Master's thesis

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

Eivind Duus Molven

Optimal Control-based Docking for Autonomous Ferries

Master's thesis in Cybernetics and Robotics
Supervisor: Glenn Bitar, Morten Breivik and Alf Magne Meling
June 2020

NTNU
Kunnskap for en bedre verden

Eivind Duus Molven

Optimal Control-based Docking for Autonomous Ferries

Master's thesis in Cybernetics and Robotics
Supervisor: Glenn Bitar, Morten Breivik and Alf Magne Meling
June 2020

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

NTNU
Kunnskap for en bedre verden

Preface

This master thesis is the culmination of work from my five years of studying at the Norwegian University of Science and Technology (NTNU). It has been the most extensive project I have ever performed and has provided me with demanding challenges and invaluable experience. I am very grateful to my advisors Glenn Bitar and Morten Breivik for excellent discussions, ideas, support, proof-reading, and the hours of testing on the Trondheimsfjord. Thanks to Alf Magne Meling at Kongsberg Maritime for providing insightful knowledge about the sensors and methods being used by automatic docking vessels in operation today. I am also indebted to Andreas Aurlien, Håvard Bærug and Magnus Jervan for helping out on the sea experiments, teaching me boating knots and working as a great crew. Finally, I wish to thank my friends for making these years of studying quite enjoyable, and of course my family for their never-ending support and interest in my work and well-being.

Eivind Duus Molven
Trondheim, June 1, 2020

The author and milliAmpere during the experimental week in April 2020.

Abstract

For any marine vessel, the docking process is an especially demanding task. Automating this process will provide increased safety, less energy-usage, reduced stress for the crew and lowering of costs. This thesis summarises the development of an energy-efficient, optimization-based method for planning docking trajectories with obstacle avoidance. The thesis covers the implementation of this method in a full-scale ship system, and demonstrates the method's feasibility, efficiency and reliability through simulations and sea trials with the experimental ferry milliAmpere.

In the field of automatic docking for Autonomous Surface Vehicles (ASVs), optimization-based methods have provided promising results. In this thesis, such a method is used to generate an optimal trajectory for the ASV to follow. By formulating the problem as an Optimal Control Problem (OCP), operational limitations on velocities, energy usage and obstacle avoidance are added to the problem. By using the multiple shooting method to transcribe the OCP to a Nonlinear Program (NLP), an optimal trajectory is generated through an interior point algorithm.

The trajectory planner is implemented on the experimental platform milliAmpere. To ensure easy monitoring of the vessel's behaviour during sea trials and feasibility of the trajectories, a graphical user interface has been developed. This interface consists of a monitor for reference tracking capabilities, visualizations of the planned trajectory in a map with the possibility of waypoint selection, and a real-time plot showing the instantaneous power consumption. In addition, to ensure obstacle avoidance capabilities, a universal method for creating OCP-constraints from map polygons is developed, along with an A*-based warm-starting algorithm that helps the OCP to avoid local minimums and improve convergence time. Simulations and sea trials with this system using the milliAmpere ferry demonstrate its ability to create robust, feasible and energy-efficient trajectories suitable for harbour docking operations.

Sammendrag

Dokking til kai er ansett som en svært krevende operasjon for de fleste marine fartøy. Automatisering av denne prosessen vil kunne føre til bedre sikkerhet, lavere energiforbruk, mindre stress for mannskapet og lavere driftskostnader. Denne oppgaven starter med å oppsummere utviklingen av en energi-effektiv, optimeringsbasert metode for å planlegge dokking-baner som unngår hindringer. Videre utdyper oppgaven hvordan denne metoden er blitt realisert i et fullskala skipssystem, samt demonstrerer metodens effektivitet, gjennomførbarhet og pålitelighet gjennom simuleringer og sjøtester på den eksperimentelle fergen milliAmpere.

Optimeringsbaserte metoder har tidligere gitt gode resultater i forbindelse med dokking av Autonome Overflatefartøy (ASV). I denne oppgaven blir en slik metode brukt for å generere en optimal bane som fartøyet skal følge. Ved å formulere problemet som et Optimalt Kontrolproblem (OCP) blir begrensninger på hastigheter og energiforbruk, samt unngåelse av hindringer inkludert i problemet. Videre blir Multiple Shooting-metoden anvendt for å transkribere OCPet til et Ulineært Program (NLP) slik at en optimal bane kan bli funnet ved hjelp av en interior point-algoritme.

Baneplanleggeren er implementert på det eksperimentelle fartøyet milliAmpere. For å sikre at båtens oppførsel og intensjoner lett kan overvåkes, blir et grafisk brukergrensesnitt presentert. Dette brukergrensesnittet består av en modul som viser følgingen av referanseposisjonen, en visualisering av den planlagte banen, samt en graf som viser fartøyets effektforbruk over tid. For å sikre at hindringer blir unngått har det også blitt utviklet en universell metode som lager OCP-restriksjoner basert på polygonkart, i tillegg til en A*-basert varm-starts algoritme som sørger for at lokale minimumer ikke opptrer og konvergenstiden blir forbedret. Simuleringer og sjøtester med dette systemet på fergen milliAmpere, viser at metoden klarer å lage robuste, gjennomførbare og energi-effektive baner som er godt tilpasset dokking i havneområder.

Table of Contents

Preface i

Abstract iii

Sammendrag v

Table of Contents ix

List of Tables xi

List of Figures xv

Acronyms xvi

Nomenclature xix

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Related Work . . . 3
- 1.3 Problem Description . . . 4
- 1.4 Contributions . . . 4
- 1.5 Outline . . . 4

2 Background Theory 7
- 2.1 Mathematical Modelling . . . 7
  - 2.1.1 Ship Kinematics and Kinetics . . . 7
- 2.2 Optimization Theory . . . 10
- 2.3 Motion Control . . . 11
  - 2.3.1 Trajectory Tracking . . . 12
  - 2.3.2 Dynamic Positioning . . . 12
- 2.4 Control Allocation . . . 13
  - 2.4.1 Nonlinear Constrained Control Allocation for Azimuth Thrusters 13

2.4.2 Nonlinear Scalar Control Allocation . . . 14
2.5 Transcription of Optimal Control Problems . . . 15
2.6 Obstacle Representation . . . 16
2.6.1 Rectangular Rotatable Obstacle . . . 17
2.7 A* Search Algorithm . . . 18
2.8 Robot Operating System . . . 19

3 The Trajectory Planner for Docking 21
3.1 The OCP Model . . . 21
3.1.1 Vessel Model . . . 21
3.1.2 Cost Function . . . 22
3.1.3 Representation of Obstacle Constraints . . . 23
3.1.4 From OCP to NLP . . . 23
3.1.5 Solver and Building of Problem . . . 25
3.2 Obstacle Representation of Maps . . . 26
3.2.1 Polygon Map Representation . . . 26
3.2.2 Polygons to Rectangle Obstacles: Concept . . . 27
3.2.3 Polygons to Rectangle Obstacles: Implementation . . . 27
3.2.4 Remarks on Efficiency . . . 29
3.3 A* Warm-starting Algorithm . . . 30
3.4 Trajectory Publisher . . . 30
3.5 Realization of the OCP-TP in the Vessel Computer System . . . 31
3.6 Dynamic Positioning Controller . . . 33

4 Development of the Graphical User Interface 35
4.1 Reference Monitor . . . 38
4.2 Trajectory Map Visualization . . . 40
4.3 Power Usage Plot . . . 42

