m}_{22}\bar{m}_{33} - \bar{m}_{23}^2} \quad (5.33)$$

5.B Appendix: Proof of lemma 5.1

The system (5.27) is given again:

$$\begin{bmatrix}\dot{e}_1\\ \dot{e}_2\\ \dot{\bar{v}}_r\end{bmatrix} = \boldsymbol{A}(e_2)\begin{bmatrix}e_1\\ e_2\\ \bar{v}_r\end{bmatrix} + \boldsymbol{B}(e_2) \quad (5.34)$$

Where $\boldsymbol{A}(e_2)$ and $\boldsymbol{B}(e_2)$ are given in (5.19) and (5.20), respectively. The system (5.34) have an equilibrium point at the origin, $\begin{bmatrix}e_1^{eq} & e_2^{eq} & \bar{v}_r^{eq}\end{bmatrix}^T = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$. Consider the quadratic LFC:

$$V \triangleq \frac{1}{2}\sigma^2 e_1^2 + \frac{1}{2}e_2^2 + \frac{1}{2}\mu\bar{v}_r^2, \quad \mu > 0 \quad (5.35)$$

The time derivative of V is:

$$\dot{V} = -\frac{\sigma^3\Delta}{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2}e_1^2 + \frac{\sigma\Delta - U_{rd}\sqrt{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2}}{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2}e_2^2 \\ + \frac{\Delta}{\sqrt{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2}}e_2\bar{v}_r + V_{\bar{E}}f(e_2) - \mu\frac{\Delta X(U_{rd})}{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2}V_{\bar{E}}f(e_2)\bar{v}_r \\ - \mu\left(-Y(U_{rd}) + \frac{\Delta^2 X(U_{rd})}{\left(\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2\right)^{\frac{3}{2}}}\right)\bar{v}_r^2 + \frac{\mu\sigma^2\Delta^2 X(U_{rd})}{\left(\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2\right)^2}e_1\bar{v}_r \\ + \mu\left(\frac{U_{rd}\Delta X(U_{rd})}{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2} - \frac{\sigma\Delta^2 X(U_{rd})}{\left(\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2\right)^{\frac{3}{2}}}\right)\frac{e_2 e_3}{\sqrt{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2}} \quad (5.36)$$

Assumptions 2.2, 5.2 and 5.3, inequality (5.26) together with the following properties:

$$\max\left\{\frac{\Delta}{\left(e_2 + \sigma\bar{E}_{int}^{eq}\right)^2 + \Delta^2}\right\} = \frac{1}{\Delta} \quad (5.37)$$

$$\max\left\{\frac{\Delta^2}{\left(\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2\right)^{\frac{3}{2}}}\right\}=\frac{1}{\Delta} \tag{5.38}$$

$$\min\left\{\sqrt{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}\right\}=\Delta \tag{5.39}$$

yield the following bound on $\dot{V}$:

$$\begin{aligned}\dot{V}\leq&-\frac{\sigma^3\Delta}{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}e_1^2-\Delta\frac{U_{rd}-\sigma-V_{max}}{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}e_2^2\\&+\frac{\Delta}{\sqrt{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}}|e_2||\bar{v}_r|+\frac{\mu\sigma^2|X(U_{rd})|}{\Delta\sqrt{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}}|e_1||\bar{v}_r|\\&+\frac{\mu|X(U_{rd})|}{\Delta\sqrt{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}}\left(U_{rd}+\sigma+V_{max}\right)|e_2||\bar{v}_r|\\&-\mu\left(|Y(U_{rd})|-\frac{|X(U_{rd})|}{\Delta}\right)\bar{v}_r^2\end{aligned} \tag{5.40}$$

The variables $\bar{e}_1=\frac{e_1}{\sqrt{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}}$ and $\bar{e}_2=\frac{e_2}{\sqrt{\left(e_2+\sigma\bar{E}^{eq}_{int}\right)^2+\Delta^2}}$ are introduced to simplify the expression (5.40). The expression becomes:

$$\begin{aligned}\dot{V}\leq&-\sigma^3\Delta\bar{e}_1^2-\Delta\left(U_{rd}-\sigma-V_{max}\right)\bar{e}_2^2-\mu\left(|Y(U_{rd})|-\frac{|X(U_{rd})|}{\Delta}\right)\bar{v}_r^2+\Delta|\bar{e}_2||\bar{v}_r|\\&+\frac{\mu\sigma^2|X(U_{rd})|}{\Delta}|\bar{e}_1||\bar{v}_r|+\frac{\mu|X(U_{rd})|}{\Delta}\left(U_{rd}+\sigma+V_{max}\right)|\bar{e}_2||\bar{v}_r|\end{aligned} \tag{5.41}$$

This can further be rearranged as:

$$\dot{V}\leq -W_1\left(|\bar{e}_1|,|\bar{v}_r|\right)-W_2\left(|\bar{e}_2|,|\bar{v}_r|\right) \tag{5.42}$$

$$\begin{aligned}W_1\triangleq\ &\sigma^3\Delta|\bar{e}_1|^2-\frac{\mu\sigma^2|X(U_{rd})|}{\Delta}|\bar{e}_1||\bar{v}_r|\\&+\mu\eta\left(|Y(U_{rd})|-\frac{|X(U_{rd})|}{\Delta}\right)|\bar{v}_r|^2\end{aligned} \tag{5.43a}$$

$$W_2\triangleq\Delta\begin{bmatrix}|\bar{e}_2| & |\bar{v}_r|\end{bmatrix}\begin{bmatrix}\beta & -\alpha\\ -\alpha & \frac{\alpha(2\alpha-1)}{\beta}\end{bmatrix}\begin{bmatrix}|\bar{e}_2|\\ |\bar{v}_r|\end{bmatrix} \tag{5.43b}$$

where $0<\eta<1$, $\beta\triangleq U_{rd}-\sigma-V_{max}$ and α is given as:

$$\alpha\triangleq(1-\eta)\frac{\left(U_{rd}-\sigma-V_{max}\right)\left(\Delta|Y(U_{rd})|-|X(U_{rd})|\right)}{|X(U_{rd})|\left(U_{rd}+\sigma+V_{max}\right)} \tag{5.44}$$

The parameter μ is chosen as:

$$\mu \triangleq \frac{\Delta^2 \left(2\alpha - 1\right)}{|X(U_{rd})| \left(U_{rd} + \sigma + V_{max}\right)} \tag{5.45}$$

Notice that, in (5.42), the term $\mu \left(|Y(U_{rd})| - \frac{|X(U_{rd})|}{\Delta}\right) |\bar{v}_r|^2$ from (5.41) has been split into $\eta\mu \left(|Y(U_{rd})| - \frac{|X(U_{rd})|}{\Delta}\right) |\bar{v}_r|^2$ and $(1-\eta)\,\mu \left(|Y(U_{rd})| - \frac{|X(U_{rd})|}{\Delta}\right) |\bar{v}_r|^2$. This makes it possible to shift the analysis of $\dot{V}$ to W_1 and W_2, as positive definitiveness of W_1 and W_2 ensures a negative definite $\dot{V}$. Positive definitiveness of W_1 is ensured if:

$$\Delta > \frac{|X(U_{rd})|}{|Y(U_{rd})|} \tag{5.46a}$$

$$\mu < \frac{4\eta\Delta^2 \left(\Delta|Y(U_{rd})| - |X(U_{rd})|\right)}{\sigma|X(U_{rd})|^2} \tag{5.46b}$$

(5.46a) is satisfied as long as constraint (5.10) holds. Combining (5.46b) and (5.44), with μ defined in (5.45) yields the inequality:

$$\frac{(1-\eta)\left(U_{rd} - \sigma - V_{max}\right)}{\left(U_{rd} + \sigma + V_{max}\right)^2} < \frac{2\eta}{\sigma} \tag{5.47}$$

It is possible to show that $\eta \geq \frac{1}{5}$ is a sufficient condition so that (5.47) hold. Therefore, if $\eta \geq \frac{1}{5}$, then μ, defined in (5.45), satisfies (5.46b). Therefore, without loss of generality, η is set as $\eta = \frac{1}{5}$. Hence, W_1 is positive definite.

To guarantee positive definitiveness of W_2, β and α must fulfill $\beta > 0$ and $\alpha > 1$. Assumption 5.3 and (5.11) guarantees that β fulfills $\beta > 0$. It is straightforward to show that (5.10) and (5.11) guarantees $\alpha > 1$. Hence, W_2 is positive definite. Further, $\alpha > 1$ guarantees $\mu > 0$, which ensures positive definitiveness of V.

Since V is positive, time-invariant and radially unbounded, and since $\dot{V}$ is negative definite the origin of the system (5.34) is UGAS.

The inequality $W \triangleq W_1 + W_1 \geq \bar{\lambda}_1 \bar{e}_1^2 + \bar{\lambda}_2 \bar{e}_2^2 + \lambda_3 \bar{v}_r^2$, for some constants $\bar{\lambda}_1, \bar{\lambda}_2, \lambda_3 > 0$, holds in a neighborhood of the origin. Thus, in any ball $\mathcal{B}_r \triangleq \{|e_2| \leq r\}, r > 0$ the function W can be bounded as $W \geq \lambda_1 e_1^2 + \lambda_2 e_2^2 + \lambda_3 \bar{v}_r^2$, where the constant λ_1 and λ_2 are given as $\lambda_i = \frac{\bar{\lambda}_i}{\left(r + \sigma \bar{E}_{int}^{eq}\right)^2 + \Delta^2}, i \in \{1, 2\}$. Hence, in any ball $\mathcal{B}_r$, $\dot{V}$ fulfills:

$$\dot{V}(\boldsymbol{x}) \leq -k_3 \left\|\boldsymbol{x}\right\|^2 \tag{5.48}$$

for $k_3 = \min\{\lambda_1, \lambda_2, \lambda_3\}$ and $\boldsymbol{x} = \begin{bmatrix} e_1 & e_2 & \bar{v}_r \end{bmatrix}^T$. Since V is a quadratic function of e_1, e_2 and $\bar{v}_r$, it fulfills:

$$k_1 \left\|\boldsymbol{x}\right\|^2 \leq V(\boldsymbol{x}) \leq k_2 \left\|\boldsymbol{x}\right\|^2 \tag{5.49}$$

for $k_1 = \min\left\{\frac{1}{2}\sigma^2, \frac{1}{2}, \frac{1}{2}\mu\right\}$ and $k_2 = \max\left\{\frac{1}{2}\sigma^2, \frac{1}{2}, \frac{1}{2}\mu\right\}$. Hence, by [Khalil, 2002, Theorem 4.10], the origin of the system (5.34) is ULES.

Chapter 6

Simulation results

6.1 Test cases

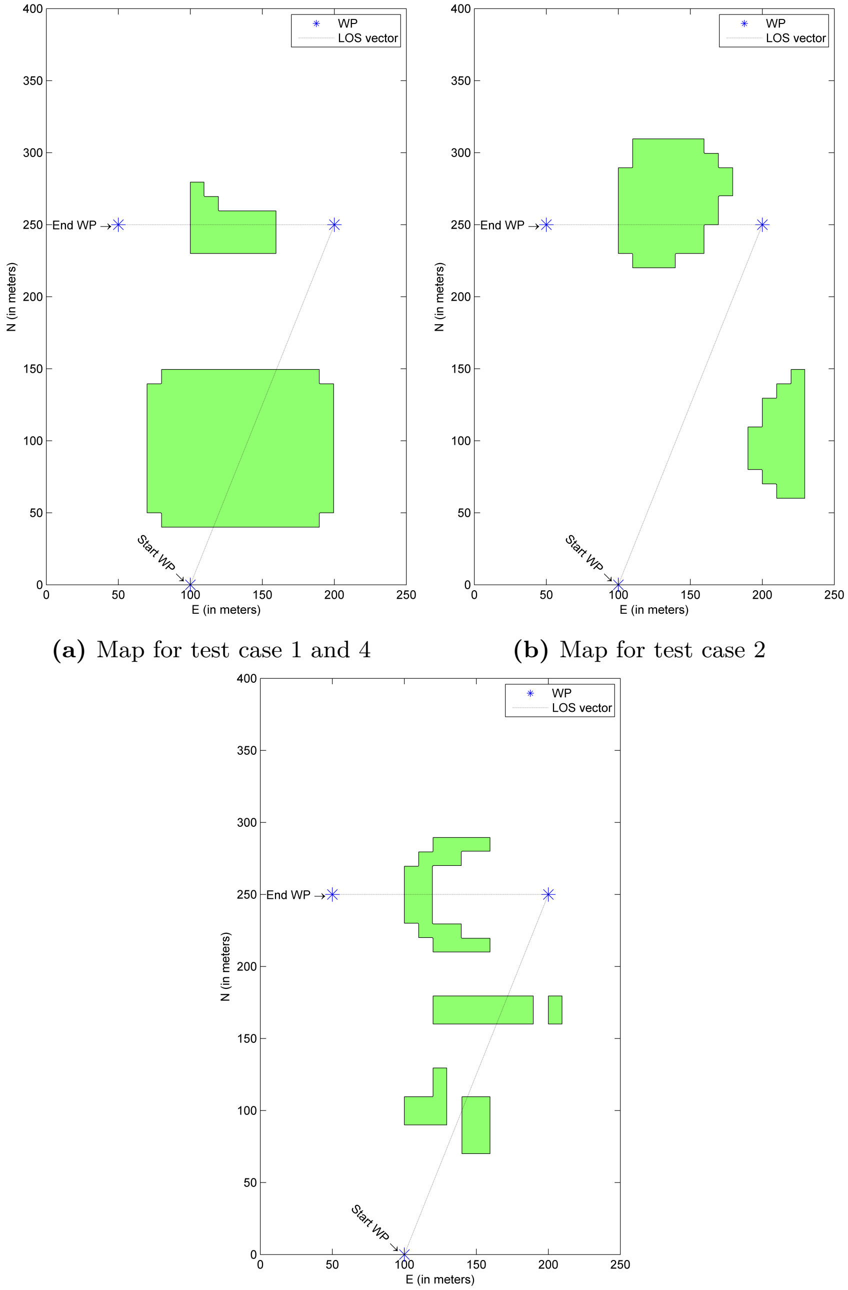
In order to evaluate the collision avoidance system performance, a number of evaluation points are defined:

- Response to ocean currents
- Convergence to the LOS vector
- Behavior when faced with local minima
- Minimum obstacle clearance
- Response in cluttered and complicated environments

To cover the points, four test cases are defined. To evaluate the modifications made to the algorithm, some simulations are compared with the ones in [Eriksen, 2014]. As described in section 2.3.2, the Dynamic window implementation in [Eriksen, 2014] filtered the objective function using a 2D FIR filter to improve the obstacle clearance. No other modifications were done to the algorithm, so the results from [Eriksen, 2014] are suited as a reference for evaluating the performance of the modifications done in this thesis.

Test case 1 and 2 uses maps equal to those used in [Eriksen, 2014]. Test case 1 contains two obstacles, which are detected fairly late by the AUV. This puts the responsiveness of the algorithm to the test. Test case 2 contains one obstacle which is detected right after a turn. The AUV travels freely for a long while, which makes it possible to evaluate the convergence to the LOS vector. Maps of the environments for the two cases are shown in figure 6.1a and 6.1b.

To compare the performance of the system with the results in [Eriksen, 2014], these maps are simulated both with and without ocean current. In addition,



(a) Map for test case 1 and 4

(b) Map for test case 2

(c) Map for test case 3

Figure 6.1: Simulation environments

simulations are also done with current while the Dynamic window algorithm assumes that no current is present. This tests the robustness of the algorithm with respect to knowledge of the ocean current.

Test case 3 exposes the AUV to tight corridors and a local minimum on the path. The system response is evaluated both with and without ocean current. A map of the environment is shown in figure 6.1c.

Test case 4 tests the exactness of the trajectory prediction in the Dynamic window algorithm. The sample time of the Dynamic window algorithm is increased to 5 seconds. To avoid oscillatory behavior, ocean current and the ILOS integrator are switched off. The same map as in test case 1 is used, again shown in figure 6.1a.

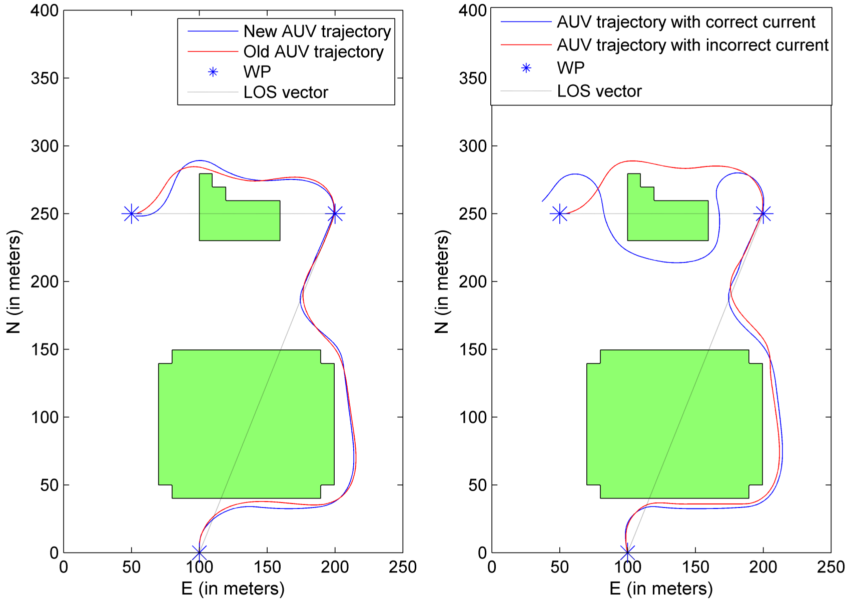
To test the robustness of the algorithm, 150 simulations in randomly generated environments are also conducted.