5 Simulation and Experimental Results 43
5.1 Trajectory Planning Methods For Comparison . . . 43
5.1.1 TRF . . . 43
5.1.2 MPC . . . 44
5.1.3 A-PP . . . 45
5.1.4 Manual . . . 46
5.2 Performance Metrics . . . 46
5.2.1 Power Consumption . . . 46
5.2.2 Trajectory Tracking Capability . . . 47
5.2.3 Obstacle Avoidance . . . 48
5.2.4 Passenger Comfort . . . 48
5.2.5 Actuator Wear and Tear . . . 48
5.3 Scenarios . . . 49
5.3.1 Scenario 1 . . . 49
5.3.2 Scenario 2 . . . 50
5.3.3 Scenario 3 . . . 51
5.4 Simulation Results and Discussion . . . 52

5.4.1 Scenario 1 . . . 52
5.4.2 Scenario 2 . . . 56
5.4.3 Scenario 3 . . . 58
5.4.4 Remarks on the Simulation Results . . . 62
5.5 The milliAmpere Experimental Vessel . . . 63
5.5.1 Vessel Characteristics . . . 63
5.5.2 Navigation Sensors . . . 63
5.5.3 Computer and Software . . . 64
5.6 Experimental Results and Discussion . . . 64
5.6.1 Weather Conditions . . . 64
5.6.2 Scenario 1 . . . 66
5.6.3 Scenario 2 . . . 72
5.6.4 Scenario 3 . . . 77
5.6.5 Concluding Remarks . . . 84

6 Conclusion and Further Work **85**

Bibliography **88**

A milliAmpere Current and Voltage Filtering **93**

B Test Plan for Experiments on milliAmpere **95**

List of Tables

2.1 Marine vessel notation 8

5.1 Normalizing parameters for the IAE metric 47
5.2 Normalizing parameters for the body acceleration metric 49
5.3 Normalizing parameters for the IAE metric 49
5.4 Parameters docking scenario 1 50
5.5 Docking methods of scenario 1 50
5.6 Parameters docking scenario 2 51
5.7 Docking methods of scenario 2 51
5.8 Parameters docking scenario 3 52
5.9 Docking methods of scenario 3 52
5.10 Scenario 1 Simulation: Metrics 53
5.11 Scenario 2 Simulation: Metrics 56
5.12 Scenario 3 Simulation: Metrics 59
5.13 Scenario 1 Experimental: Metrics 66
5.14 Scenario 2 Experimental: Metrics 72
5.15 Scenario 3 Experimental: Metrics 77

A.1 Threshold values for the current and voltage filtering. 93

List of Figures

1.1 Autonomous vessel examples 1
1.2 Conceptual illustrations of the milliAmpere ferry 2
1.3 Illustrations of vessels fitted with docking systems 3

2.1 6-DOF notation for ship 9
2.2 Non-convex function 11
2.3 Thruster configuration for double-ended ferries 15
2.4 Illustration of multiple shooting 16
2.5 Obstacle representation using p-norm 17
2.6 Rectangular rotatable obstacle 18
2.7 Discovery strategy of the A* algorithm 19
2.8 milliAmpere ROS-structure 20

3.1 Trondheim polygon map representation 26
3.2 Square partitioning of the allowed space of movement 27
3.3 Different representations of the Munkholmen island 28
3.4 Rectangular obstacle representation of the Trondheim harbour area. . . . 29
3.5 A* warm-starting 31
3.6 Organization of the OCP-TP 32

4.1 Preliminary graphical interface 36
4.2 New graphical interface 37
4.3 Reference monitor display 38
4.4 Reference monitor with waypoint 39
4.5 Different scale levels of the reference monitor 40
4.6 Map visualizations 41
4.7 Interface power usage plot 42

5.1 Convex feasible region of the Docking MPC method. 45
5.2 Different stages of the A-PP 46
5.3 Illustration of Scenario 1 50

5.4 Illustration of Scneario 2 . . . 51
5.5 Illustration of Scenario 3 . . . 52
5.6 Scenario 1 Simulation: Trajectories . . . 53
5.7 Scenario 1 Simulation: Individual trajectories . . . 54
5.8 Scenario 1 Simulation: Accumulated power consumption . . . 54
5.9 Scenario 1 Simulation: DNO . . . 55
5.10 Scenario 1 Simulation: IAE . . . 55
5.11 Scenario 2 Simulation: Trajectories . . . 56
5.12 Scenario 2 Simulation: Individual trajectories . . . 57
5.13 Scenario 2 Simulation: Accumulated power consumption . . . 57
5.14 Scenario 2 Simulation: DNO . . . 58
5.15 Scenario 2 Simulation: IAE . . . 58
5.16 Scenario 3 Simulation: Path . . . 59
5.17 Scenario 3 Simulation: Path poses . . . 60
5.18 Scenario 3 Simulation: Individual trajectories . . . 60
5.19 Scenario 3 Simulation: Accumulated power consumption . . . 61
5.20 Scenario 3 Simulation: DNO . . . 61
5.21 Scenario 3 Simulation: IAE . . . 62
5.22 Photo of the milliAmpere vessel . . . 63
5.23 Control station of milliAmpere . . . 64
5.24 Location for sea experiments . . . 65
5.25 Illustration of weather conditions . . . 65
5.26 Scenario 1 Experimental: Trajectories . . . 67
5.27 Scenario 1 Experimental: Individual trajectories . . . 68
5.28 Scenario 1 Experimental: DNO . . . 68
5.29 Scenario 1 Experimental: Total energy . . . 69
5.30 Scenario 1 Experimental: IAE . . . 69
5.31 Scenario 1 Experimental: IAEW . . . 70
5.32 Scenario 1 Experimental: Passenger comfort . . . 70
5.33 Scenario 1 Experimental: AWT . . . 71
5.34 Scenario 2 Experimental: Trajectories . . . 72
5.35 Scenario 2 Experimental: Individual trajectories . . . 73
5.36 Scenario 2 Experimental: Total energy . . . 74
5.37 Scenario 2 Experimental: DNO . . . 74
5.38 Scenario 2 Experimental: IAE . . . 75
5.39 Scenario 2 Experimental: IAEW . . . 75
5.40 Scenario 2 Experimental: AWT . . . 76
5.41 Scenario 2 Experimental: Passenger comfort . . . 76
5.42 Scenario 3 Experimental: Path . . . 78
5.43 Scenario 3 Experimental: Path poses . . . 78
5.44 Scenario 3 Experimental: Individual trajectories . . . 79
5.45 Scenario 3 Experimental: Aerial photographs . . . 79
5.46 Scenario 3 Experimental: Total energy . . . 80
5.47 Scenario 3 Experimental: DNO . . . 81
5.48 Scenario 3 Experimental: IAE . . . 81

5.49 Scenario 3 Experimental: IAEW 82
5.50 Scenario 3 Experimental: AWT 82
5.51 Scenario 3 Experimental: Passenger comfort 83
5.52 Scenario 3 Experimental: OCP-TP comparison in wind 83

A.1 Original current and voltage measurements 94
A.2 Current and voltage measurements after filtering 94

Acronyms

A-PP A* Path Planner.

ASV Autonomous Surface Vehicle.

DNO Distance to the Nearest Obstacle.

DOF Degree of Freedom.

DP Dynamic Positioning.

ESKF Error State Kalman Filter.

GNSS Global Navigation Satellite System.

GUI Graphical User Interface.

HTML Hypertext Markup Language.

IAE Integral of the Absolute Error.

IAEW Integral of the Absolute Error multiplied by the energy consumption.

IMU Inertial Measurement Unit.

IoT Internet of Things.

IPM Interior Point Method.

IPOPT Interior Point Optimizer.

JS JavaScript.

JSON JavaScript Object Notation.

MBPC Model Based Predicitive Controller.

MPC Model Predictive Control.

MSMC Multivariable Sliding Mode Controller.

NED North-East-Down.

NLP Nonlinear Program.

NMPC Nonlinear Model Predictive Controller.

NSCA Nonlinear Scalar Control Allocation.

NTNU Norwegian University of Science and Technology.

OBC Onboard Computer.

OCP Optimal Control Problem.

OCP-TP Optimal Control Problem Trajectory Planner.

PD Proportional-Derivative.

PID Proportional-Integral-Derivative.

RC Radio Control.

ROS Robot Operating System.

SNAME Society of Naval Architects and Marine Engineers.

SQP Sequential Quadratic Programming.

TRF Trajectory Reference Filter.

Nomenclature

α	Clockwise rotation of rectangular obstacle
$\boldsymbol{\alpha}_{az}$	Azimuth thruster angle
$\boldsymbol{\eta}$	Vector in the NED frame, giving the vessel pose in north, east and yaw
$\boldsymbol{\eta}_d$	Vector of desired pose variables in the NED frame
$\boldsymbol{\nu}$	Linear and angular velocity vector in the body reference frame
$\boldsymbol{\nu}_d$	Vector of desired velocity variables in the body frame
$\boldsymbol{\tau}_c$	Controller force vector
$\boldsymbol{\tau}_{fb}$	Feedback controller term
$\boldsymbol{\tau}_{ff}$	Feed-forward controller term
$\boldsymbol{C}$	Coriolis matrix
$\boldsymbol{C}_A$	Added-mass Coriolis matrix
$\boldsymbol{C}_{RB}$	Rigid-body Coriolis matrix
$\boldsymbol{D}$	Damping matrix
$\boldsymbol{g}$	Constraints in NLP
$\boldsymbol{g}_{lb}$	Lower bounds on constraints in NLP
$\boldsymbol{g}_{ub}$	Upper bounds on constraints in NLP
$\boldsymbol{K}_d$	Derivative controller gain
$\boldsymbol{K}_i$	Integral controller gain
$\boldsymbol{K}_p$	Proportionate controller gain

$\boldsymbol{M}$	System inertia matrix
$\boldsymbol{M}_A$	Hydrodynamic-added mass matrix
$\boldsymbol{M}_{RB}$	Rigid body mass matrix
$\boldsymbol{R}$	Rotation matrix, rotating body vectors into the NED frame
$\boldsymbol{T}$	Thrust configuration matrix
$\boldsymbol{u}$	Control inputs
$\boldsymbol{w}$	Combined vector of states and inputs in NLP
$\boldsymbol{w}_{lb}$	Lower bounds on states and inputs in NLP
$\boldsymbol{w}_{ub}$	Upper bounds on states and inputs in NLP
$\boldsymbol{x}_0$	Initial vessel state
$\boldsymbol{x}_d$	Combined vector of desired pose values and desired body velocities
$\boldsymbol{x}_f$	Final vessel state
$\mathcal{C}$	Rectangular obstacle equation
ϕ	General NLP cost function
ψ	Yaw angle of vessel
$\tilde{\boldsymbol{\eta}}$	Controller tracking error
a	Length of rectangular obstacle
b	Width of rectangular obstacle
E_{elec}	Total energy consumption of the azimuth thrusters based on current and voltage measurements
E_{mech}	Total energy consumption of the azimuth thrusters based on propeller revolution speed
F_1	Azimuth 1 thruster force
F_2	Azimuth 2 thruster force
h	Length of timestep in OCP
J	Accumulated cost in OCP
N	Yaw moment
n_s	Propeller shaft revolution speed

P_{elec}	Instantaneous power consumption from azimuth thrusters based on current and voltage measurements
P_{mech}	Instantaneous power consumption from azimuth thrusters based on the propeller revolution speed
r	Yaw rate
t_f	End time in OCP
u	Surge velocity
v	Sway velocity
X	Surge force
x	North position of vessel
x_c	Center of rectangular obstacle in x-direction
Y	Sway force
y	East position of vessel
y_c	Center of rectangular obstacle in y-direction

Chapter 1

Introduction

1.1 Motivation

Automatic and autonomous systems have gained momentum over the last two decades, and are now starting to make a real impact on our society. Heavy investments, increased computational powers, small-sized components and the Internet of Things (IoT) have paved the way for technology such as self-parking cars, agriculture robots, helicopter drones and autonomous underwater vehicles. In recent years, the shipping industry has joined this revolution. Now, several test projects for autonomous vessels are emerging, such as the autonomous, unmanned container ship Yara Birkeland [1] and the electric, short-sea vessel Revolt developed by DNV-GL [2], both illustrated in Figure 1.1.

Figure 1.1: Left: The autonomous container-vessel Yara Birkeland, developed by Kongsberg Maritime for the Yara company. ©Kongsberg Maritime **Right**: ReVolt, an autonomous, fully battery-powered vessel designed for the short-sea shipping segment. ©DNV-GL

The constant growth of the human population in combination with infrastructural challenges has also sparked the idea of using urban waterways for transportation. The idea is

to let a small, electric and fully autonomous vessel take advantage of the unused potential for transportation on these waterways. Such a vessel will serve as a cheap and environmentally friendly alternative to transportation on land.

Motivated by this, the milliAmpere experimental vessel has been developed at the Norwegian University of Science and Technology (NTNU). It serves as a development platform for systems needed on a fully autonomous vessel, and incorporates everything from thrust allocation and collision avoidance, to interactive design and regulations. When finished, the ferry will be capable of carrying at least 12 passengers, and safely carry them over the waterways, while adapting to the surrounding boat traffic [3]. It will serve as on-demand ferry in the channel of Trondheim, saving commuters for a 15-minute walk to a nearest bridge. Illustrations of the final vessel and ferry route is given in Figure 1.2

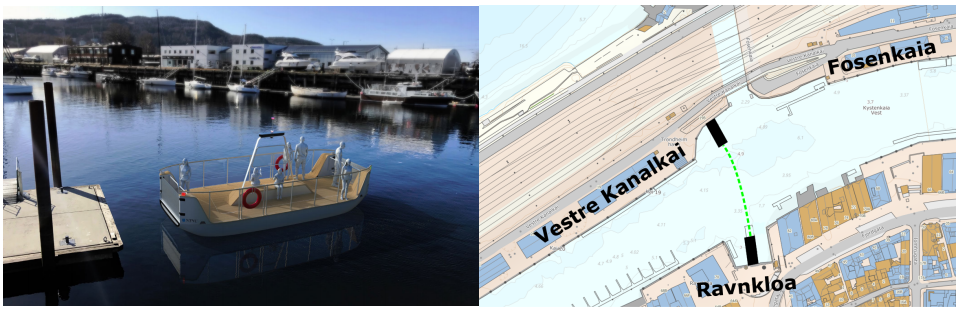


Figure 1.2: Left: A conceptual illustration of the autonomous passenger ferry intended to operate as a floating bridge in Trondheim. ©Petter Mustvedt **Right**: Map showing the planned route of the autonomous milliAmpere ferry in Trondheim.

For the milliAmpere ferry and other autonomous marine vessels, the docking part of the operation will be important to master. Several companies have demonstrated automatic docking capabilities for vessels, such as Kongsberg Marititme [4] and Volvo Penta [5], illustrated in Figure 1.3, but few details of these methods are available to the public. The procedure of automatic docking is challenging and requires high precision and maximum safety, and increases in complexity as regulations on velocity and power usage must be taken into account. In addition, the docking method must be adaptable to different scenarios, being able to navigate in difficult harbour layouts with obstacles and restricted zones present. The development of such a method is what motivated the work of this thesis.