For test case 1, 2 and 3, video illustrations have been made. The videos are attached to the report .zip file, and are also available at `https://goo.gl/O8ZiEM`.

Unless otherwise is noted in the specific test cases, the simulation parameters are chosen as in table 6.1. Note especially the size of the avoidance and antitarget regions, set as 6 m and 3.5 m respectively. When the AUV model is defined in the pivot point the maximum radius of the AUV is about 3 m, which leaves a safety margin of 0.5 m.

Table 6.1 Simulation parameters

Parameter	**Value**	**Comment**
k_u	$1\ \mathrm{s}^{-1}$	Surge controller gain
k_r	$1\ \mathrm{s}^{-1}$	Yaw rate controller gain
k_ψ	$0.2\ \mathrm{s}^{-1}$	Yaw controller gain
Δ	8 m	ILOS lookahead distance
σ	0.4 m	ILOS integral gain
α	1	Dynamic window yaw rate scaling constant
β	$9\ \mathrm{s}^{-1}$	Dynamic window distance scaling constant
γ	3	Dynamic window surge speed scaling constant
U_d	$2\ \mathrm{m\,s}^{-1}$	Desired surge speed
ΔT_{DWA}	1 s	Dynamic window algorithm sample time
$\boldsymbol{V}_c$	$\begin{bmatrix}0.4 & -0.2\end{bmatrix}^T\ \mathrm{m\,s}^{-1}$	Ocean current, where applicable
$\bar{r}$	6 m	Size of the avoidance region Ω
r^*	3.5 m	Size of the antitarget region $\mathcal{T}$



(a) Old and new AUV trajectories

(b) AUV trajectories with correct and incorrect current information given to the Dynamic window algorithm

Figure 6.2: AUV trajectories for test case 1

6.2 Test case 1

The trajectories for test case 1 are shown in figure 6.2. Figure 6.2a contains the trajectories of the new and old Dynamic window implementations, and is commented further in section 6.2.1. The simulation videos *Case1_New.mp4* and *Case1_Old.mp4* illustrate the trajectories. Figure 6.2b show trajectories when the AUV is exposed to ocean currents. The Dynamic window algorithm is both given correct and wrong ocean current information. These trajectories are further commented in section 6.2.2. The simulation videos *Case1_CorrectCurrent.mp4* and *Case1_IncorrectCurrent.mp4* illustrate the trajectories.

6.2.1 Comparison of the old and new Dynamic window implementation

Figure 6.2a show the AUV trajectory using the new Dynamic window implementation, and the old one from [Eriksen, 2014]. Both the implementations choose roughly the same path through the environment, but the new implementation achieves a larger obstacle clearance.

From table 6.2 it is clear that the new implementation uses slightly more time to reach the last waypoint, but the average surge speed is higher. This points in the direction that the larger obstacle clearance demands a slightly longer path, which takes more time to complete. Obstacle clearance is however more important than a short path, therefore the new implementation is favored.

Table 6.2 Trajectory data, case 1, old and new Dynamic window implementation

Parameter	**Old implementation**	**New implementation**
Trajectory length to end WP	533 m	539 m
Trajectory time to end WP	269 s	273 s
Average surge speed	1.94 $\mathrm{m\,s^{-1}}$	1.95 $\mathrm{m\,s^{-1}}$
Minimum obstacle clearance	1.5 m	5.9 m

From figure 6.3 it is clear that both the surge and yaw rate controllers in the new implementation track their desired values well. The desired yaw rate in the new implementation is much smoother than the old one (see figure 6.4). This is probably caused by the focusing of the new objective function, which passes the reference from the yaw controller straight through the algorithm when no obstacles are present. It is also worth noting that old implementation have a large error in the yaw rate at the start. This is caused by the AUVs limited ability to actuate yaw at low surge velocities, which was not taken into account in [Eriksen, 2014]. The new search space includes time varying acceleration limits and hence accounts for this, as one can see in figure 6.3.

From figure 6.5 it is clear that the new implementation seldom saturates the actuators, except for the propeller at the start of the trajectory. This is in contrast to the old implementation, which also saturates the rudder at the start of the trajectory (see figure 6.6). This is again a result of the lack of modeling of the AUVs limited actuation in yaw at low surge velocities in the old implementation. Further, the rudder usage is more noisy in the new implementation. This is due to the feed forward of the desired yaw rate acceleration in the new yaw rate controller. In a practical implementation, some sort of filtering would be applied to the yaw control loop to reduce the noise in the rudder command signal.

6.2.2 Performance with ocean current

The first trajectory in figure 6.2b is generated by providing the Dynamic window algorithm with the correct ocean current, $\boldsymbol{V}_c = \begin{bmatrix}0.4 & -0.2\end{bmatrix}^T \ \mathrm{m\,s^{-1}}$. The second trajectory is generated by giving the Dynamic window algorithm no information about the ocean current, hence it assumes that the ocean current is zero. The

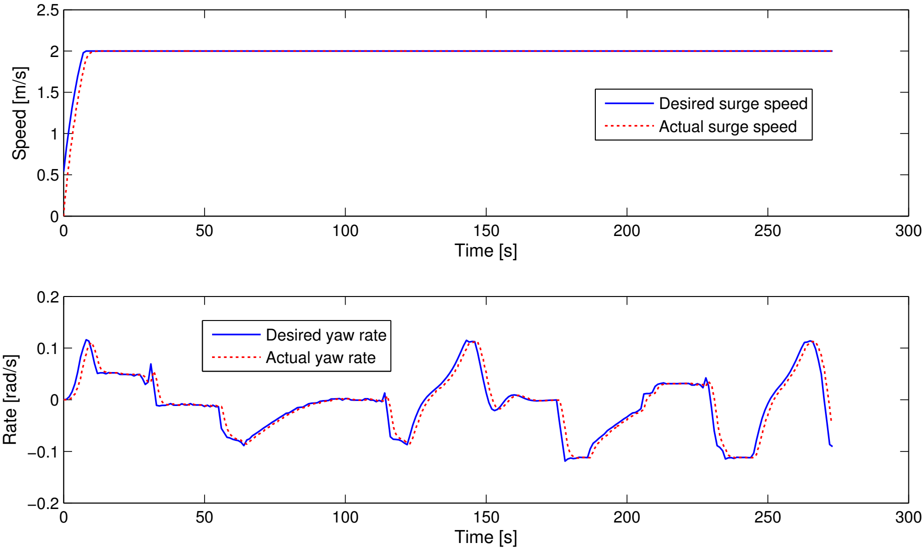


Figure 6.3: Desired and actual surge speed and yaw rate, case 1, new Dynamic window implementation

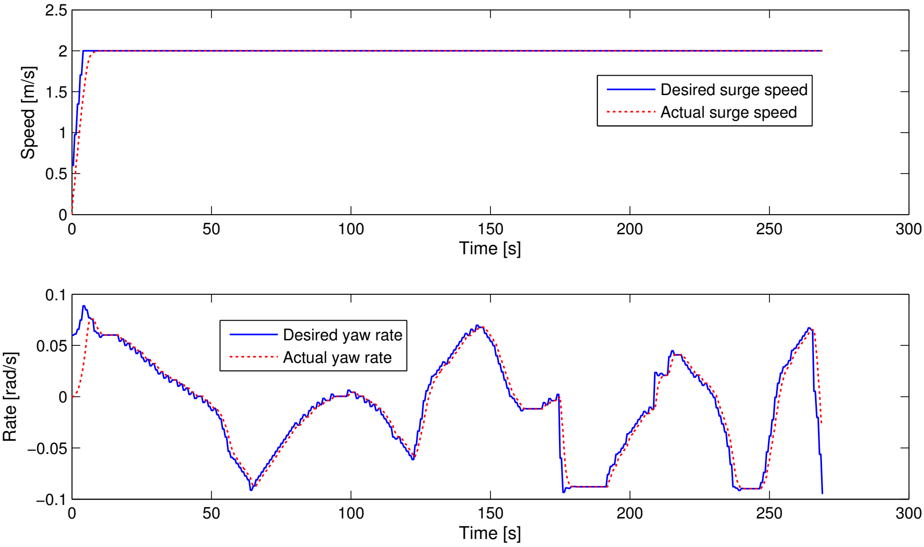


Figure 6.4: Desired and actual surge speed and yaw rate, case 1, old Dynamic window implementation

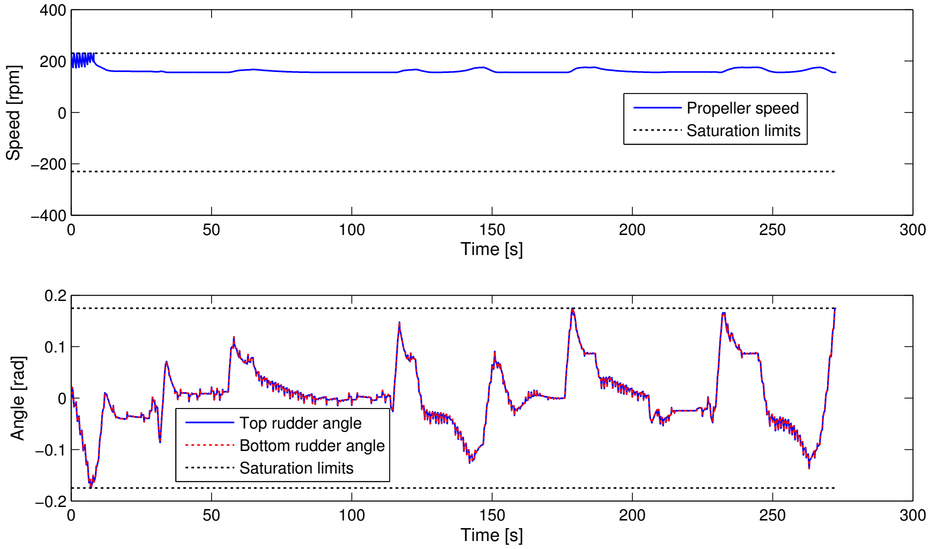


Figure 6.5: Actuator usage, case 1, new Dynamic window implementation

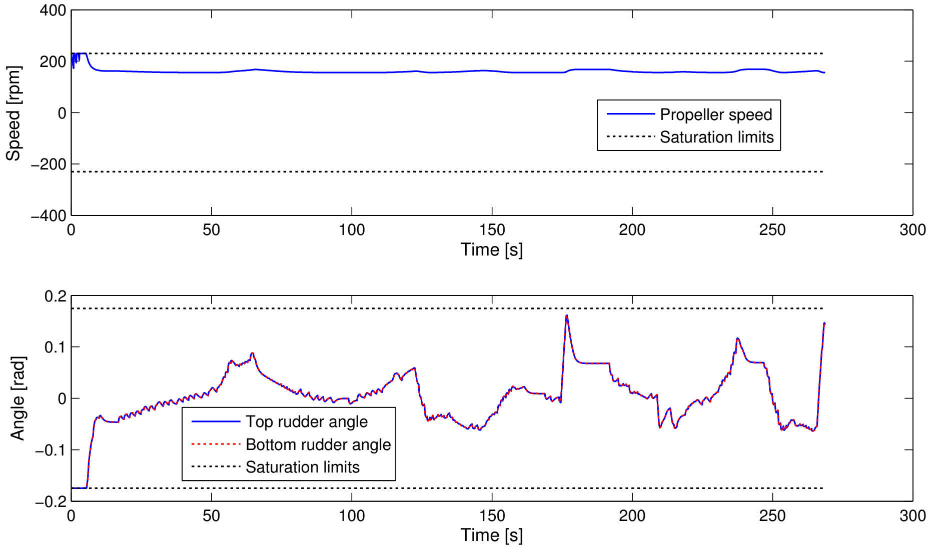


Figure 6.6: Actuator usage, case 1, old Dynamic window implementation

algorithm succeeds in avoiding collision in both cases, although the AUV travels closer to the obstacle boundary when it is provided with incorrect current information. Interestingly, the AUV reaches the end waypoint when it is given the incorrect current information, but not when it is given the correct current information. This is due to the short distance after the last obstacle to the end waypoint. The ocean current causes the ILOS guidance system to overshoot, and pass the waypoint before it reaches the circle of acceptance. By coincidence, the algorithm chooses a different route when given the incorrect current information, which makes it converge to the LOS vector faster and therefore reach the end waypoint.

The trajectories are summarized in table 6.3. Since the AUV does not reach the end waypoint when given the correct current information, the trajectory length and time is not computed for this. The obstacle clearance is satisfactory for both trajectories, 5.9 m and 3.3 m.

The controller responses and the actuator usage is presented in figures 6.7, 6.8, 6.9 and 6.10. It is worth noting that the initial surge speed is chosen lower when providing the Dynamic window algorithm with correct current information, since the predicted trajectories will be closer to the obstacle. This results in a noisy propeller speed. For a practical implementation, some sort of filtering would be introduced to the surge speed control loop to reduce this. It is also worth noting the rapidly changed propeller speed just past $t = 250$ s in figure 6.9. This is caused by the Dynamic window algorithm selecting a lower desired surge speed (see figure 6.7).

Table 6.3 Trajectory data, case 1, with ocean current

Parameter	**Correct current information**	**Incorrect current information**
Trajectory length to end WP	NA	542 m
Trajectory time to end WP	NA	256 s
Average surge speed	$1.91\ \mathrm{m\,s^{-1}}$	$1.95\ \mathrm{m\,s^{-1}}$
Minimum obstacle clearance	5.9 m	3.3 m

6.3 Test case 2

The trajectories for test case 2 are shown in figure 6.11. Figure 6.11a contains the trajectories of the new and old Dynamic window implementations, and is commented further in section 6.3.1. The simulation videos *Case2_New.mp4* and *Case2_Old.mp4* illustrate the trajectories. Figure 6.11b show trajectories when the AUV is exposed to ocean currents. As in test case 1, the Dynamic

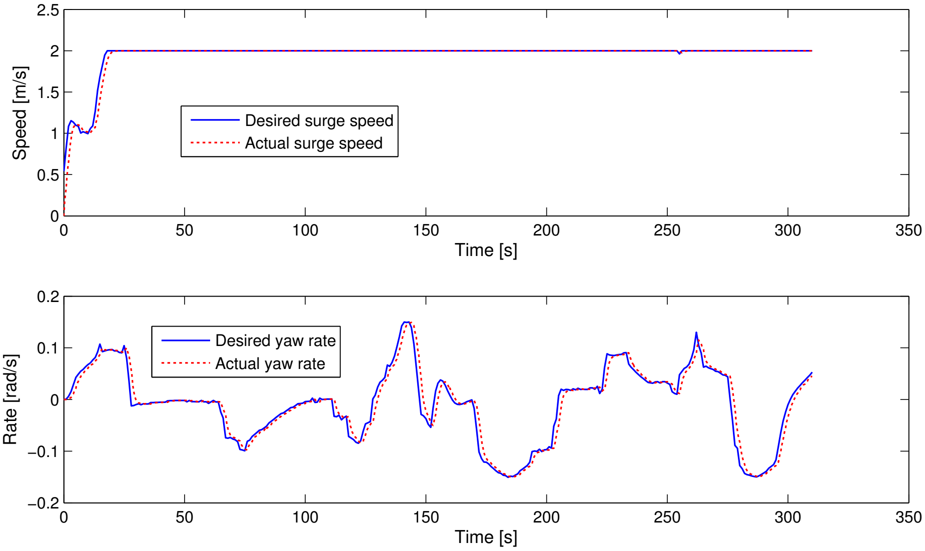


Figure 6.7: Desired and actual surge speed and yaw rate, case 1, correct current information

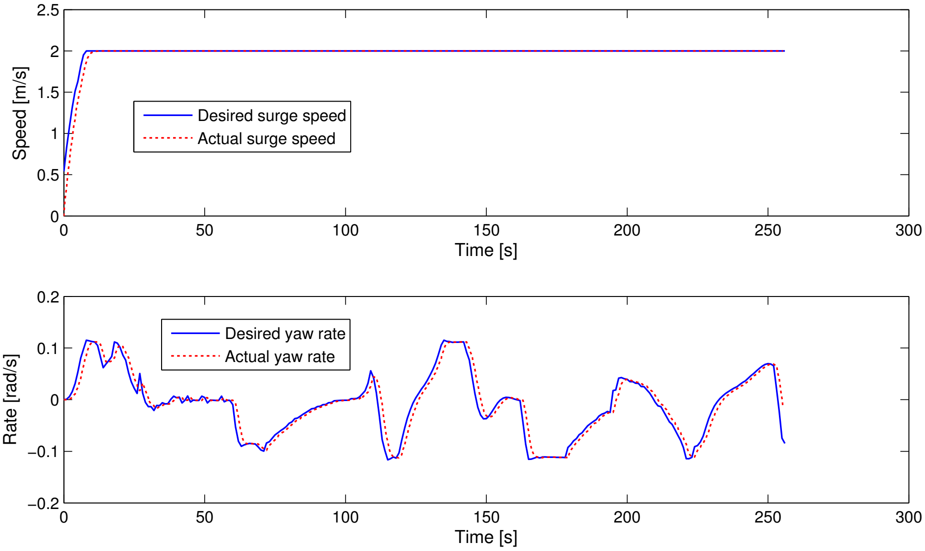


Figure 6.8: Desired and actual surge speed and yaw rate, case 1, incorrect current information