Figure 1.3: Left: The electrical ferries "Gloppefjord" and "Eidsfjord" serving the Anda-Lote connection in western Norway are fitted with an autodocking system, developed by Rolls-Royce, now a part of Kongsberg Maritime. ©Fjord 1 **Right**: Volvo Penta has developed and unveiled a self-docking technology for leisure boats and yachts. ©Volvo Penta

1.2 Related Work

The challenge of automatic docking started gaining interest in the 1990s. A number of different strategies were tested, with Yamato [6] and Djouani and Hamam [7] exploring the usage of Artificial Neural Networks (ANN), Rae and Smith [8] proposing a control system based on fuzzy logic for an underwater vehicle, and Le et al. [9] suggesting a Multivariable Sliding Mode Controller (MSMC) to control the ship's pose in harbour areas.

The idea of solving the automatic docking problem as an OCP also emerged in this period. Shouji et al. [10] formulated the task as a two-point boundary value problem, by specifying conditions on initial and final state variables, rudder, propeller, side thruster actuation and time normalization. Djouani and Hamam [11] optimized the docking operation by proposing a ship optimal path planning method that included the minimization of a time-energy cost function and bounds on states and inputs. Later, the OCP strategy has been the center of several studies. Mizuno et al. [12] proposed a Model Based Predicitive Controller (MBPC) for tracking a minimum time optimal trajectory while Martinsen et al. [13] used a Nonlinear Model Predictive Controller (NMPC) to calculate a time- and energy-efficient docking trajectory. This method was capable of avoiding land constraints, and the problem was solved using the direct collocation method.

The OCP strategy for trajectory generation was also explored in Bitar et al. [14], There, the shortest path was found using an A*-based algorithm, and an optimal trajectory was generated for a 3 Degree of Freedom (DOF) under-actuated model of an ASV. The effectiveness of warm-starting of OCPs was also demonstrated. Still, there is a limited number of studies investigating automatic docking for fully-actuated vessels in obstacle-filled, non-convex harbours.

1.3 Problem Description

The objective of this thesis is to develop and implement an automatic docking method on the milliAmpere ferry. The problem description can be summarized as follows:

- Develop a trajectory planner for use in automatic docking operations.
- Find a method for representing the map land polygons as obstacle constraints in the trajectory planner.
- Implement this trajectory planner in the milliAmpere ferry system.
- Create graphical tools that allow for simple debugging during the development process, as well as verification and safety during the actual docking process.
- Develop an A*-inspired method for creating a feasible path that can be used as an initial guess to the trajectory planner.
- Perform simulations and sea trials showing the capabilities of the trajectory planner in terms of feasibility, obstacle avoidance, energy efficiency, time consumption and experienced passenger comfort.

1.4 Contributions

The contributions of this thesis include:

- Further development of an automatic docking method based on optimal control-based trajectory planning.
- A method for generating OCP-constraints from JSON map polygons to provide obstacle avoidance capabilities.
- An A* algorithm used for warm-starting of the docking method.
- Implementation of the method on a Robot Operating System (ROS)-based computer system for the milliAmpere experimental ferry.
- A Graphical User Interface (GUI) for the experimental ferry, including reference monitoring, map trajectory visualization and power consumption plots.
- Simulations and sea trials on the milliAmpere ferry, demonstrating the efficiency and feasibility of the docking method.

1.5 Outline

Chapter 2 presents the theory needed in order to understand the basic concepts of automatic docking and OCP's. Chapter 3 covers the development of the trajectory planner and how it is implemented as a part of the vessel system. The development of the graphical user

interface is described in Chapter 4, while Chapter 5 presents the results from simulations and sea trials. This chapter also defines performance metrics, comparing methods and docking scenarios. Chapter 6 concludes the thesis and proposes further work, while the appendices provide information regarding filtering of voltage and current measurements and the test plan for the sea trials.

Chapter 2

Background Theory

This thesis covers the development of an automatic docking method. To be able to follow this process, the mathematical notation and concepts in use must be understood. In this chapter, a brief introduction to the necessary ideas will be given. First, the vessel dynamics and notations will be explained. Then, optimization theory is presented. Finally, information regarding control allocation, obstacle representation, path planning and a common operating system on a vessel will be provided. Most of the material in this chapter is taken from the preceding project thesis [15].

2.1 Mathematical Modelling

To develop controllers, simulations and trajectory planners, the vessel must be properly described mathematically. This section, taken from [15], describes a decoupled vessel model, and the matrices, vectors and frames it encompasses.

2.1.1 Ship Kinematics and Kinetics

The marine vessel notation used in this thesis is listed in Table 2.1 and is defined according to [16].

An illustration of the Degrees of Freedom (DOF) is found in Figure 2.1. For most conventional ships, and especially during docking, it can be assumed that the heave z, roll ϕ and pitch θ motions are very small, and their dynamics can be neglected. By doing so, a 3DOF horizontal plane ship model based on maneuvering theory can be constructed using the framework given in Fossen [17]. The horizontal motion of a ship will be described by the motion components in surge, sway and yaw, resulting in the following vessel model:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{2.1}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_{\text{cl}} \tag{2.2}$$

DOF		Forces and moments	Linear and angular moments	Positions and Euler angles
1	motions in the x direction (surge)	X	u	x
2	motions in the y direction (sway)	Y	v	y
3	motions in the z direction (heave)	Z	w	z
4	rotation about the x axis(roll, heel)	K	p	ϕ
5	rotation about the y axis (pitch, trim)	M	q	θ
6	rotation about the z axis (yaw)	N	r	ψ

Table 2.1: The notation for marine vessels defined by the Society of Naval Architects and Marine Engineers (SNAME)

For this model

$$\boldsymbol{\eta} = \begin{bmatrix} x, y, \psi \end{bmatrix}^\top \in \mathbb{R}^3 \tag{2.3}$$

denotes the position and orientation vector in the Earth-fixed North-East-Down (NED) frame, while the linear and angular velocity vector

$$\boldsymbol{\nu} = \begin{bmatrix} u, v, r \end{bmatrix}^\top \in \mathbb{R}^3 \tag{2.4}$$

and the control force and moment vector

$$\boldsymbol{\tau}_c = \begin{bmatrix} X, Y, N \end{bmatrix}^\top \in \mathbb{R}^3 \tag{2.5}$$

are decomposed in the body fixed frame. The environmental loads $\boldsymbol{\tau}_{wind} \in \mathbb{R}^3$ are also specified in the body-frame. The rotation matrix $\boldsymbol{R}(\psi) \in \mathbb{R}^{3\times3}$, which rotates a vector defined in the body frame into the NED-frame, is given by

$$\boldsymbol{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

If the vessel is assumed symmetric with the center of the body-frame set in the centerline of the craft, the system inertia matrix $\boldsymbol{M} \in \mathbb{R}^{3\times3}$ is symmetric. It consists of the rigid body mass $\boldsymbol{M}_{RB} \in \mathbb{R}^{3\times3}$ and the hydrodynamic added mass $\boldsymbol{M}_A \in \mathbb{R}^{3\times3}$. Due to symmetry considerations, surge is decoupled from sway and yaw, and the inertia matrix has the resulting structure of

$$\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{32} & m_{33} \end{bmatrix} \tag{2.7}$$

Similarly, the Coriolis matrix $\boldsymbol{C}(\boldsymbol{\nu}) \in \mathbb{R}^{3\times3}$ depends on the rigid body Coriolis mass matrix $\boldsymbol{C}_{RB} \in \mathbb{R}^{3\times3}$ and the added mass Coriolis matrix $\boldsymbol{C}_A \in \mathbb{R}^{3\times3}$. The symmetry

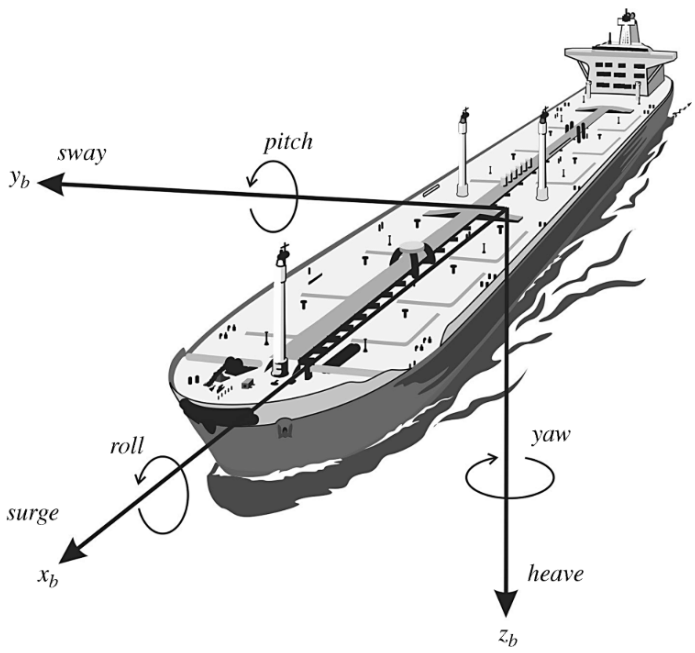


Figure 2.1: Illustration of motion in 6 DOF from [17]. ©2011 John Wiley Sons

considerations give a resulting skew-symmetric Coriolis matrix of

$$\boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C}_{RB} + \boldsymbol{C}_A = \begin{bmatrix} 0 & 0 & c_{13}(\boldsymbol{\nu}) \\ 0 & 0 & c_{23}(\boldsymbol{\nu}) \\ -c_{13}(\boldsymbol{\nu}) & -c_{23}(\boldsymbol{\nu}) & 0 \end{bmatrix} \tag{2.8}$$

where the elements correspond to

$$c_{13}(\boldsymbol{\nu}) = -m_{22}v - m_{23}r \tag{2.9a}$$
$$c_{23}(\boldsymbol{\nu}) = m_{11}u \tag{2.9b}$$

Finally, the damping matrix $\boldsymbol{D} \in \mathbb{R}^{3\times 3}$ is

$$\boldsymbol{D}(\boldsymbol{\nu}) = \begin{bmatrix} d_{11}(\boldsymbol{\nu}) & 0 & 0 \\ 0 & d_{22}(\boldsymbol{\nu}) & d_{23}(\boldsymbol{\nu}) \\ 0 & d_{32}(\boldsymbol{\nu}) & d_{33}(\boldsymbol{\nu}) \end{bmatrix} \tag{2.10}$$

where the elements are described by

$$d_{11} = -X_u - X_{|u|u}|u| - X_{uuu}u^2 \tag{2.11a}$$
$$d_{22} = -Y_v - V_{|v|v}|v| - Y_{|r|v}|r| - Y_{vvv}v^2 \tag{2.11b}$$
$$d_{23} = -Y_r - Y_{|v|r}|v| - Y_{|r|r}|r| \tag{2.11c}$$
$$d_{32} = -N_v - N_{|v|v}|v| - N_{|r|v}|r| \tag{2.11d}$$
$$d_{33} = -N_r - N_{|v|r}|v| - N_{|r|r}|r| - N_{rrr}r^2 \tag{2.11e}$$

2.2 Optimization Theory

The selection of a best element from a set of available alternatives, known as mathematical optimization, has been the focus of research in mathematics for centuries. These kinds of problems arise in numerous engineering disciplines, and marine vessel control and automatic docking is no exception. Optimization theory has applications in control allocation, motion controllers and trajectory planning. The following section, taken from [15], will explain some basic concepts connected to optimization problems.

In general, optimization models aim to maximize or minimize a performance criterion, subject to constraints that define the feasible operation space [18]. Thus, the most general formulation of optimization problems can be stated as

$$\text{Minimize} \quad f(\boldsymbol{x}) \tag{2.12a}$$
$$\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{x}) \leq 0 \tag{2.12b}$$
$$\boldsymbol{h}(\boldsymbol{x}) = 0 \tag{2.12c}$$
$$\boldsymbol{x} \in \mathbb{R}^n \tag{2.12d}$$

where $f(\boldsymbol{x})$ is the objective function, $\boldsymbol{g}(\boldsymbol{x})$ give the inequality constraints and $\boldsymbol{h}(\boldsymbol{x})$ define the equality constraints. Now, if the objective function is convex, $\boldsymbol{g}(\boldsymbol{x})$ is concave and $\boldsymbol{h}(\boldsymbol{x})$ is linear, the problem belongs to the field of *Convex Programming*. Such programs have the important property that every local minimum is also a global minimum and the problem can be easily solved using active-set methods.

For a non-convex problem, it can be quite hard to find the global optimum as illustrated in Figure 2.2. After all, many optimization problems cannot be expressed in terms of convex functions. This includes the vessel model (2.1) where the rotation matrix $\boldsymbol{R}(\psi)$ is nonlinear of nature. Still, such problems can be formulated and solved using Nonlinear Programming (NLP). Different strategies are used for solving these problems, one of them being the Sequential Quadratic Programming (SQP) method. SQP is an iterative method, that can be used when the objective function and the constraints are twice differentiable. In each iteration, the original problem is modeled by a quadratic programming subproblem, by approximating the objective function as a quadratic polynomial and linearizing the constraints. The solution of this subproblem is then used to propose a new iteration that moves towards the optimal solution.

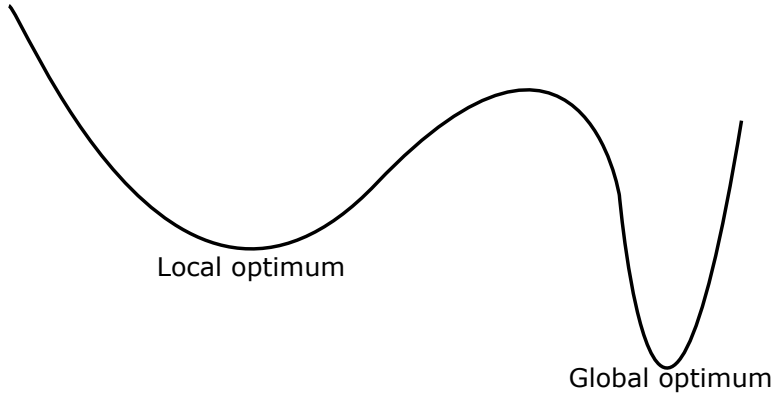


Figure 2.2: For a non-convex function, any local optimal solution is not necessarily a global optimal solution.

Another possibility for solving NLPs, is using an Interior Point Method (IPM). As the name indicates, these algorithms approach a solution by travelling on the interior (or exterior) of the feasible region, and not on the boundary. A logarithmic barrier function is introduced, and its gradient along with the gradient of the objective function is used to determine the search direction. An example of an IPM is the Interior Point Optimizer (IPOPT) developed by Wächter and Biegler [19]. This algorithm is made for the purpose of optimizing large scale nonlinear continuous systems, and is a good alternative for trajectory planning problems.

2.3 Motion Control

The following section, taken from [15], provides necessary information related to motion control of vessels. Developing a good trajectory planner for docking is only useful if the desired setpoints can be reached within the given time. To manage this, the vessel needs to have a functioning motion control system. The development of the gyrocompass and local and global positioning systems have paved the way for guidance systems that are capable of controlling vehicles without direct or continuous human control. Recently, waypoint-tracking and path-following control systems have grown popular, and are normally classified according to:

- *Setpoint regulation*, where the objective is to maintain a specific position or attitude, known as the pose.
- *Trajectory tracking*, where the system is desired to follow a specific output, dependent on time.
- *Path following*, where the goal is to follow a pre-defined path independent of time.