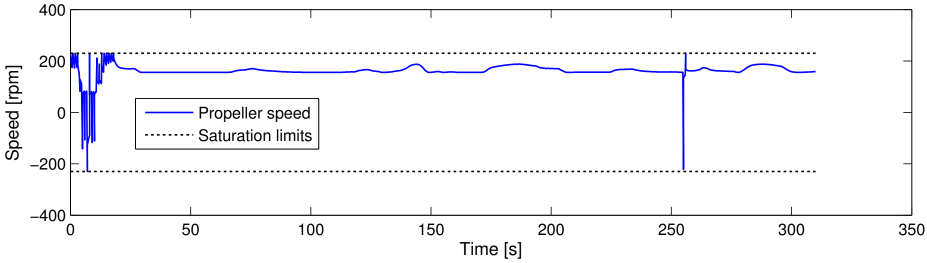


Figure 6.9: Actuator usage, case 1, correct current information

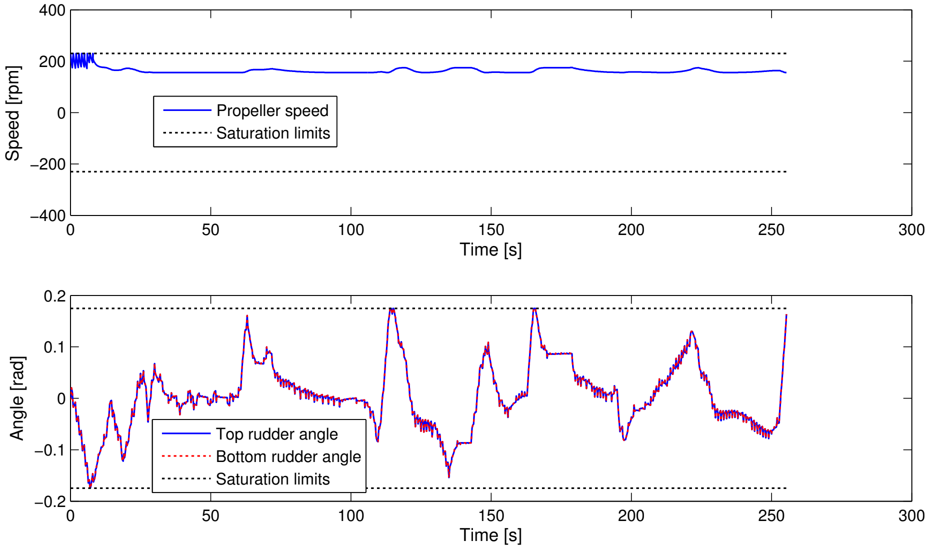
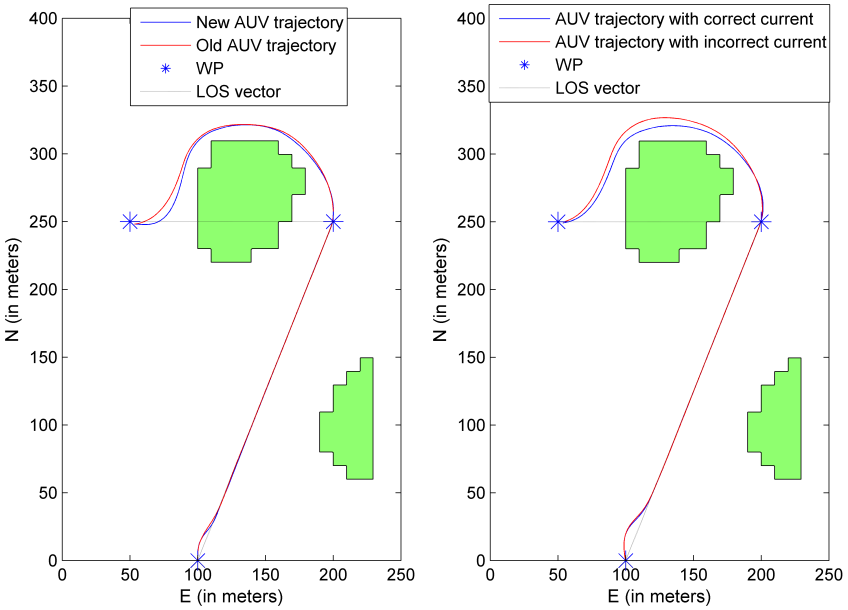


Figure 6.10: Actuator usage, case 1, incorrect current information



(a) Old and new AUV trajectories

(b) AUV trajectories with correct and incorrect current information given to the Dynamic window algorithm

Figure 6.11: AUV trajectories for test case 2

window algorithm is both given correct and wrong ocean current information. These trajectories are further commented in section 6.3.2. The simulation videos *Case2_CorrectCurrent.mp4* and *Case2_IncorrectCurrent.mp4* illustrate the trajectories.

6.3.1 Comparison of the old and new Dynamic window implementation

Figure 6.11a show the AUV trajectory using the new Dynamic window implementation, and the one in [Eriksen, 2014]. Both the implementations choose roughly the same path through the environment, but the new implementation achieves a more consistent obstacle clearance.

From table 6.4 it is difficult to distinct the trajectories. The only clear distinction is the difference in obstacle clearance. The new implementation steers clear of the antitarget region, and only slightly enters the avoidance region.

From figure 6.12 it is again clear that both the surge and yaw rate controllers in the new implementation tracks the desired values well and is smoother than the old one (see figure 6.13). Again, the large error in the yaw rate at the start with

Table 6.4 Trajectory data, case 2, old and new Dynamic window implementation

Parameter	**Old implementation**	**New implementation**
Trajectory length to end WP	500 m	502 m
Trajectory time to end WP	253 s	255 s
Average surge speed	1.94 $\mathrm{m\,s^{-1}}$	1.95 $\mathrm{m\,s^{-1}}$
Minimum obstacle clearance	6.9 m	5.8 m

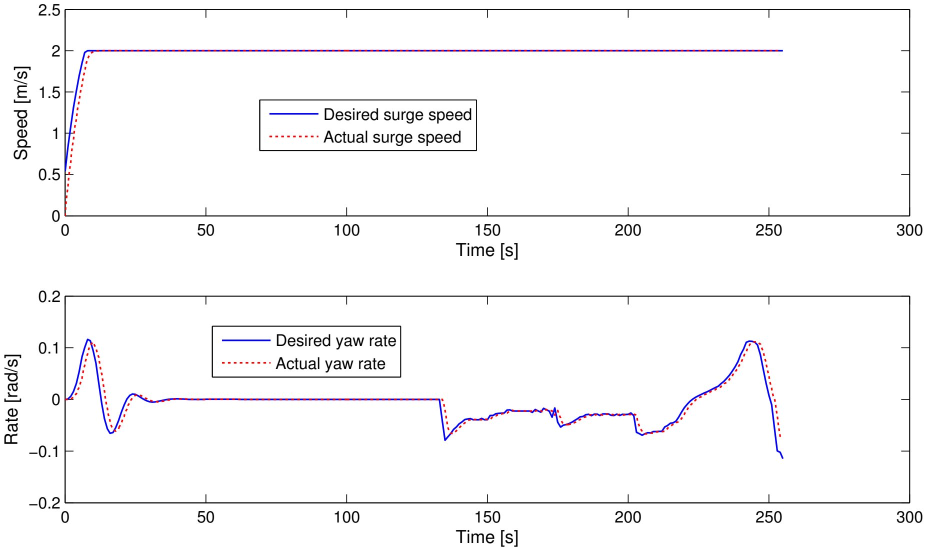


Figure 6.12: Desired and actual surge speed and yaw rate, case 2, new Dynamic window implementation

the old implementation is is caused by the AUVs limited ability to actuate yaw at low surge velocities, which was not taken into account in [Eriksen, 2014].

The actuator usage in figure 6.14 and 6.15 show the same results as in section 6.2.1. The new implementation saturates the actuators less than the old implementation, caused by the improved search space in the new implementation. On the other hand, the rudder usage is more noisy in the new implementation. This is again due to the feed forward of the desired yaw rate acceleration in the new yaw rate controller.

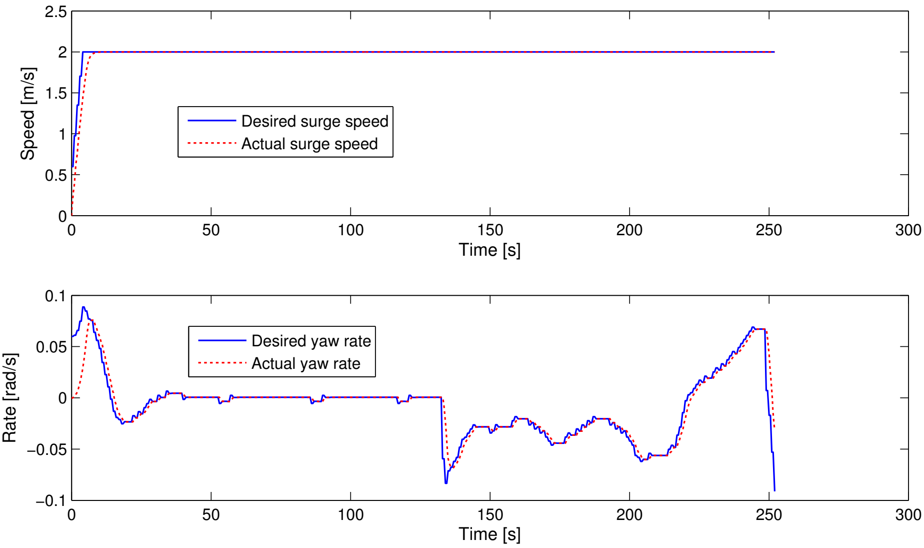


Figure 6.13: Desired and actual surge speed and yaw rate, case 2, old Dynamic window implementation

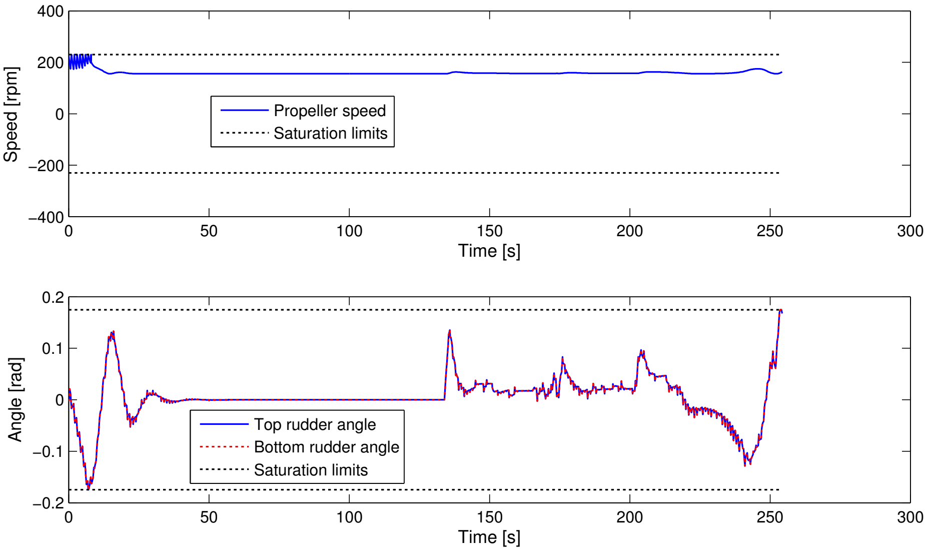


Figure 6.14: Actuator usage, case 2, new Dynamic window implementation

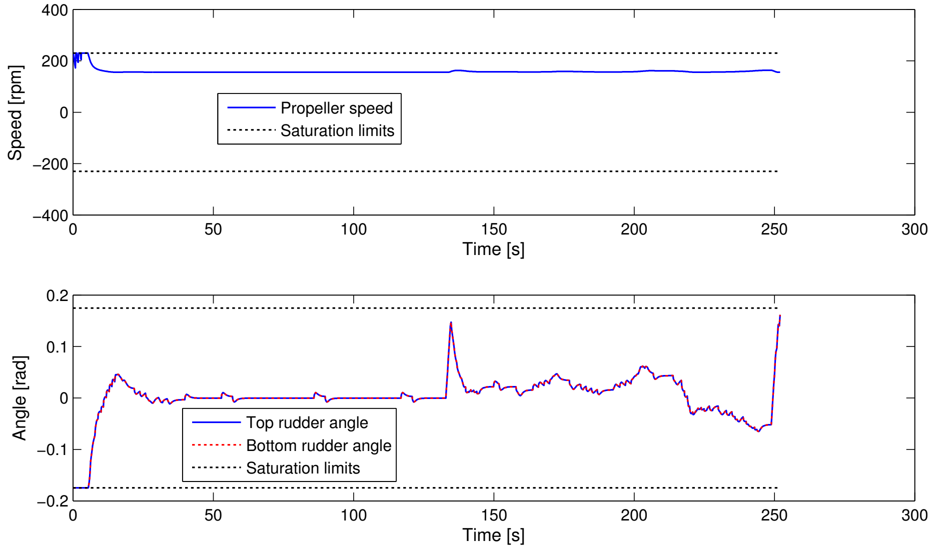


Figure 6.15: Actuator usage, case 2, old Dynamic window implementation

6.3.2 Performance with ocean current

As in test case 1, the first trajectory in figure 6.11b is generated by providing the Dynamic window algorithm with the correct ocean current information, $\boldsymbol{V}_c = \begin{bmatrix}0.4 & -0.2\end{bmatrix}^T$ $\mathrm{m\,s^{-1}}$. The second trajectory is generated by giving the Dynamic window algorithm no information about the ocean current, hence it assumes that the ocean current is zero. Both the trajectories reach the end waypoint, and avoid collision. The trajectories are very similar, but the AUV is driven a bit farther away from the LOS vector than necessary when the Dynamic window algorithm is unaware of the ocean current. This is verified by the trajectory data in table 6.5, where the minimum obstacle clearance is 9.3 m when the Dynamic window algorithm is unaware of the current and 5.8 m when it is provided with the correct ocean current.

Table 6.5 Trajectory data, case 2, with ocean current

Parameter	**Correct current information**	**Incorrect current information**
Trajectory length to end WP	500 m	507 m
Trajectory time to end WP	234 s	237 s
Average surge speed	1.95 $\mathrm{m\,s^{-1}}$	1.95 $\mathrm{m\,s^{-1}}$
Minimum obstacle clearance	5.8 m	9.3 m

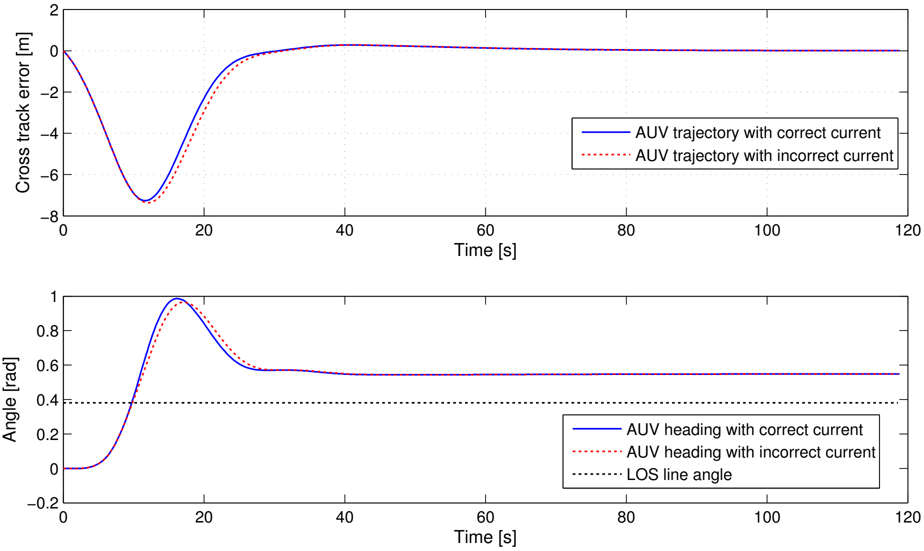


Figure 6.16: Cross track error and AUV heading, case 2, with correct and wrong current information

Figure 6.16 show the cross track error and the AUV heading angle, for both trajectories. Notice that the cross track error converges slightly faster to zero when correct current information is passed to the Dynamic window algorithm. The AUV heading converges to a value unequal to the path angle, which means the AUV side-slips to achieve zero cross track error. This coincides with the ILOS theory, and the stability proof in chapter 5.

6.4 Test case 3

The trajectories for test case 3 is shown in figure 6.17, and illustrated in the simulation videos *Case3_Current.mp4* and *Case1_NoCurrent.mp4*. When no ocean current is applied, the Dynamic window algorithm ensures a clearance of 3.2 m from the obstacles. The AUV is however trapped in a local minimum. This is expected, since no global information is given to the Dynamic window algorithm. To escape from the local minimum, a new route must be planned which avoids the local minimum the AUV is trapped in. For instance, a global planning algorithm could calculate new waypoints and feed them to the ILOS guidance system.

When ocean current is introduced the AUV collides with the last obstacle. The collision occurs since the AUV enters a local minimum and stops. The ocean current then drives the AUV into the obstacle. No collision detection is implemented

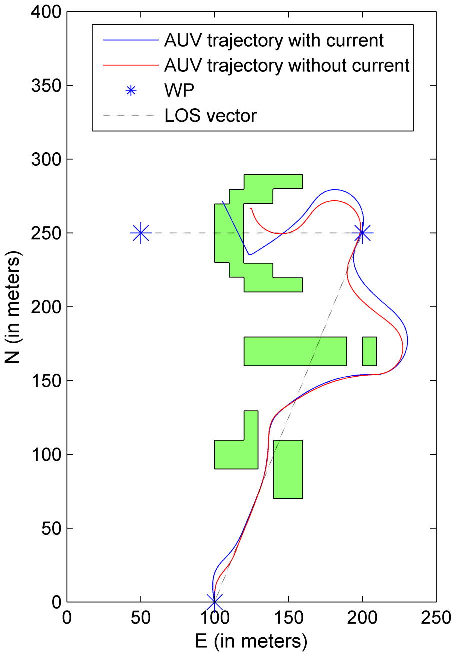


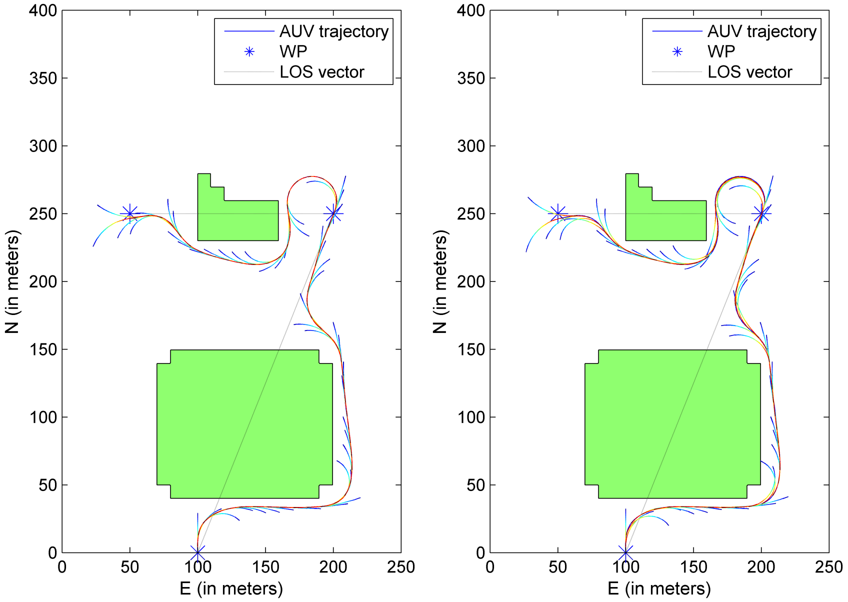
Figure 6.17: AUV trajectories, test case 3, with and without ocean currents

in the simulator, therefore it appears as if the AUV travels through the obstacle. It might be possible to avoid collision by allowing the relative surge speed to be negative, and hence back up against the current. This would require a model of the AUV when it is backing, and some sensing of the environment behind the AUV. Again, a new route would have to be calculated in order to escape from the local minimum.

The trajectories are summarized in table 6.6.

Table 6.6 Trajectory data, case 3

Parameter	**With ocean current**	**Without ocean current**
Trajectory length to end WP	NA	NA
Trajectory time to end WP	NA	NA
Average surge speed	1.51 $\mathrm{m\,s^{-1}}$	1.56 $\mathrm{m\,s^{-1}}$
Minimum obstacle clearance	0 m	3.2 m



(a) AUV trajectory, with the predicted trajectories using linear trajectory prediction in varying colors

(b) AUV trajectory, with the predicted trajectories using circular trajectory prediction in varying colors

Figure 6.18: AUV trajectories for test case 4

6.5 Test case 4

Test case 4 evaluates the exactness of the predicted AUV trajectories using the new linear trajectory prediction, and compares it to using circular trajectory prediction. For this test case, the Dynamic window algorithm sample time is increased to $\Delta T_{\text{DWA}} = 5$ s, so that the AUV travels a while before generating new trajectories (at 2 $\text{m}\,\text{s}^{-1}$ this corresponds to 10 m). Figure 6.18 show the AUV trajectory together with the predicted AUV trajectories in each time step. The AUV trajectories are predicted for 20 s, and plotted in a varying colormap with warm colors at the start of the trajectory and cold colors at the end of the trajectory. It should be noted that only the first 5 s of the predicted trajectories should be compared to the actual AUV trajectory. Figure 6.18a show the AUV trajectory, together with the predicted AUV trajectories using the linear prediction. Figure 6.18b show the same AUV trajectory, together with circular predictions of the AUV trajectories.

From the trajectories in figure 6.18 and the prediction error (see figure 6.19), it is clear that the predicted trajectories using the linear approximation is much more accurate than when using the circular prediction, especially when doing a fast maneuver (such as a sharp turn). Note that the prediction error is reset to zero

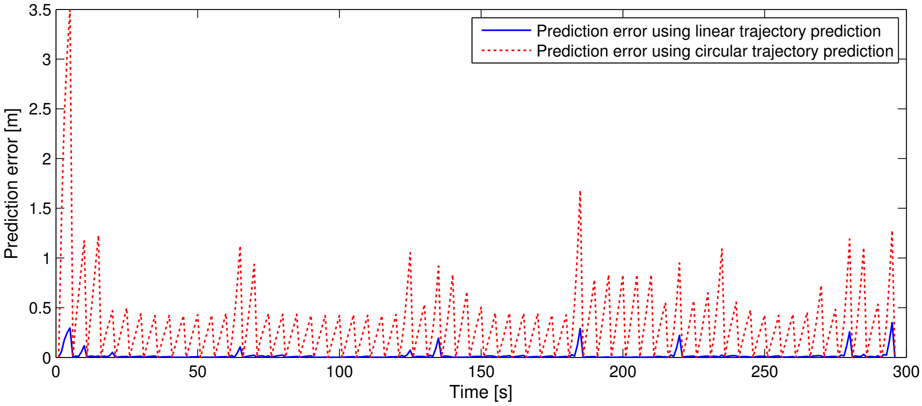


Figure 6.19: Prediction error using linear and circular trajectory prediction, case 4

every 5 s since the Dynamic window algorithm is run every 5 s. The mean square error, and root mean square error of the predicted AUV trajectories is shown in table 6.7. Table 6.7 gives a more correct view of the prediction error than the $t = [0, 5]$ s elements in table 3.1, since it is averaged over a simulation. Notice that the mean square error is reduced with a factor greater than 100 when using linear trajectory prediction instead of circular trajectory prediction.

Table 6.7 Prediction error of the predicted AUV trajectories, case 4

Parameter	**Linear approximation**	**Circular approximation**
Mean square error	0.00228 m^2	0.271 m^2
Root mean square error	0.0478 m	0.521 m

6.6 Simulations in randomly generated environments

To thoroughly test the robustness of the collision avoidance system, the AUV is simulated in 150 randomly generated environments. The simulations was done without ocean current and with the ILOS integrator switched off. As shown in table 6.8, 18 of the 150 simulations came closer than 3 m to an obstacle, and hence caused a collision. One trajectory even resulted in a minimum distance of 0.1 m. All the 150 simulations are presented in appendix B.

A closer inspection of the trajectories where collisions occur reveals that it takes a long time from the Dynamic window algorithm commands the AUV to stop

Table 6.8 Summary of trajectories in randomly generated environments

Minimum obstacle clearance	**Number of simulations**	**Of which reached the end waypoint**
$[0,1]$ m	4	0
$(1,2]$ m	4	0
$(2,3]$ m	10	0
$(3,4]$ m	55	1
$(4,5]$ m	9	1
$(5,6]$ m	64	36
$(6,\infty)$ m	4	2

(by specifying $u_{rd} = r_d = 0$), until the sway speed goes to zero. Therefore, the AUV moves sideways for a long time after it is commanded to stop. For one of the trajectories where a collision occurs, it takes about 50 s for the sway speed and yaw rate to passively converge to zero (see figure 6.20). During this time, the AUV moved 4.03 m sideways. This can also be seen by viewing the end of the video illustration of test case 3 without ocean current (*Case3_NoCurrent.mp4*), where the AUV gets trapped in a local minimum. At zero surge speed, both the sway and yaw dynamics are uncontrollable and is hence only affected by the vehicle dynamics. It seems unlikely that the AUV uses such long time to passively reach zero sway speed. Therefore it is concluded that the simulation model is inaccurate at low surge velocities, which is natural since the damping of the system is scaled with the surge speed (see the AUV model in section 2.1).

Due to the inaccuracy at low surge velocities, the simulations where the AUV stops in a local minma should not be trusted unconditionally. 132 of the 150 simulations achieve a minimum distance to obstacles of more than 3 m, and 40 of the simulations reach the end waypoint. Seen in light of the model inaccuracy, this is considered as a good result. It should be noted that the minimum clearance for the trajectories where the AUV reached the end waypoint is 3.6 m. This indicates that the algorithm works well, and avoids entering the antitarget region, when not presented with any local minima (see appendix B).

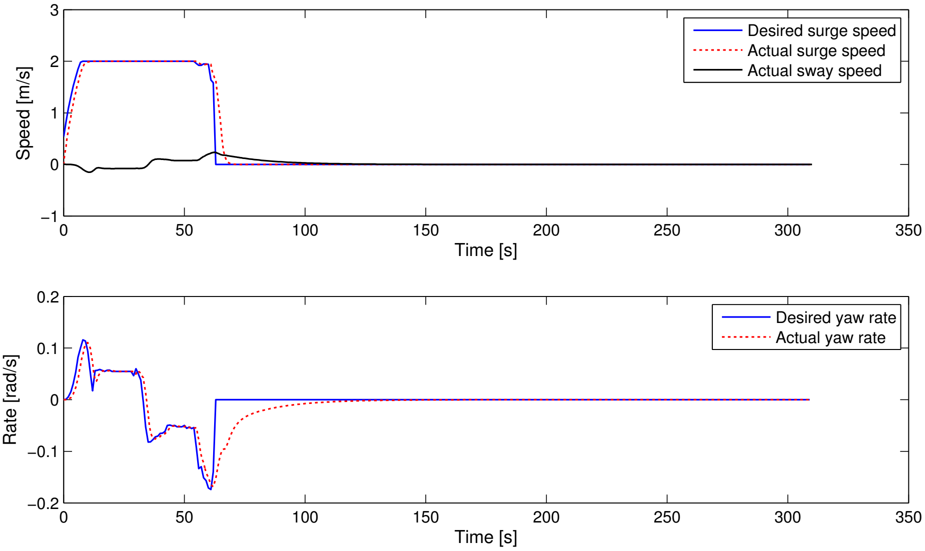


Figure 6.20: Passive response to sway and yaw dynamics

Chapter 7

Concluding remarks and suggestions for future work

A great deal of effort has been put into improving the Dynamic window algorithm, both in this thesis and by researchers before me. Despite this, to the authors knowledge, no modifications has previously been made to consider vehicles with second order non-holonomic constraints and time varying acceleration limits. This thesis presents a number of modifications to adapt the algorithm for use with such vehicles. In addition, the algorithm has been extended to account for ocean current.

A new method for predicting the AUV trajectory which takes account of second order non-holonomic constraints and ocean current has been developed, and the search space has been modified to account for the time varying acceleration limits of AUVs equipped with rudders for yaw actuation. The architecture has been changed to facilitate a more layered implementation where the Dynamic window algorithm takes a desired surge speed and yaw rate as input, and produces references for surge speed and yaw rate controllers. In order to control the obstacle clearance when circumnavigating obstacles, a local map based on sonar measurements is created and modified such that the AUV achieves a minimum desired obstacle clearance.

A simulator containing a 6DOF model of the HUGIN 1000 AUV, sonar sensors, controllers, a horizontal ILOS guidance system and the collision avoidance system has been developed. The simulator include functionality for simulating constant irrotational ocean current in the horizontal plane. The modified Dynamic window algorithm is compared to the implementation in [Eriksen, 2014], which is quite close to the original Dynamic window algorithm. Based on the simulations done it is clear that the modified Dynamic window algorithm is superior to the old implementation. In the simulated environments, the two implementations choose roughly the same paths, and both avoid collision. However, the new implementation achieves a more consistent and controllable obstacle clearance and produces references with a higher degree of feasibility. The new algorithm can also compensate for constant irrotational ocean current. The new method for predicting the AUV trajectory reduces the MSE of the predictions to about one percent of the original method, and makes the algorithm well suited for vehicles with second order non-holonomic constraints.

The simulations show that the modified Dynamic window algorithm succeeds in avoiding collision when it is not trapped in a local minimum, both when under influence of ocean currents and when not. The simulation model is inaccurate at low surge velocities, therefore it is difficult to assess the performance when the AUV is caused to stop due to local minima. However, closer inspections of the simulations suggest that the algorithm is able to avoid collisions in local minima when no ocean current is present. This should however be investigated further. If faced with ocean current and the AUV is forced to stop in a local minimum, the ocean current may drive the AUV into obstacles and hence cause a collision. It may be possible to extend the algorithm to counteract this.

To test the robustness of the system, 150 simulations were carried out in randomly generated environments. 132 of these simulations achieved a minimum clearance to obstacles greater than 3 m. Due to the modeling inaccuracies at low surge velocities the AUV slides sideways a long time after the surge speed reaches zero. As a result, the AUV sometimes slide sideways into obstacles when trapped in local minima. Therefore, the simulations where the AUV gets trapped in a local minimum should not be trusted unconditionally. The simulations infer that the Dynamic window algorithm is well suited for horizontal collision avoidance for AUVs, but the simulation model must be improved in order to verify the performance and robustness of the algorithm.

Convergence to the LOS vector, when the AUV is under influence of ocean current, is guaranteed through a Lyapunov based proof, given that no obstacles are present. The simulations support the stability analysis.

As expected, the AUV is prone to getting trapped in local minima. The Dynamic window algorithm is a reactive collision avoidance algorithm, which does not assume knowledge of any global information. Hence, global convergence can not be guaranteed. By carefully tuning the objective function, local minima may often be avoided. However, this can never guarantee global convergence to a goal and limited effort has therefore been put into tuning the objective function. Any practical implementation would need to implement some sort of global path planning running in parallel with the Dynamic window algorithm, to make the AUV avoid local minima in the environment.

Undoubtedly, there are still issues to be solved before the Dynamic window algorithm can be considered as a robust and suitable collision avoidance algorithm for practical implementation on the HUGIN 1000 AUV. The following topics are therefore suggested to continue the work of this thesis:

- Improve the simulation model to correctly model the AUV behavior at low surge velocities.
- Investigate the robustness of the algorithm with respect to model uncertainties.
- Investigate possible modifications to avoid drifting into obstacles when trapped in local minima and influenced by ocean currents.
- Implement some sort of global path planning to avoid and, if trapped, escape from local minima.
- Investigate if adding a cost to the velocity pairs based on how much of the predicted trajectory resides inside the avoid and antitarget regions, as an alternative to the distance term, improves the performance.
- Experiment with a more predictive approach, for example by using an MPC inspired approach as in [Ögren and Leonard, 2002].

Appendices

Appendix A

Stability of cascaded systems

Given a time-varying nonlinear cascaded system:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_1(t,\boldsymbol{x}) + \boldsymbol{g}(t,\boldsymbol{x},\boldsymbol{y}) \tag{A.1a}$$

$$\dot{\boldsymbol{y}} = \boldsymbol{f}_2(t,\boldsymbol{y}) \tag{A.1b}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{y} \in \mathbb{R}^m$. The functions $\boldsymbol{f}_1(\cdot,\cdot)$ and $\boldsymbol{f}_2(\cdot,\cdot)$ are continuously differentiable in their arguments. The stability properties of the origin $(\boldsymbol{x},\boldsymbol{y}) = (\mathbf{0},\mathbf{0})$ of the cascaded system (A.1) is given by

Theorem A.1. *[Panteley and Loria, 1998][Theorem 2]. Given the cascaded system* (A.1)*. Assume that the nominal system of* (A.1a)*, given as $\dot{\boldsymbol{x}} = \boldsymbol{f}_1(t,\boldsymbol{x})$, is UGAS with a Lyapunov function $V(\boldsymbol{x},t)$ satisfying:*

$$\left\| \frac{\partial V}{\partial \boldsymbol{x}} \right\| \|\boldsymbol{x}\| \leq c_1 V(\boldsymbol{x},t), \quad \forall \, \|\boldsymbol{x}\| \geq \eta \tag{A.2}$$

where $c_1, \eta > 0$. If, in addition, the function $\boldsymbol{g}(t,\boldsymbol{x},\boldsymbol{y})$ satisfies:

$$\|\boldsymbol{g}(t,\boldsymbol{x},\boldsymbol{y})\| \leq \theta_1 \left(\|\boldsymbol{y}\|\right) + \theta_2 \left(\|\boldsymbol{y}\|\right) \|\boldsymbol{x}\|, \quad \theta_1, \theta_2 : \mathbb{R}^+ \to \mathbb{R}^+ \quad C^0 \tag{A.3}$$

and, the system (A.1b)*, given as $\dot{\boldsymbol{y}} = \boldsymbol{f}_2(t,\boldsymbol{y})$, is UGAS and, $\forall t_0 \geq 0$:*

$$\int_{t_0}^{t} \|\boldsymbol{y}(s,t_0,\boldsymbol{y}(t_0))\| \, \mathrm{d}s \leq \phi \left(\|\boldsymbol{y}(t_0)\|\right) \tag{A.4}$$

where $\phi(\cdot)$ is a class $\mathcal{K}$ function, then, the cascaded system (A.1) *is UGAS.*

Definition A.1. *[Khalil, 2002][Definition 4.2]. A continuous function $\alpha : [0,a) \to [0,\infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$.*

Remark A.1. *If the nominal system of* (A.1a)*, given as* $\dot{\boldsymbol{x}} = \boldsymbol{f}_1(t, \boldsymbol{x})$ *is UGAS with a quadratic Lyapunov function, then the condition* (A.2) *is satisfied trivially [Caharija, 2014].*

Remark A.2. *If the perturbing system* (A.1b)*, given as* $\dot{\boldsymbol{y}} = \boldsymbol{f}_2(t, \boldsymbol{y})$ *is UGAS and ULES (or equivalently exponentially stable in any ball of initial conditions), then the condition* (A.4) *is satisfied trivially [Caharija, 2014].*

Lemma A.1. *[Panteley et al., 1998][Lemma 8], [Caharija, 2014][Lemma A.2]. Assume that, in addition to the requirements made in Theorem A.1, both the nominal system of* (A.1a)*, and* (A.1b)*, given as* $\dot{\boldsymbol{x}} = \boldsymbol{f}_1(t, \boldsymbol{x})$ *and* $\dot{\boldsymbol{y}} = \boldsymbol{f}_2(t, \boldsymbol{y})$ *respectively, are globally* κ*-exponentially stable (exponentially stable in any ball of initial conditions). Then. the cascaded system* (A.1) *is globally* κ*-exponentially stable.*

Appendix B

Simulations in randomly generated environments

This appendix presents 150 simulations in randomly generated environments. The trajectories are sorted by the minimum obstacle clearance and if it reached the end waypoint. Key information about the trajectories are presented in table B.1, while the trajectories and the environments are shown in figure B.1 to B.150.