2.3.1 Trajectory Tracking

Tracking a smooth, time-varying trajectory reference $\boldsymbol{\eta}_d(t) \in \mathbb{R}^3$ is achieved by minimizing the tracking error:

$$\tilde{\boldsymbol{\eta}}(t) := \boldsymbol{\eta}(t) - \boldsymbol{\eta}_d(t) = \begin{bmatrix} x(t) - x_d(t) \\ y(t) - y_d(t) \\ \psi(t) - \psi_d(t) \end{bmatrix} \tag{2.13}$$

Fully actuated vessels with more than three control inputs ($\boldsymbol{u}(t) \in \mathbb{R}^i, i \geq 3$) are capable of minimizing the tracking error by using the controllers given in section 2.3.2.

The trajectory is usually generated by a reference model that converts a given waypoint into a feasible track. This is done by taking into account limitations on speed, acceleration and input. In [17], such a reference model, motivated by mass-spring-damper systems, is stated as:

$$\boldsymbol{M}_d\ddot{\boldsymbol{\eta}}_d + \boldsymbol{D}_d\dot{\boldsymbol{\eta}}_d + \boldsymbol{G}_d\boldsymbol{\eta}_d = \boldsymbol{G}_d\boldsymbol{r} \tag{2.14}$$

where $\boldsymbol{M}_d$,$\boldsymbol{D}_d$ and $\boldsymbol{G}_d$ are positive definite design matrices $\in \mathbb{R}^{3x3}$ giving the dynamics of the system while $\boldsymbol{r} \in \mathbb{R}^3$ is the desired reference point.

2.3.2 Dynamic Positioning

In Dynamic Positioning (DP), the control objective is station keeping or low-speed maneuvering, something that makes such controllers suitable for automatic docking operations. A low-speed model suitable for DP control is:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{2.15}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{D}\boldsymbol{\nu} = \boldsymbol{\tau}_c \tag{2.16}$$

This model is a simplified version of (2.1), where the Coriolis matrix $\boldsymbol{C}(\boldsymbol{\nu})$ has been removed and the damping matrix $\boldsymbol{D}(\boldsymbol{\nu})$ has been assumed constant. This simplification is valid for low-speed applications, since the quadratic and bilinear terms of the matrices become very small. With such a model, the linear multivariable Proportional-Derivative (PD) controller:

$$\boldsymbol{\tau}_c = -\boldsymbol{G}_p\boldsymbol{e}_2 - \boldsymbol{G}_d\boldsymbol{e}_1 \tag{2.17}$$

with error vectors

$$\boldsymbol{e}_2 = \boldsymbol{R}^\top(\psi)\left[\boldsymbol{\eta} - \boldsymbol{\eta_d}\right]^\top \tag{2.18}$$

$$\boldsymbol{e}_1 = \dot{\boldsymbol{e}}_2 \tag{2.19}$$

has the capability of controlling the vessel. The matrices $\boldsymbol{G}_p, \boldsymbol{G}_d \in \mathbb{R}^{3x3} \succ 0$ can be found by appropriate control synthesis methods, for instance by forming a linear quadratic performance problem

$$J = \min \int_0^T \boldsymbol{e}^\top\boldsymbol{Q}\boldsymbol{e} + \boldsymbol{\tau}_c^\top\boldsymbol{P}\boldsymbol{\tau}_c dt \tag{2.20}$$

that is solvable through the Ricatti equation. Here, the matrices $\boldsymbol{Q}, \boldsymbol{P} \succ 0$ penalizes the state errors $\boldsymbol{e} = [e_1^\top, e_2^\top]^\top$ and the control input $\boldsymbol{\tau}_c$ respectively. These matrices are often chosen to be diagonal, letting each matrix insert correspond to the penalization of only one vector element. Integral action and reference feed-forward terms can also be added to the controller [20].

A similar alternative is the Proportional-Integral-Derivative (PID) controller with acceleration feedback

$$\begin{aligned}\boldsymbol{\tau}_c = &- \boldsymbol{R}^\top\big(\psi(t)\big)\boldsymbol{K}_p\big(\boldsymbol{\eta}(t) - \boldsymbol{\eta_d}(t)\big) - \boldsymbol{K}_d\big(\boldsymbol{\nu}(t) - \boldsymbol{\nu_d}(t)\big) \\ &- \boldsymbol{R}^\top\big(\psi(t)\big)\boldsymbol{K}_i \int_0^t \big(\boldsymbol{\eta}(\tau) - \boldsymbol{\eta_d}(\tau)\big)d\tau + \boldsymbol{K}_a\boldsymbol{a}_d(t)\end{aligned} \tag{2.21}$$

proposed in [17], where $\boldsymbol{\nu}_d \in \mathbb{R}^3$ and $\boldsymbol{a}_d \in \mathbb{R}^3$ are the desired velocities and accelerations in the body frame, while $\boldsymbol{K}_p, \boldsymbol{K}_d, \boldsymbol{K}_i, \boldsymbol{K}_a \in \mathbb{R}^{3x3} \succ 0$ are tunable controller parameters.

2.4 Control Allocation

Having a well functioning motion controller is only useful if the thrusters of the vessel can produce the demanded forces and moments at the right time instances. This is known as *Control Allocation*. The following section, taken from [15], goes deeper into this field of research. For ships operating in n DOF, the process of mapping the desired force vector $\boldsymbol{\tau}_\mathrm{c} \in \mathbb{R}^n$ from the controller into r number of control inputs $\boldsymbol{u} \in \mathbb{R}^r$ is known as the control allocation problem. The control problem is called *underactuated* if $r < n$ and *overactuated* if $r > n$, and examples of control inputs can be thruster output or azimuth orientation. A ship modeled in 3-DOF (2.1) typically has the input mapping

$$\boldsymbol{\tau}_c = \boldsymbol{T}\big(\boldsymbol{\alpha}_{az}\big)\boldsymbol{f}, \qquad \boldsymbol{f} = \boldsymbol{K}\boldsymbol{u} \tag{2.22}$$

Here, the configuration matrix $\boldsymbol{T}(\boldsymbol{\alpha}_{az}) \in \mathbb{R}^{3\times r}$ is dependent on the azimuth thruster angles $\boldsymbol{\alpha}_{az}$. The thruster force vector $\boldsymbol{f} \in \mathbb{R}^r$ is determined by the control inputs $\boldsymbol{u}$ and the gain matrix $\boldsymbol{K} \in \mathbb{R}^{r\times r}$. For marine vessels, control allocation problems can usually be formulated as optimization problems. The objective is typically to minimize the use of control effort while taking into account actuator rate, position and other operational constraints. For the control allocation algorithms explained below, the control actuators are assumed to be azimuth thrusters.