Table B.1: Trajectory data for simulations in randomly generated environments

Figure nr	**Min clearance**	**End WP reached**
B.1	11.3 m	Yes
B.2	10.0 m	Yes
B.3	6.1 m	No
B.4	6.1 m	No
B.5	6.0 m	Yes
B.6	6.0 m	Yes
B.7	6.0 m	Yes
B.8	6.0 m	Yes
B.9	6.0 m	Yes
B.10	6.0 m	Yes
B.11	6.0 m	Yes
B.12	6.0 m	Yes
B.13	6.0 m	Yes
B.14	6.0 m	Yes
B.15	6.0 m	Yes
B.16	6.0 m	No
B.17	6.0 m	No
B.18	6.0 m	No
B.19	5.9 m	Yes
B.20	5.9 m	Yes
B.21	5.9 m	Yes
B.22	5.9 m	Yes
B.23	5.9 m	Yes
B.24	5.9 m	Yes
B.25	5.9 m	Yes
B.26	5.9 m	Yes
B.27	5.9 m	Yes
B.28	5.9 m	Yes
B.29	5.9 m	No
B.30	5.9 m	No
B.31	5.9 m	No
B.32	5.9 m	No
B.33	5.9 m	No
B.34	5.8 m	Yes
B.35	5.8 m	Yes
B.36	5.8 m	Yes
B.37	5.8 m	Yes
B.38	5.8 m	Yes
B.39	5.8 m	No
B.40	5.8 m	No
B.41	5.8 m	No
B.42	5.8 m	No
B.43	5.8 m	No
B.44	5.8 m	No
B.45	5.8 m	No
B.46	5.7 m	Yes
B.47	5.7 m	Yes
B.48	5.7 m	Yes
B.49	5.7 m	Yes
B.50	5.7 m	Yes
B.51	5.7 m	No
B.52	5.7 m	No
B.53	5.7 m	No
B.54	5.7 m	No
B.55	5.7 m	No
B.56	5.7 m	No
B.57	5.7 m	No
B.58	5.6 m	Yes
B.59	5.6 m	Yes
B.60	5.6 m	Yes
B.61	5.6 m	No
B.62	5.6 m	No
B.63	5.6 m	No
B.64	5.6 m	No
B.65	5.5 m	No
B.66	5.3 m	Yes
B.67	5.2 m	Yes
B.68	5.2 m	No
B.69	5.0 m	Yes
B.70	4.9 m	No
B.71	4.9 m	No
B.72	4.8 m	No
B.73	4.7 m	No
B.74	4.7 m	No
B.75	4.5 m	No
B.76	4.2 m	No

continued on next page

continued from previous page

continued on next page

continued from previous page

continued from previous page

Figure nr	Min clearance	End WP reached
B.77	4.1 m	No
B.78	4.0 m	No
B.79	4.0 m	No
B.80	3.9 m	No
B.81	3.7 m	No
B.82	3.6 m	Yes
B.83	3.6 m	No
B.84	3.6 m	No
B.85	3.5 m	No
B.86	3.5 m	No
B.87	3.5 m	No
B.88	3.5 m	No
B.89	3.5 m	No
B.90	3.5 m	No
B.91	3.5 m	No
B.92	3.5 m	No
B.93	3.5 m	No
B.94	3.5 m	No
B.95	3.5 m	No
B.96	3.4 m	No
B.97	3.4 m	No
B.98	3.4 m	No
B.99	3.4 m	No
B.100	3.4 m	No
B.101	3.4 m	No
B.102	3.4 m	No
B.103	3.4 m	No
B.104	3.4 m	No
B.105	3.4 m	No
B.106	3.4 m	No
B.107	3.3 m	No
B.108	3.3 m	No
B.109	3.3 m	No
B.110	3.3 m	No
B.111	3.2 m	No
B.112	3.2 m	No
B.113	3.2 m	No
B.114	3.2 m	No
B.115	3.2 m	No
B.116	3.2 m	No
B.117	3.2 m	No
B.118	3.2 m	No
B.119	3.2 m	No
B.120	3.2 m	No
B.121	3.2 m	No
B.122	3.1 m	No
B.123	3.1 m	No
B.124	3.1 m	No
B.125	3.1 m	No
B.126	3.1 m	No
B.127	3.1 m	No
B.128	3.1 m	No
B.129	3.1 m	No
B.130	3.1 m	No
B.131	3.0 m	No
B.132	3.0 m	No
B.133	2.9 m	No
B.134	2.8 m	No
B.135	2.8 m	No
B.136	2.7 m	No
B.137	2.6 m	No
B.138	2.5 m	No
B.139	2.4 m	No
B.140	2.4 m	No
B.141	2.4 m	No
B.142	2.2 m	No
B.143	1.9 m	No
B.144	1.2 m	No
B.145	1.1 m	No
B.146	1.1 m	No
B.147	1.0 m	No
B.148	0.6 m	No
B.149	0.2 m	No
B.150	0.1 m	No