2.4.1 Nonlinear Constrained Control Allocation for Azimuth Thrusters

For a marine craft with rotatable azimuth thrusters, the control allocation problem is non-convex and quite hard to solve. The main constraint is

$$\boldsymbol{\tau}_c = \boldsymbol{T}\big(\boldsymbol{\alpha}_{az}\big)\boldsymbol{f} \tag{2.23}$$

where the azimuth angles $\boldsymbol{\alpha}_{az}$ must be computed at each sample along with the control inputs $\boldsymbol{u}$ which are subject to amplitude and rate saturations. The azimuth thrusters must

also operate within feasible sectors $\alpha_{i,min} \leq \alpha_i \leq \alpha_{i,max}$ and with a limited turning rate $\dot{\alpha}_i$. A proposed minimization problem is given in [21] as:

$$J = \min_{\boldsymbol{f},\boldsymbol{\alpha},\boldsymbol{s}} \{ \sum_{i=1}^{r} \bar{\boldsymbol{P}}_{\boldsymbol{i}} |f_i|^{3/2} + \boldsymbol{s}^\top \boldsymbol{Q} \boldsymbol{s} + \big(\boldsymbol{\alpha_{az}} - \boldsymbol{\alpha_{az,0}}\big)^\top \boldsymbol{\Omega} \big(\boldsymbol{\alpha_{az}} - \boldsymbol{\alpha_{az,0}}\big) + \frac{\varrho}{\epsilon + \det\big(\boldsymbol{T}\big(\boldsymbol{\alpha_{az}}\big) \boldsymbol{W^{-1}} \boldsymbol{T}^\top \big(\boldsymbol{\alpha_{az}}\big)\big)} \}$$

subject to: (2.24)

$$\begin{aligned} &\boldsymbol{T}\big(\boldsymbol{\alpha_{az}}\big)\boldsymbol{f} = \boldsymbol{\tau_c} + \boldsymbol{s} \\ &\boldsymbol{f}_{\text{min}} \leq \boldsymbol{f} \leq \boldsymbol{f}_{\text{max}} \\ &\boldsymbol{\alpha}_{\text{min}} \leq \boldsymbol{\alpha_{az}} \leq \boldsymbol{\alpha}_{\text{max}} \\ &\boldsymbol{\Delta\alpha}_{\text{min}} \leq \boldsymbol{\alpha_{az}} - \boldsymbol{\alpha_{az,0}} \leq \boldsymbol{\Delta\alpha}_{\text{max}} \end{aligned}$$

where

- $\sum_{i=1}^{r} \bar{\boldsymbol{P}}_{\boldsymbol{i}} |f_i|^{3/2}$ represents power consumption where $\bar{P}_i > 0 \big(i = 1, ..., r\big)$ are positive weights
- $\boldsymbol{s}^\top \boldsymbol{Q} \boldsymbol{s}$ punishes deviation from the commanded and achieved force. This guarantees that the problem has a feasible solution for any $\boldsymbol{\tau}_c$ and $\boldsymbol{\alpha_{az,0}}$. $\boldsymbol{Q}$ should be chosen large enough to ensure $\boldsymbol{s} \approx \boldsymbol{0}$ whenever possible.
- $\boldsymbol{f}_{\text{min}} \leq \boldsymbol{f} \leq \boldsymbol{f}_{\text{max}}$ is used to handle saturation issues
- $\boldsymbol{\alpha}_{\text{min}} \leq \boldsymbol{\alpha_{az}} \leq \boldsymbol{\alpha}_{\text{max}}$ gives the feasible sector of the azimuth angles.
- $\boldsymbol{\Delta\alpha}_{\text{min}} \leq \boldsymbol{\alpha_{az}} - \boldsymbol{\alpha_{az,0}} \leq \boldsymbol{\Delta\alpha}_{\text{max}}$ is a way to limit the turning rate $\dot{\boldsymbol{\alpha}}_{\boldsymbol{az}}$ of the azimuth thrusters. This is done by taking α_0 equal to the angle at the previous sample.
- The term $\frac{\varrho}{\epsilon + \det\big(\boldsymbol{T}\big(\boldsymbol{\alpha_{az}}\big) \boldsymbol{W^{-1}} \boldsymbol{T}^\top \big(\boldsymbol{\alpha_{az}}\big)\big)}$ is used to avoid singular configurations.

This optimization problem is nonconvex and nonlinear and thus require a substantial amount of computations for each sample. It is usually solved through iterations, and several implementation strategies exist.

2.4.2 Nonlinear Scalar Control Allocation

For double ended ferries with symmetric design and symmetric thruster configuration, the Nonlinear Scalar Control Allocation (NSCA) [22] is a solution to the control allocation problem. For a ship with a front F_1 and aft F_2 thruster, the thrust is decomposed into XY-components, namely $F_{1,x}$,$F_{1,y}$,$F_{2,x}$ and $F_{2,y}$ as illustrated in Figure 2.3. With X, Y and N being the surge force, sway force and yaw moment demanded from the controller,

the control allocation equations to solve are given as:

$$F_{1,x} + F_{2,x} = X \tag{2.25}$$

$$F_{1,y} = \frac{N + L_x Y}{2L_x} \tag{2.26}$$

$$F_{2,y} = -\frac{N - L_x Y}{2L_x} \tag{2.27}$$

Thus, $F_{1,y}$ and $F_{2,y}$ are uniquely determined, while (2.25) only have one degree of freedom. This makes the optimization problem scalar and bounded, and $F_{1,x}$ can be found with fast and robust nonlinear solvers. Bounds on maximum demanded thrust, penalization of deviation from thruster home angles and change in azimuth angle can also be accounted for in the cost function and constraints.

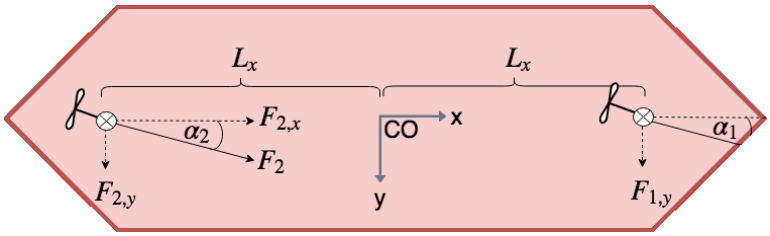


Figure 2.3: Thruster configuration for double-ended ferries. From [22]. ©2019 CAMS

The NSCA process outperforms other optimization based control allocation algorithms in terms of computational speed, and shows very promising real-time performance for use in a DP system [22].

2.5 Transcription of Optimal Control Problems

A possible strategy for solving the problem of automatic docking is to formulate it as a general optimal control problem. Next, in order to solve such a problem, it must be transcribed into an NLP. This can be done with methods like single shooting and direct collocation, but this section, taken from [15], explains how the simultaneous method of *Direct Multiple Shooting* [23] can be used.

An OCP can be given as

$$J = \min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad \Phi\big[\boldsymbol{x}(t_0), t_0, \boldsymbol{x}(t_f), t_f\big] + \int_{t_0}^{t_f} \mathcal{L}\big[\boldsymbol{x}(t), \boldsymbol{u}(t), t\big]dt \tag{2.28a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{F}\big[\boldsymbol{x}(t), \boldsymbol{u}(t), t\big] \tag{2.28b}$$

$$\boldsymbol{h}_{ocp}\big[\boldsymbol{x}(t), \boldsymbol{u}(t), t\big] \leq \boldsymbol{0} \tag{2.28c}$$

$$\boldsymbol{\phi}_{ocp}\big[\boldsymbol{x}(t_0), t_o, \boldsymbol{x}(t_f), t_f\big] = 0 \tag{2.28d}$$