continued on next page

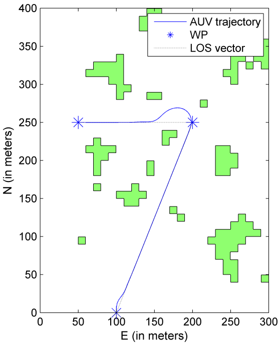


Figure B.1: Simulation nr. 1

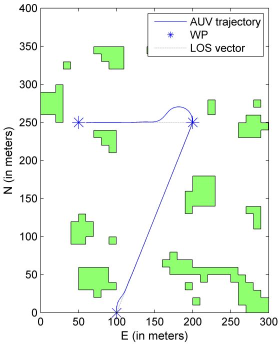


Figure B.2: Simulation nr. 2

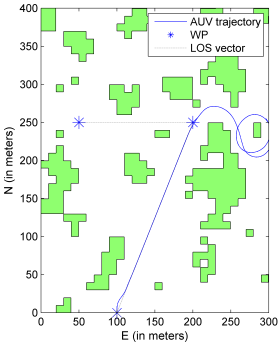


Figure B.3: Simulation nr. 3

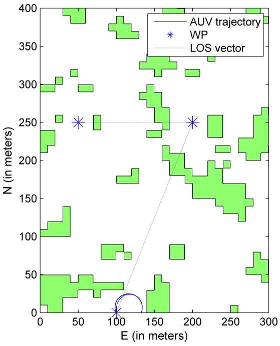


Figure B.4: Simulation nr. 4

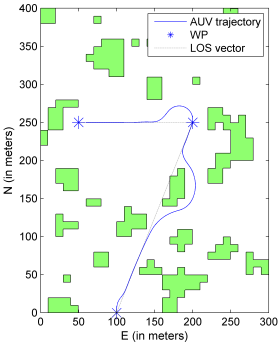


Figure B.5: Simulation nr. 5

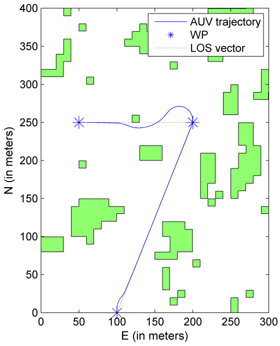


Figure B.6: Simulation nr. 6

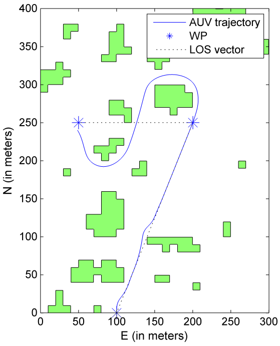


Figure B.7: Simulation nr. 7

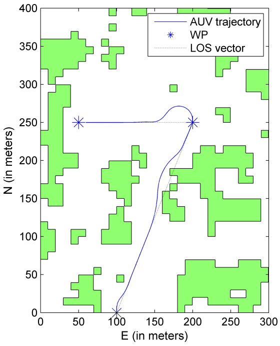


Figure B.8: Simulation nr. 8

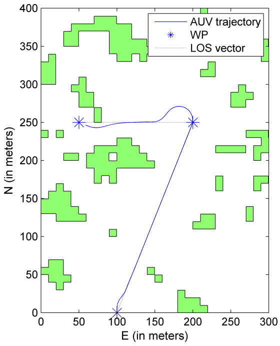


Figure B.9: Simulation nr. 9

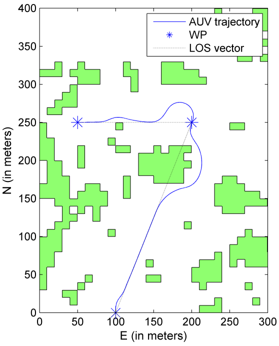


Figure B.10: Simulation nr. 10

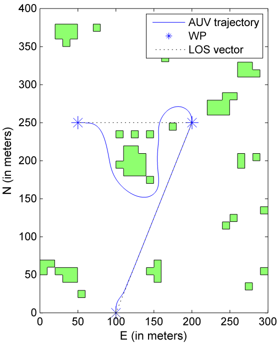


Figure B.11: Simulation nr. 11

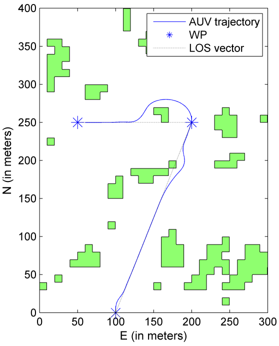


Figure B.12: Simulation nr. 12

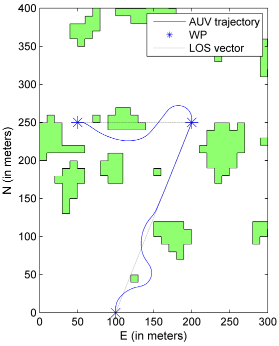


Figure B.13: Simulation nr. 13

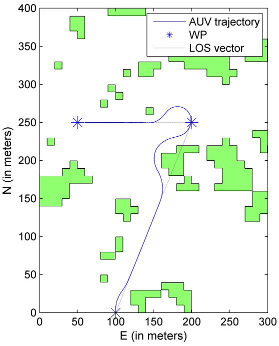


Figure B.14: Simulation nr. 14

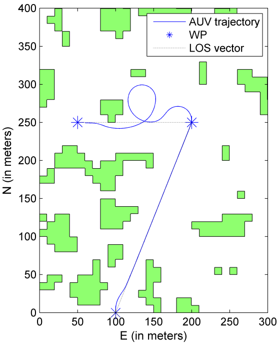


Figure B.15: Simulation nr. 15

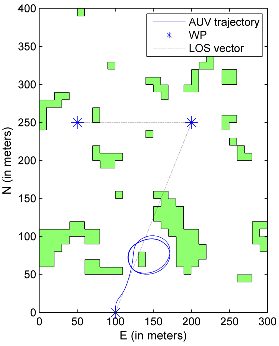


Figure B.16: Simulation nr. 16

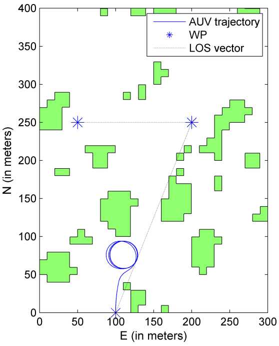


Figure B.17: Simulation nr. 17

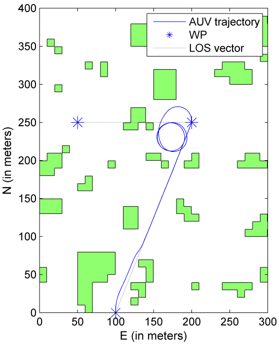


Figure B.18: Simulation nr. 18

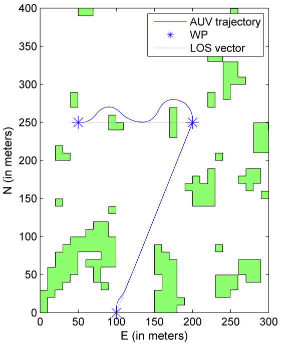


Figure B.19: Simulation nr. 19

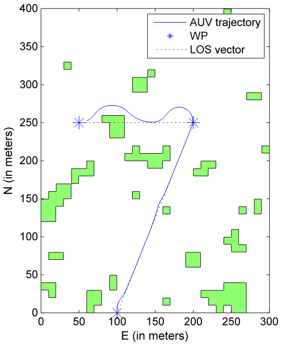


Figure B.20: Simulation nr. 20

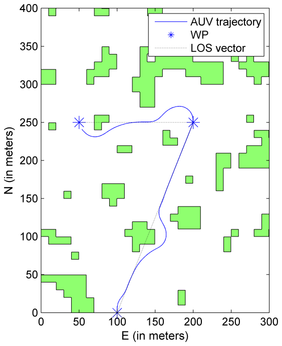


Figure B.21: Simulation nr. 21

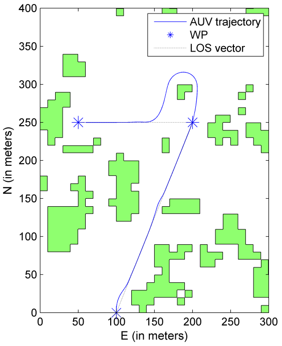


Figure B.22: Simulation nr. 22

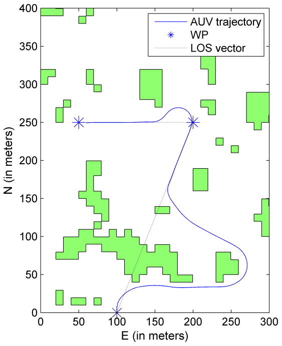


Figure B.23: Simulation nr. 23

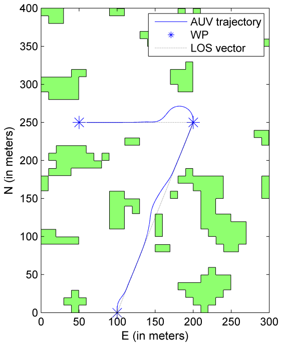


Figure B.24: Simulation nr. 24

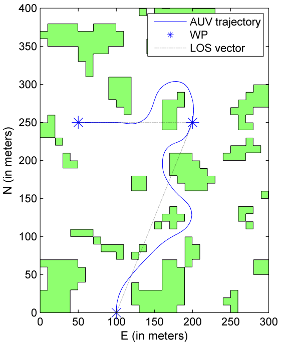


Figure B.25: Simulation nr. 25

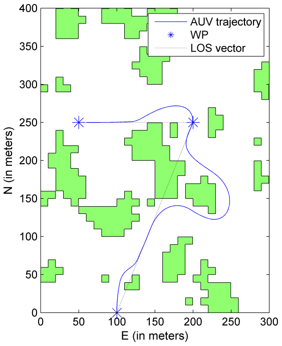


Figure B.26: Simulation nr. 26

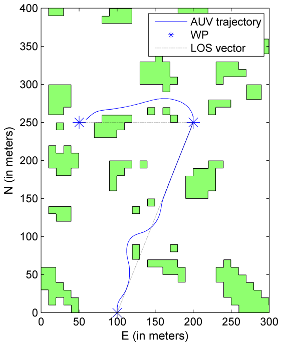


Figure B.27: Simulation nr. 27

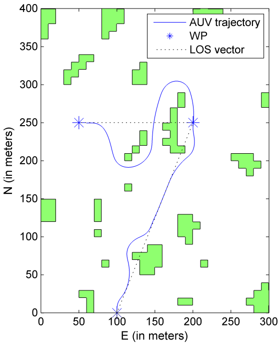


Figure B.28:
Simulation nr. 28

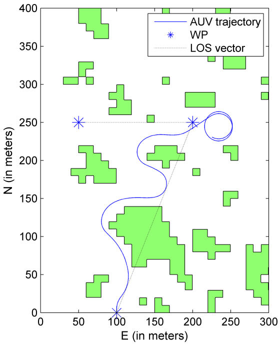


Figure B.29:
Simulation nr. 29

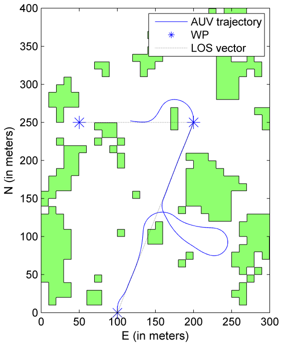


Figure B.30:
Simulation nr. 30

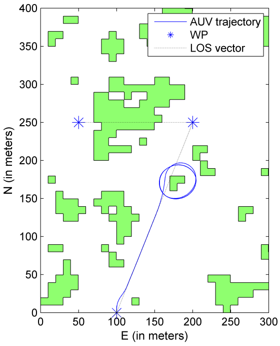


Figure B.31:
Simulation nr. 31

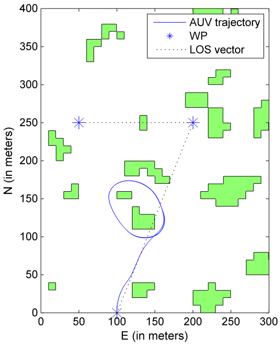


Figure B.32:
Simulation nr. 32

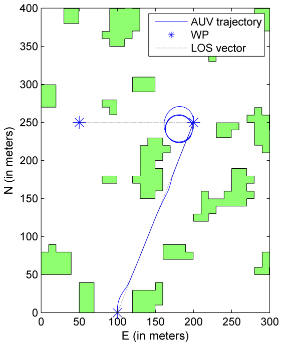


Figure B.33:
Simulation nr. 33

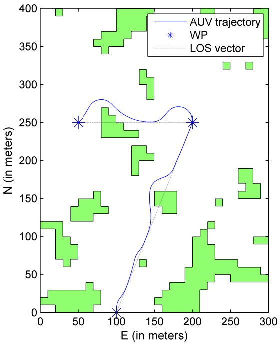


Figure B.34:
Simulation nr. 34

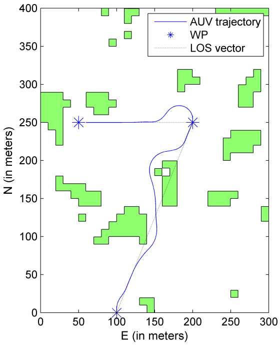


Figure B.35:
Simulation nr. 35

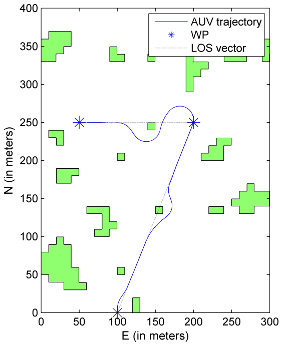


Figure B.36:
Simulation nr. 36

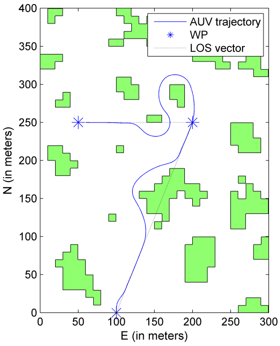


Figure B.37: Simulation nr. 37

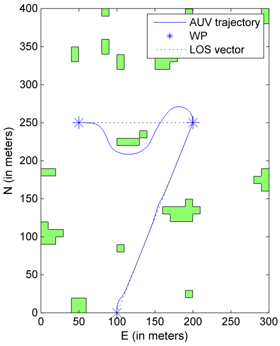


Figure B.38: Simulation nr. 38

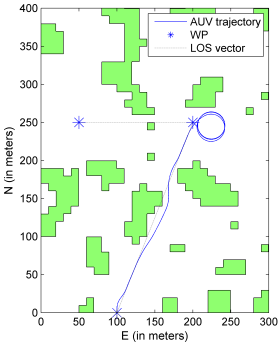


Figure B.39: Simulation nr. 39

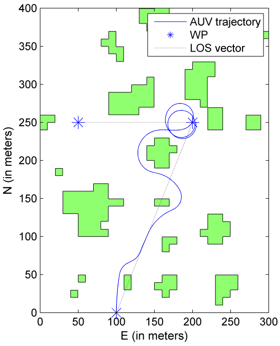


Figure B.40: Simulation nr. 40

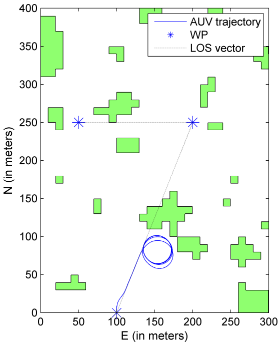


Figure B.41: Simulation nr. 41

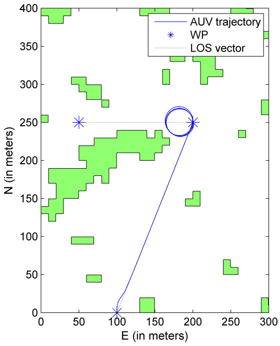


Figure B.42: Simulation nr. 42

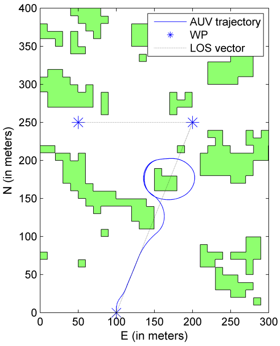


Figure B.43: Simulation nr. 43

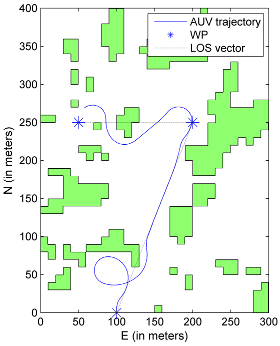


Figure B.44: Simulation nr. 44

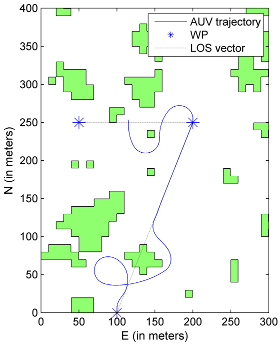


Figure B.45: Simulation nr. 45

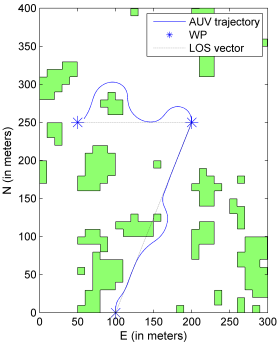


Figure B.46: Simulation nr. 46

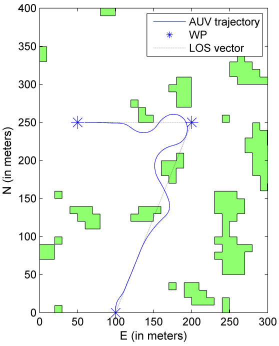


Figure B.47: Simulation nr. 47

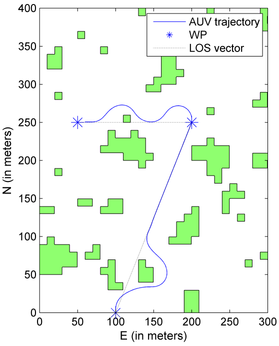


Figure B.48: Simulation nr. 48

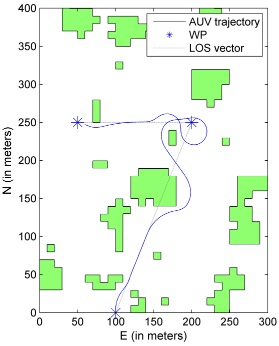


Figure B.49: Simulation nr. 49

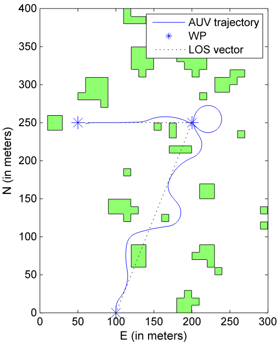


Figure B.50: Simulation nr. 50

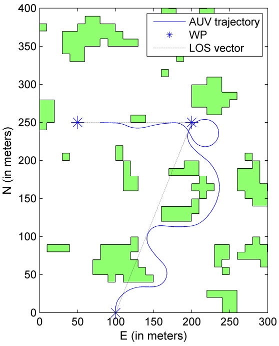


Figure B.51: Simulation nr. 51

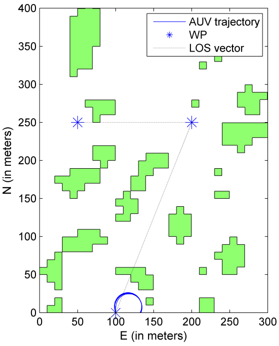


Figure B.52: Simulation nr. 52

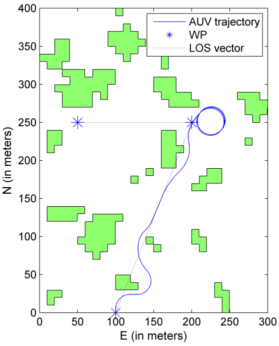


Figure B.53: Simulation nr. 53

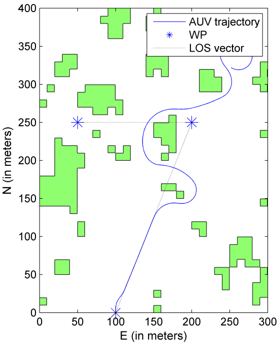


Figure B.54: Simulation nr. 54

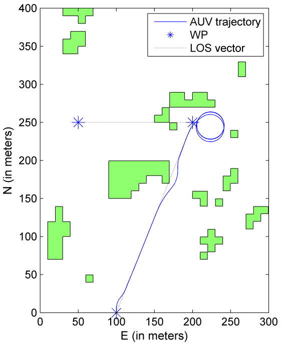


Figure B.55: Simulation nr. 55

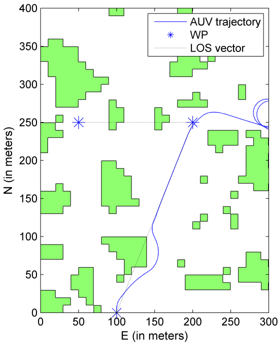


Figure B.56: Simulation nr. 56

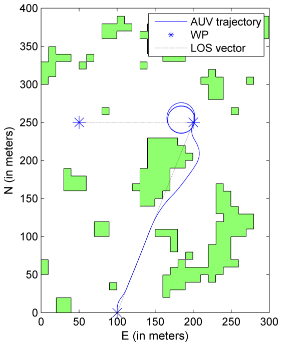


Figure B.57: Simulation nr. 57

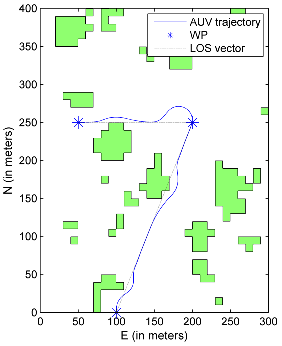


Figure B.58: Simulation nr. 58

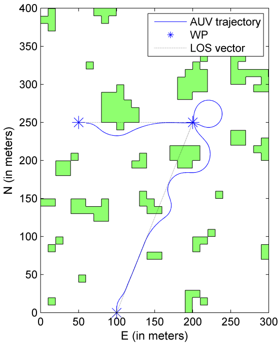


Figure B.59: Simulation nr. 59

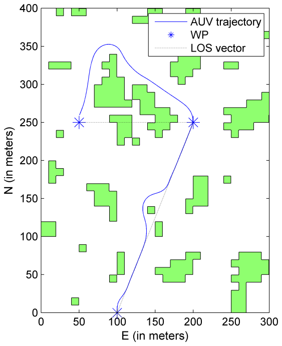


Figure B.60: Simulation nr. 60

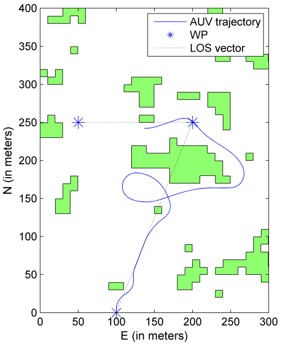


Figure B.61: Simulation nr. 61

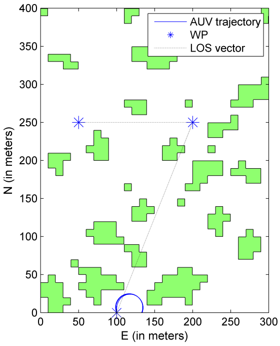


Figure B.62: Simulation nr. 62

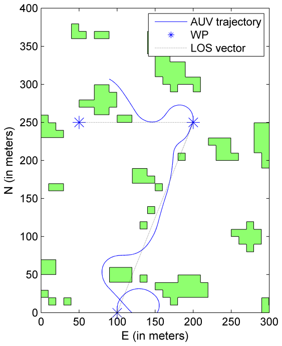


Figure B.63: Simulation nr. 63

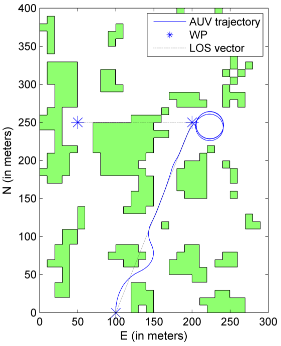


Figure B.64: Simulation nr. 64

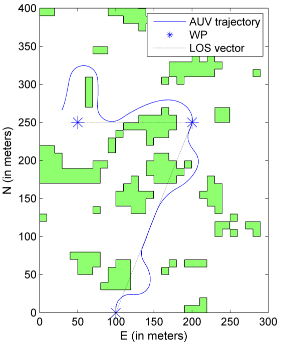


Figure B.65: Simulation nr. 65

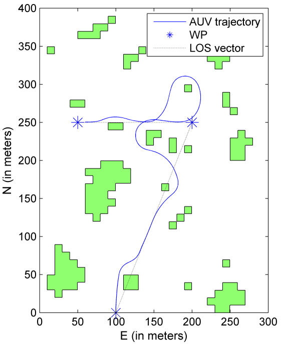


Figure B.66: Simulation nr. 66

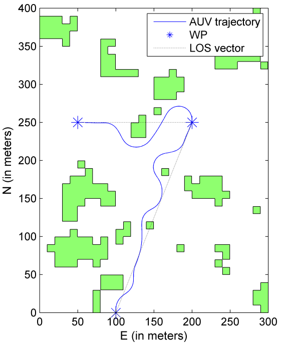


Figure B.67: Simulation nr. 67

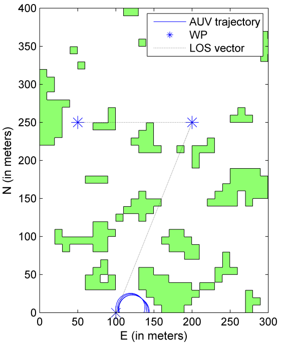


Figure B.68: Simulation nr. 68

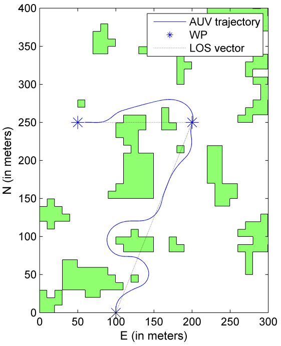


Figure B.69: Simulation nr. 69

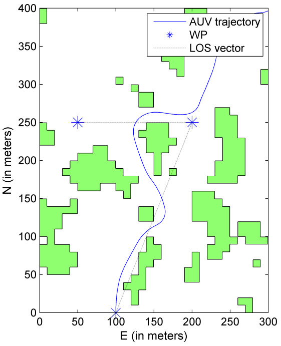


Figure B.70: Simulation nr. 70

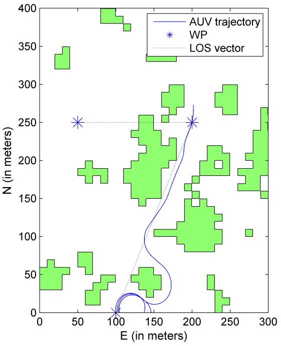


Figure B.71: Simulation nr. 71

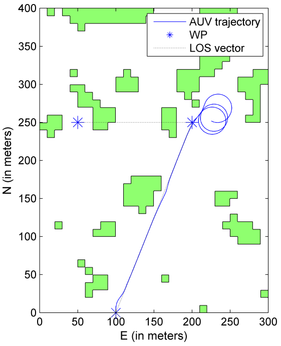


Figure B.72: Simulation nr. 72

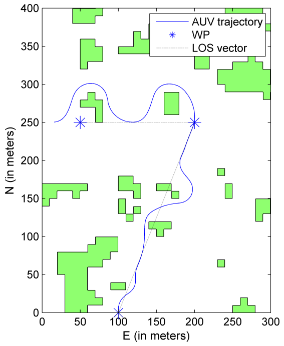


Figure B.73: Simulation nr. 73

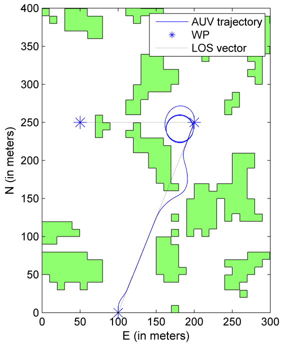


Figure B.74: Simulation nr. 74

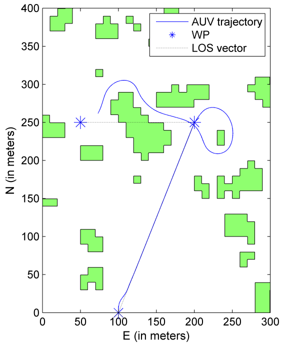


Figure B.75: Simulation nr. 75

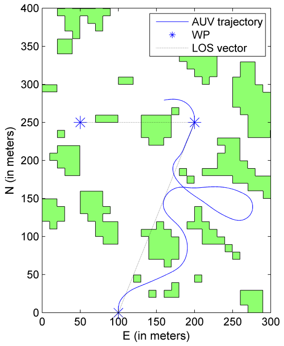


Figure B.76: Simulation nr. 76

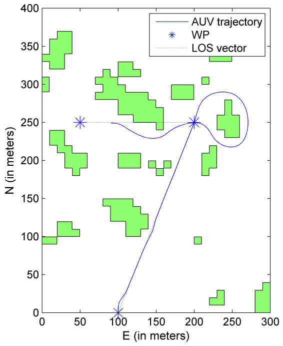


Figure B.77: Simulation nr. 77

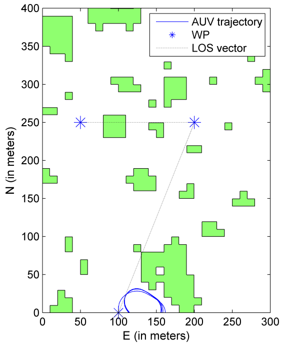


Figure B.78: Simulation nr. 78

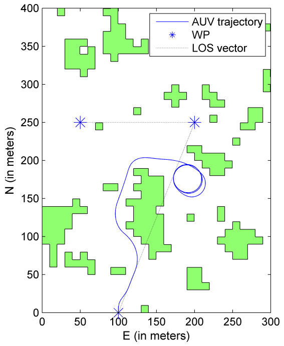


Figure B.79: Simulation nr. 79

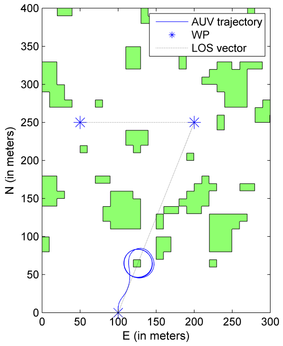


Figure B.80: Simulation nr. 80

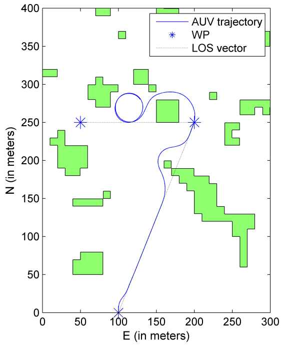


Figure B.81: Simulation nr. 81

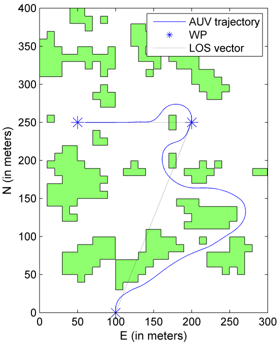


Figure B.82: Simulation nr. 82

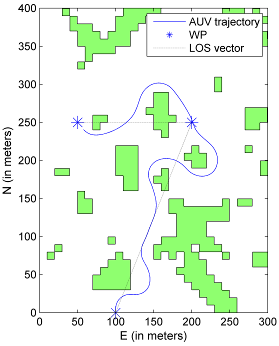


Figure B.83: Simulation nr. 83

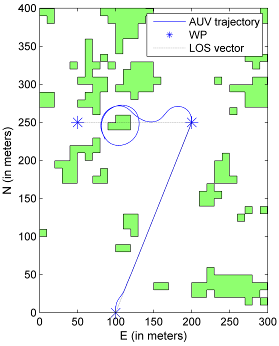


Figure B.84: Simulation nr. 84

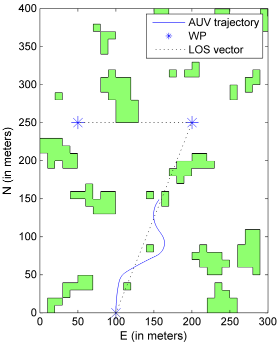


Figure B.85: Simulation nr. 85

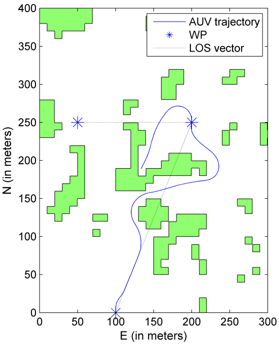


Figure B.86: Simulation nr. 86

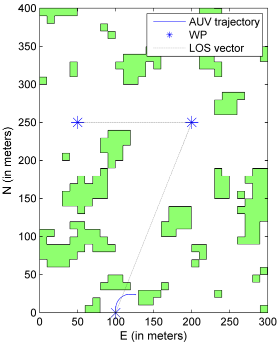


Figure B.87: Simulation nr. 87

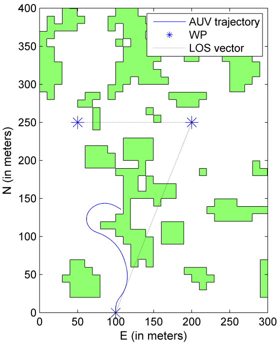


Figure B.88: Simulation nr. 88

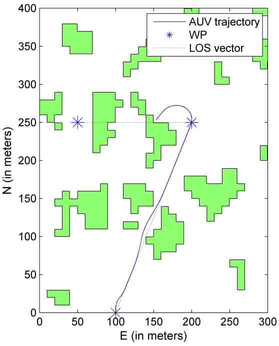


Figure B.89: Simulation nr. 89

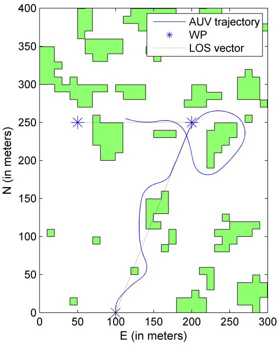


Figure B.90: Simulation nr. 90

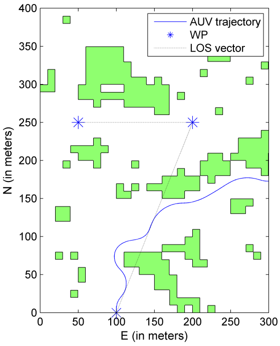


Figure B.91: Simulation nr. 91

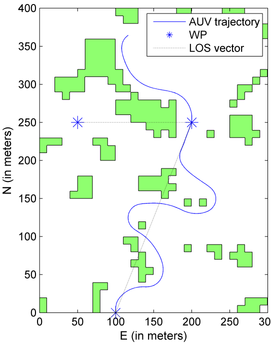


Figure B.92: Simulation nr. 92

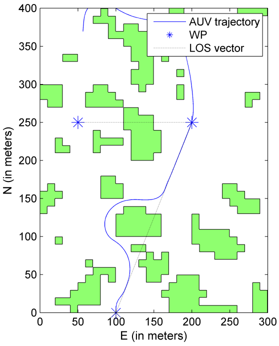


Figure B.93: Simulation nr. 93

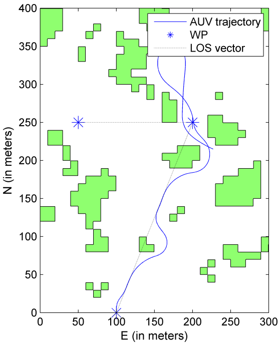


Figure B.94: Simulation nr. 94

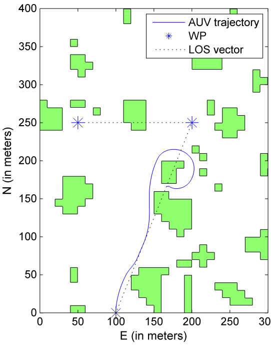


Figure B.95: Simulation nr. 95

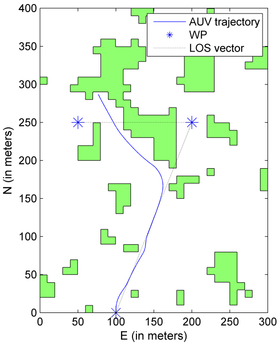


Figure B.96: Simulation nr. 96

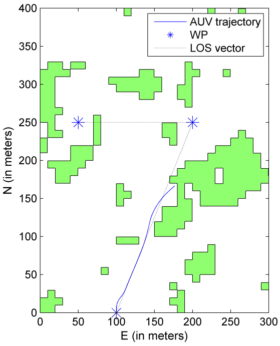


Figure B.97: Simulation nr. 97

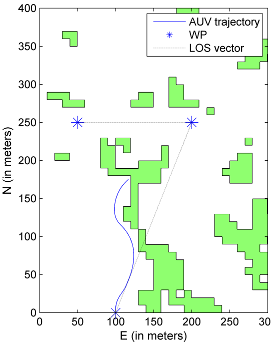


Figure B.98: Simulation nr. 98

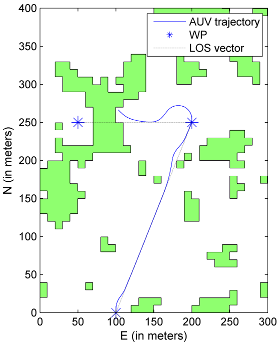


Figure B.99: Simulation nr. 99

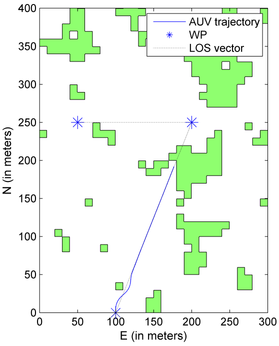


Figure B.100: Simulation nr. 100

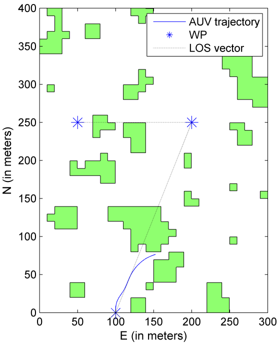


Figure B.101: Simulation nr. 101

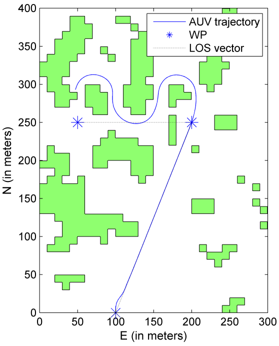


Figure B.102: Simulation nr. 102

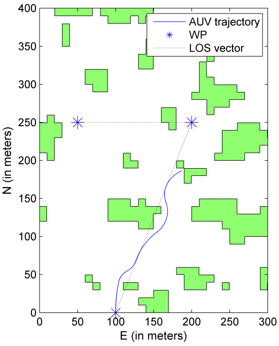


Figure B.103: Simulation nr. 103

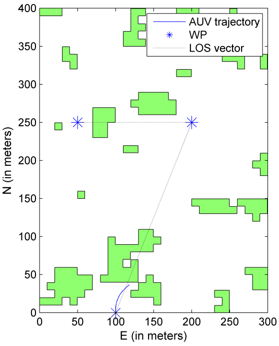


Figure B.104: Simulation nr. 104

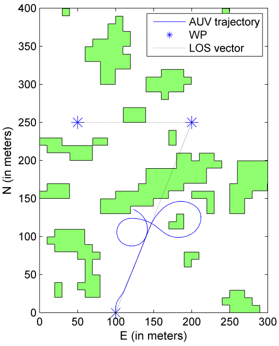


Figure B.105: Simulation nr. 105

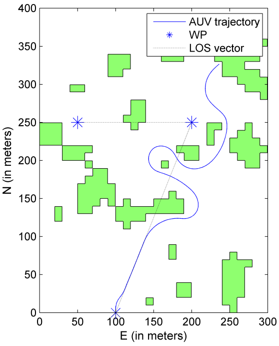


Figure B.106: Simulation nr. 106

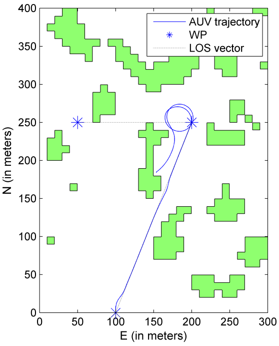


Figure B.107: Simulation nr. 107

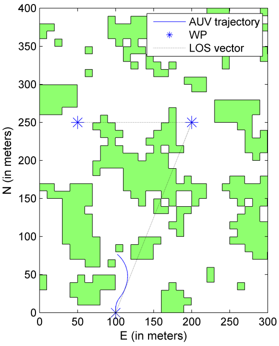


Figure B.108: Simulation nr. 108

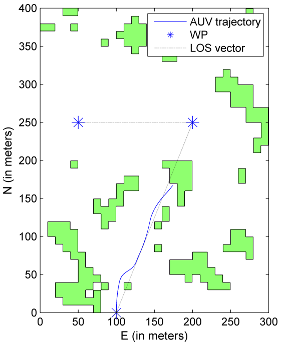


Figure B.109: Simulation nr. 109

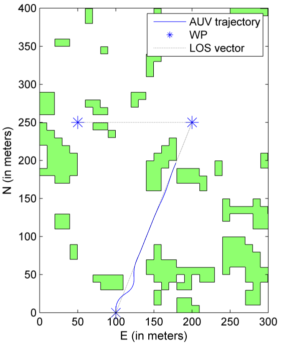


Figure B.110: Simulation nr. 110

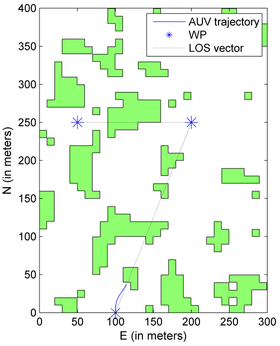


Figure B.111: Simulation nr. 111

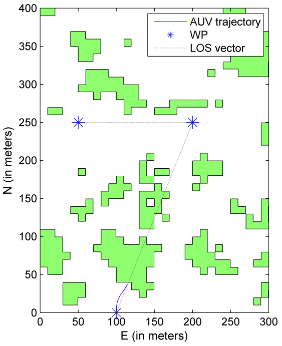


Figure B.112: Simulation nr. 112

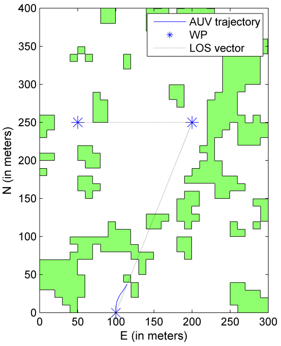


Figure B.113: Simulation nr. 113

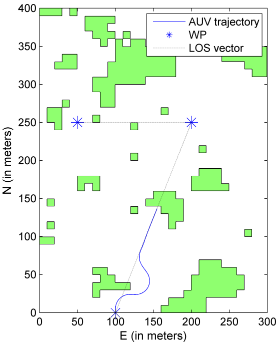


Figure B.114: Simulation nr. 114

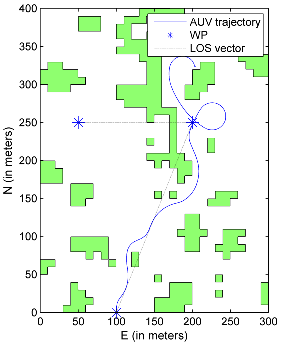


Figure B.115: Simulation nr. 115

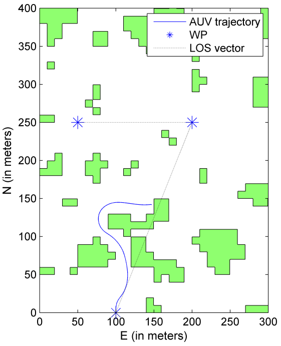


Figure B.116: Simulation nr. 116

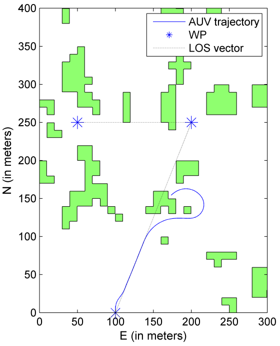


Figure B.117: Simulation nr. 117

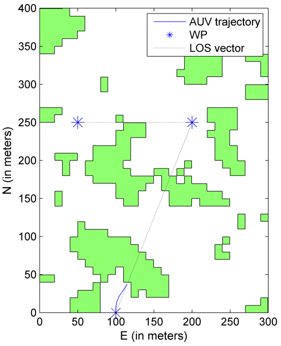


Figure B.118: Simulation nr. 118

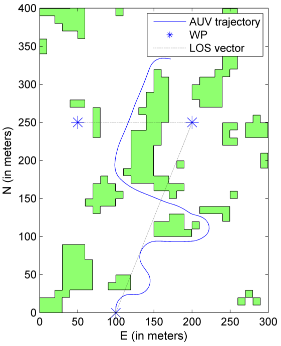


Figure B.119: Simulation nr. 119

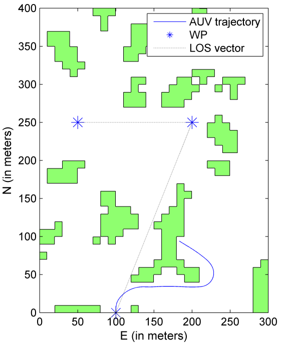


Figure B.120: Simulation nr. 120

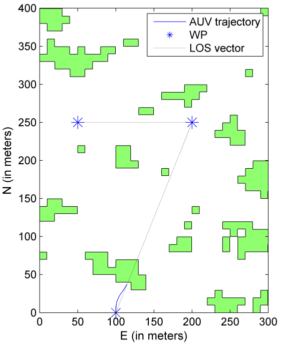


Figure B.121: Simulation nr. 121

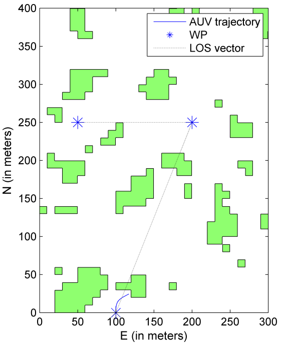


Figure B.122: Simulation nr. 122

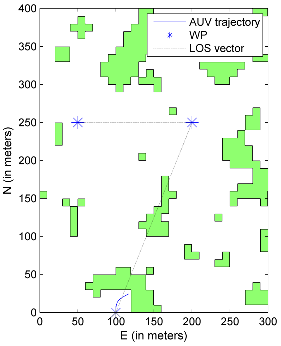


Figure B.123: Simulation nr. 123

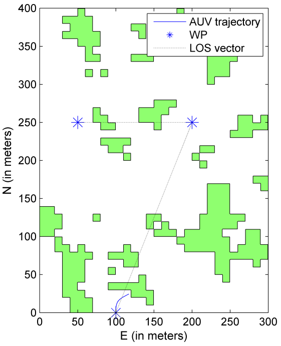


Figure B.124: Simulation nr. 124

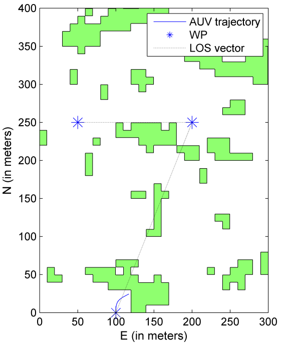


Figure B.125: Simulation nr. 125

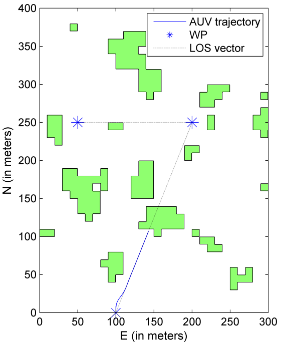


Figure B.126: Simulation nr. 126

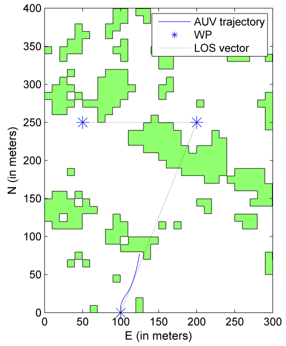


Figure B.127: Simulation nr. 127

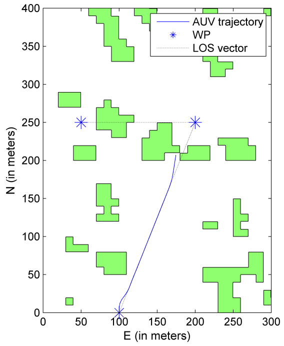


Figure B.128: Simulation nr. 128

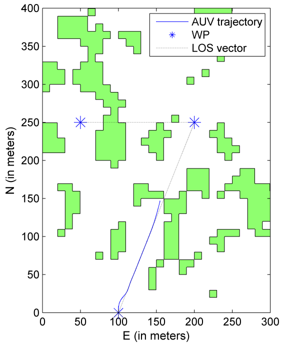


Figure B.129: Simulation nr. 129

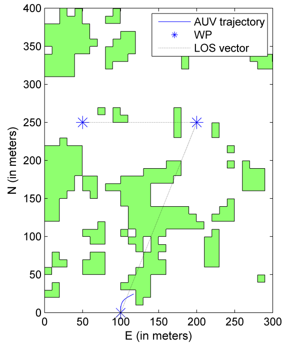


Figure B.130: Simulation nr. 130

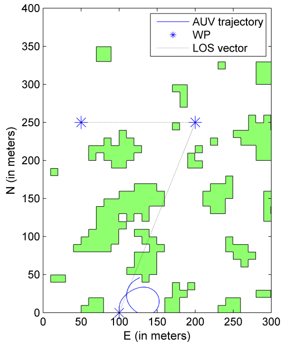


Figure B.131: Simulation nr. 131

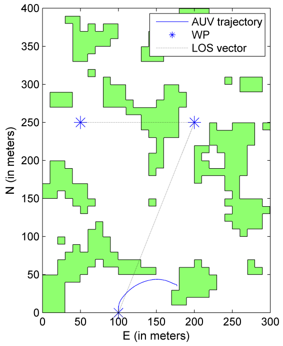


Figure B.132: Simulation nr. 132

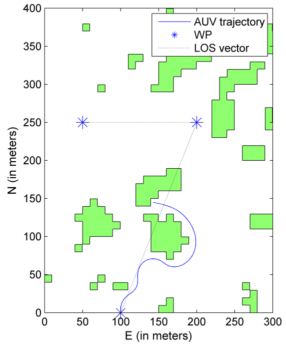


Figure B.133: Simulation nr. 133

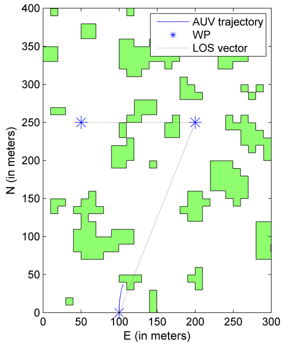


Figure B.134: Simulation nr. 134

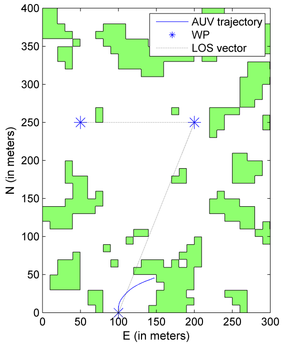


Figure B.135: Simulation nr. 135

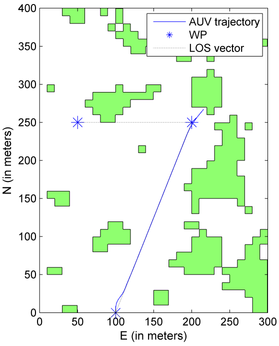


Figure B.136:
Simulation nr. 136

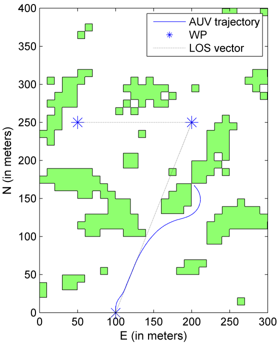


Figure B.137:
Simulation nr. 137

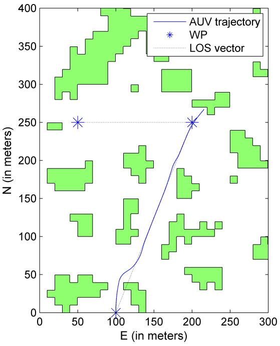


Figure B.138:
Simulation nr. 138

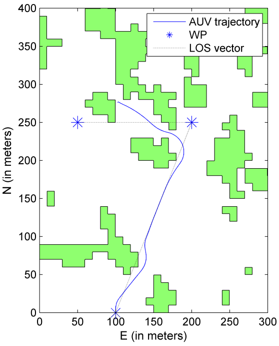


Figure B.139:
Simulation nr. 139

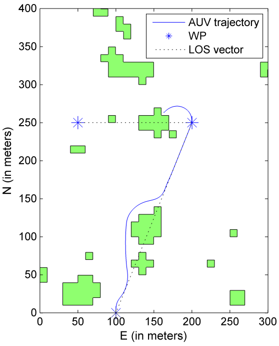


Figure B.140:
Simulation nr. 140

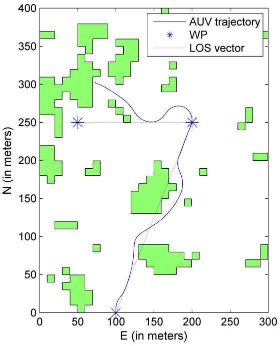


Figure B.141:
Simulation nr. 141

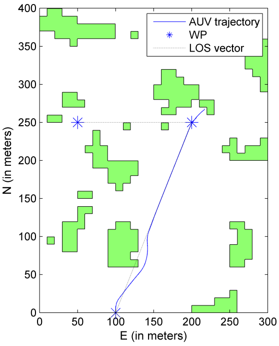


Figure B.142:
Simulation nr. 142

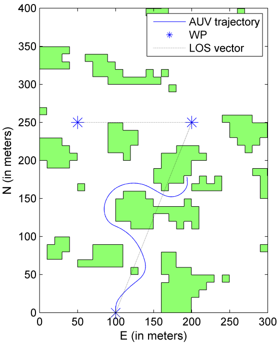


Figure B.143:
Simulation nr. 143

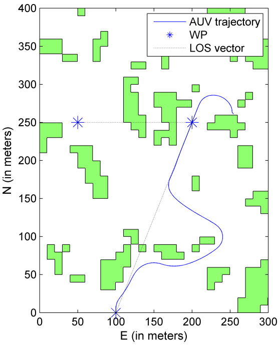


Figure B.144:
Simulation nr. 144

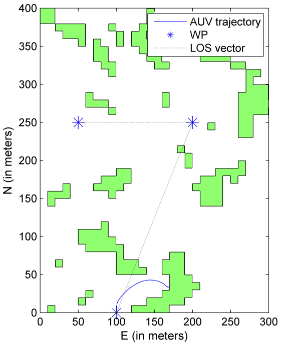


Figure B.145: Simulation nr. 145

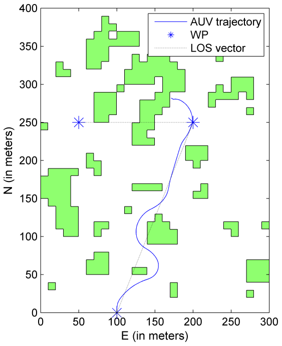


Figure B.146: Simulation nr. 146

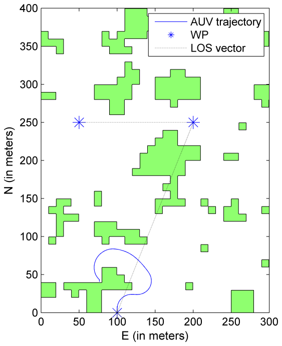


Figure B.147: Simulation nr. 147

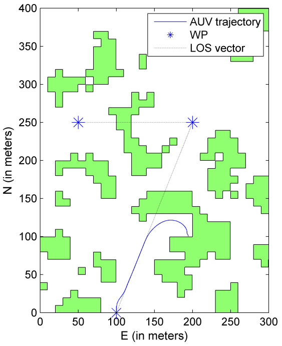


Figure B.148: Simulation nr. 148

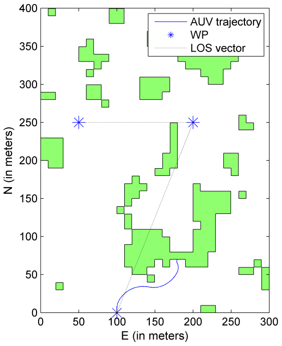


Figure B.149: Simulation nr. 149

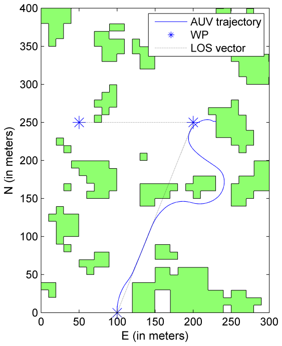


Figure B.150: Simulation nr. 150

Bibliography

[Berti et al., 2008] Berti, H., Sappa, A., and Agamennoni, O. (2008). Improved dynamic window approach by using Lyapunov stability criteria. *Latin American Applied Research*, 38(4):289–298.

[Borenstein and Koren, 1991] Borenstein, J. and Koren, Y. (1991). The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288.

[Børhaug et al., 2007] Børhaug, E., Pavlov, A., and Pettersen, K. Y. (2007). Straight Line Path Following for Formations of Underactuated Underwater Vehicles. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 2905–2912, New Orleans, LA, USA.

[Børhaug et al., 2008] Børhaug, E., Pavlov, A., and Pettersen, K. Y. (2008). Integral LOS Control for Path Following of Underactuated Marine Surface Vessels in the Presence of Constant Ocean Currents. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 4984–4991.

[Brock and Khatib, 1999] Brock, O. and Khatib, O. (1999). High-speed navigation using the global dynamic window approach. *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, 1(May).

[Caharija, 2014] Caharija, W. (2014). *Integral Line-of-Sight Guidance and Control of Underactuated Marine Vehicles.* PhD thesis, NTNU.

[Caharija et al., 2012] Caharija, W., Pettersen, K. Y., Gravdahl, J. T., and Bø rhaug, E. (2012). Integral LOS Guidance for Horizontal Path Following of Underactuated Autonomous Underwater Vehicles in the Presence of Vertical Ocean Currents. In *2012 American Control Conference*, pages 5427–5434.

[Castro et al., 2002] Castro, D., Nunes, U., and Ruano, A. (2002). Reactive local navigation. In *IECON Proceedings (Industrial Electronics Conference)*, volume 3, pages 2427–2432.

[Egeland and Gravdahl, 2003] Egeland, O. and Gravdahl, J. T. (2003). *Modeling and Simulation for Automatic Control.* Marine Cybernetics, Trondheim, Norway.

[Elfes, 1987] Elfes, A. (1987). Sonar-based real-world mapping and navigation. *IEEE Journal on Robotics and Automation*, 3(3).

[Engelhardtsen, 2007] Engelhardtsen, Ø. (2007). 3D AUV Collision Avoidance.

[Eriksen, 2014] Eriksen, B.-O. H. (2014). Horizontal Collision Avoidance for Autonomous Underwater Vehicles.

[Fleury et al., 1995] Fleury, S., Soueres, P., Laumond, J.-P., and Chatila, R. (1995). Primitives for smoothing mobile robot trajectories. *IEEE Transactions on Robotics and Automation*, 11:441–448.

[Fossen, 2011] Fossen, T. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, Ltd, Trondheim, Norway.

[Fox et al., 1997] Fox, D., Thrun, S., and Burgard, W. (1997). The Dynamic Window Approach to Collision Avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33.

[Fredriksen and Pettersen, 2006] Fredriksen, E. and Pettersen, K. (2006). Global k-exponential way-point maneuvering of ships: Theory and experiments. *Automatica*, 42(4):677–687.

[Hespanha, 2009] Hespanha, J. P. (2009). *Linear systems theory*. Princeton University Press, New Jersey.

[Inñigo Blasco et al., 2014] Inñigo Blasco, P., Díaz-del Río, F., Vicente Díaz, S., and Cagigas Muñiz, D. (2014). The Shared Control Dynamic Windows Approach for Non-Holonomic Semi-Autonomous Robots. In *Isr Robotik*, pages 355–360.

[Jalving et al., 2004] Jalving, B., Gade, K., Hagen, O. K., and Vestgård, K. (2004). A toolbox of aiding techniques for the HUGIN AUV integrated inertial navigation system. *Modeling, Identification and Control*, 25(September):173–190.

[Khalil, 2002] Khalil, H. K. (2002). *Nonlinear Systems*. Prentice Hall, Upper Saddle River, New Jersey 07458, third edition.

[Khatib, 1985] Khatib, O. (1985). Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *The International Journal of Robotics Research*, 5(1):500–505.

[Kiss and Tevesz, 2012] Kiss, D. and Tevesz, G. (2012). Advanced dynamic window based navigation approach using model predictive control. *2012 17th International Conference on Methods & Models in Automation & Robotics (MMAR)*, pages 148–153.

[Kongsberg Maritime, 2014] Kongsberg Maritime (2014). Autonomous underwater vehicle - HUGIN. `http://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/B3F87A63D8E419E5C1256A68004E946C?OpenDocument`. Accessed: 2014-10-06.

[Koren and Borenstein, 1991] Koren, Y. and Borenstein, J. (1991). Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 1398–1404, Sacramento, California.

[Loe, 2008] Loe, Ø. A. G. (2008). Collision Avoidance for Unmanned Surface Vehicles.

[Mathworks, 2014a] Mathworks (2014a). MATLAB - The Language of Technical Computing. `http://se.mathworks.com/products/matlab/`. Accessed: 2014-12-18.

[Mathworks, 2014b] Mathworks (2014b). SIMULINK - Simulation and Model-Based Design. `http://se.mathworks.com/products/simulink/`. Accessed: 2014-12-18.

[Mathworks, 2015] Mathworks (2015). Vectorization - MATLAB & Simulink - MathWorks Nordic. `http://se.mathworks.com/help/matlab/matlab_prog/vectorization.html?refresh=true`. Accessed: 2015-05-05.

[Morgado et al., 2011] Morgado, M., Batista, P., Oliveira, P., and Silvestre, C. (2011). Position USBL/DVL sensor-based navigation filter in the presence of unknown ocean currents. *Automatica*, 47(12):2604–2614.

[National Ocean Service, 2014] National Ocean Service (2014). What is sonar? `http://oceanservice.noaa.gov/facts/sonar.html`. Accessed: 2014-12-17.

[Ögren and Leonard, 2002] Ögren, P. and Leonard, N. E. (2002). A Tractable Convergent Dynamic Window Approach to Obstacle Avoidance. In *Proceedings of the 2992 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems EPFL*, pages 595–600.

[Oriolo and Nakamura, 1991] Oriolo, G. and Nakamura, Y. (1991). Control of mechanical systems with second-order nonholonomic constraints: underactuated manipulators. In *30th IEEE Conference on Decision and Control.*

[Panteley et al., 1998] Panteley, E., Lefeber, E., Loria, A., and Nijmeijer, H. (1998). Exponential Tracking Control of a Mobile Car Using a Cascaded Approach. In *IFAC Workshop on Motion Control*, pages 221–226.

[Panteley and Loria, 1998] Panteley, E. and Loria, A. (1998). On Global Uniform Asymptotic Stability of Nonlinear Time-varying Systems in Cascade. *Systems and Control Letters*, 33(2):131–138.

[Rodriguez-Seda et al., 2014] Rodriguez-Seda, E. J., Tang, C., Spong, M. W., and Stipanovi, D. M. (2014). Trajectory tracking with collision avoidance for nonholonomic vehicles with acceleration constraints and limited sensing. *The International Journal of Robotics Research*, 33(12):1569–1592.

[Schröter et al., 2007] Schröter, C., Höchemer, M., and Gross, H.-M. (2007). A Particle Filter for the Dynamic Window Approach to Mobile Robot Control. In *Proc. 52nd Int. Scientific Colloquium (IWK)*, volume I, pages 425–430.

[Seder et al., 2005] Seder, M., Macek, K., and Petrovic, I. (2005). An integrated approach to real-time mobile robot control in partially known indoor environments. *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005.*, pages 1785–1790.

[Seder and Petrović, 2007] Seder, M. and Petrović, I. (2007). Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1986–1991.

[Simmons, 1996] Simmons, R. (1996). The curvature-velocity method for local obstacle avoidance. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 4, pages 3375 –3382 vol.4.

[SNAME, 1950] SNAME (1950). Nomenclature for Treating the Motion of a Submerged Body Through a Fluid. Technical report, The Society of Naval Architects and Marine Engineers, New York, USA.

[Spong et al., 2006] Spong, M. W., Hutchinson, S., and Vidyasagar, M. (2006). *Robot Modeling and Control*. John Wiley & Sons, Ltd.

[Statens Kartverk, 2014] Statens Kartverk (2014). Terrengmodeller. `http://www.statkart.no/Kart/Kartdata/Terrengmodeller/`. Accessed: 2014-10-07.

[Tan et al., 2004a] Tan, C. S., Sutton, R., and Chudley, J. (2004a). Collision Avoidance Systems for Autonomous Underwater Vehicles Part A: A Review of Obstacle Detection. *Journal of Marine Science and Environment*, Part C(No C2):39–50.

[Tan et al., 2004b] Tan, C. S., Sutton, R., and Chudley, J. (2004b). Collision Avoidance Systems for Autonomous Underwater Vehicles Part B: A Review of Obstacle Avoidance. *Journal of Marine Science and Environment*, Part C(No C2):51–62.

[Tusseyeva et al., 2013] Tusseyeva, I., Kim, S.-g., and Kim, Y.-g. (2013). 3D Global Dynamic Window Approach for Navigation of Autonomous Underwater Vehicles. *International Journal of Fuzzy Logic and Intelligent Systems*, 13(2):91–99.

[Wikipedia, 2015] Wikipedia (2015). Heaviside step function. `http://en.wikipedia.org/wiki/Heaviside_step_function`. Accessed: 2015-05-13.