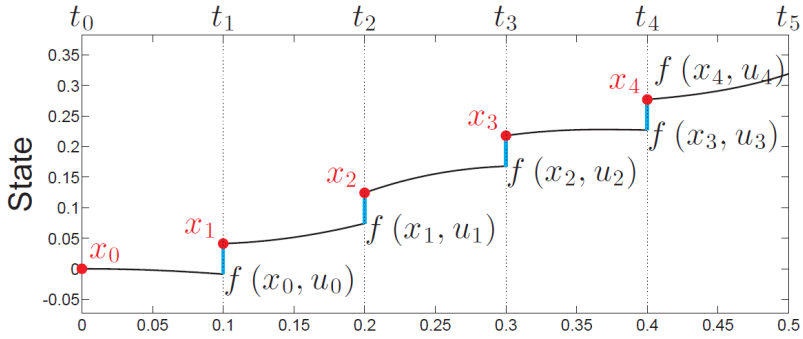


Figure 2.4: In multiple shooting, the simulation time is cut into time intervals, with integration being performed on these. ©2017 Gros

where $\boldsymbol{x}$ are the states, $\boldsymbol{u}$ are the control inputs, the terms Φ and $\mathcal{L}$ are called the endpoint cost and the Lagrangian respectively, $\boldsymbol{h}_{ocp}[\cdot]$ are the algebraic state constraints and $\boldsymbol{\phi}_{ocp}[\cdot]$ are the boundary conditions. The method of multiple shooting now handles this transformation by breaking the system integration down into short time intervals as illustrated in Figure 2.4. In this way, the integrating function $\boldsymbol{f}\left(\boldsymbol{x}_k, \boldsymbol{u}_{\boldsymbol{k}}\right)$ capturing the dynamic constraints of (2.28b), can be held arbitrarily linear. As the system is discretised on the time grid $\{t_0, t_1, ..., t_N\}$, $\boldsymbol{x}_{\boldsymbol{k}}$ and $\boldsymbol{u}_{\boldsymbol{k}}$ will become decision variables, and the shooting gaps:

$$\boldsymbol{f}\left(\boldsymbol{x}_k, \boldsymbol{u}_{\boldsymbol{k}}\right) - \boldsymbol{x}_{k+1} = 0, \quad k \in \left[0, 1, ..., N\right] \tag{2.29}$$

will be the constraints (2.30b) in the NLP:

$$\min_{\boldsymbol{w}} \quad \boldsymbol{\phi}(\boldsymbol{w}) \tag{2.30a}$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{w}) = \boldsymbol{0} \tag{2.30b}$$

$$\boldsymbol{h}(\boldsymbol{w}) \leq \boldsymbol{0} \tag{2.30c}$$

where $\boldsymbol{w} = [\boldsymbol{x}_0^\top, \boldsymbol{u}_0^\top, ..., \boldsymbol{x}_{\boldsymbol{N-1}}^\top, \boldsymbol{u}_{\boldsymbol{N-1}}^\top, \boldsymbol{x}_{\boldsymbol{N}}^\top]^\top$ and

$$\boldsymbol{g}(\boldsymbol{w}) = \begin{bmatrix} \bar{\boldsymbol{x}}_o - \boldsymbol{x}_0 \\ \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{u}_0) - \boldsymbol{x}_1 \\ ... \\ \boldsymbol{f}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) - \boldsymbol{x}_N \end{bmatrix}, \quad \boldsymbol{h}(\boldsymbol{w}) = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{x}_0, \boldsymbol{u}_0) \\ ... \\ \boldsymbol{h}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) \\ \boldsymbol{h}(\boldsymbol{x}_N) \end{bmatrix} \tag{2.31}$$

2.6 Obstacle Representation

In order to accommodate for different harbour layouts during automatic docking, it is necessary to have a way of representing obstacles. This section is taken from [15], and

explains how obstacle representation can be done.

The path constraints can be mathematically expressed by a differentiable algebraic function, and the *p-norm* is well suited for this:

$$\left\| \left(x, y\right) \right\|_p \triangleq \left(|x|^p + |y|^p \right)^{\frac{1}{p}}, \quad p = 1, 2, ... \tag{2.32}$$

As Figure 2.5 shows, this can be used to create diamond-, circle-, ellipse- and square obstacles, and the technique can be extended to cover more general polygon shapes [24]. In optimal control problems, these obstacles can be represented by the path constraints:

$$h(x, y) = \left\| \left(\frac{x - x_c}{a}, \frac{y - y_c}{b} \right) \right\|_p - c \geq 0 \tag{2.33}$$

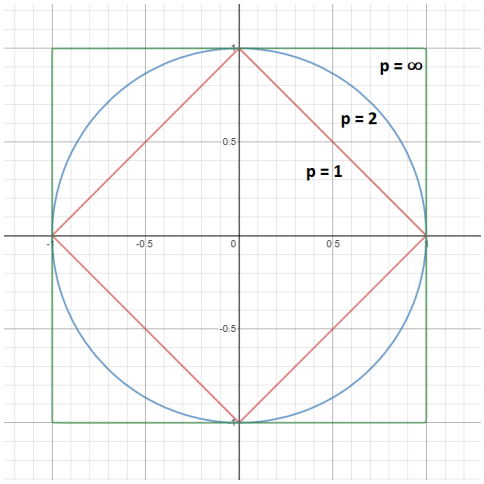


Figure 2.5: Resulting obstacles using p-norms 1, 2 and ∞.

2.6.1 Rectangular Rotatable Obstacle

A rectangular shaped obstacle with a given length, width and rotation will be quite suitable for automatic docking purposes, as it can take on the shapes of docks, piers and other vessels quite easily. From the equation of a rectangle in the Cartesian plane we have:

$$\left| \frac{x}{a} + \frac{y}{b} \right| + \left| \frac{x}{a} - \frac{y}{b} \right| = 2 \tag{2.34}$$

where a is the length in x-direction, and b is the length in y-direction. This rectangle can be moved to any position $\left(x_c, y_c\right)$ in the plane by defining the variable

$$\tilde{x} = \left(x - x_c\right) \tag{2.35}$$

$$\tilde{y} = \left(y - y_c\right) \tag{2.36}$$

Next, the rectangle can be rotated by an angle α if the variables $\bar{x}$ and $\bar{y}$ are defined as:

$$\bar{x} = \tilde{x}\cos\alpha - \tilde{y}\sin\alpha \tag{2.37}$$

$$\bar{y} = \tilde{x}\sin\alpha + +\tilde{y}\cos\alpha \tag{2.38}$$

giving a resulting equation of:

$$\left|\frac{\bar{x}}{a} + \frac{\bar{y}}{b}\right| + \left|\frac{\bar{x}}{a} - \frac{\bar{y}}{b}\right| = 2 \tag{2.39}$$

Thus, any rectangle-shaped obstacle in the Cartesian plane can be constructed by using the five parameters a, b, α, x_c and y_c as illustrated in Figure 2.6.

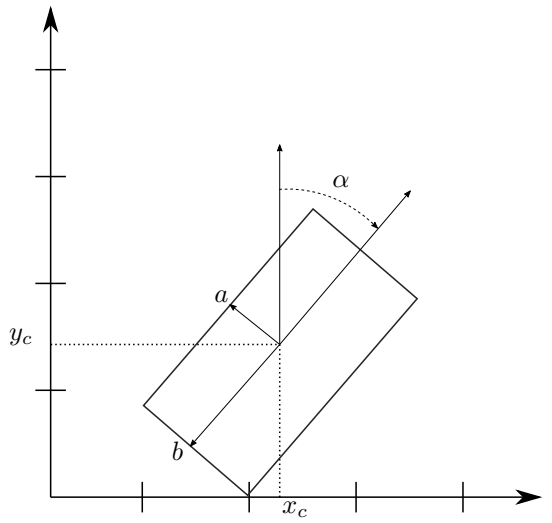


Figure 2.6: Parameters of a rectangular rotatable obstacle.

2.7 A* Search Algorithm

Many engineering and scientific problems involve finding a feasible path through an obstacle-filled environment. For the trajectory planner proposed in this thesis, this ability is needed to plan docking trajectories in complex harbours. The following section gives an introduction to the search algorithm that is used.

In graph theory, finding the shortest route between two points is known as the pathfinding problem. One of the first solutions to this problem was Dijkstra's algorithm, and later, several other methods have followed. In this thesis, the *A*-algorithm* is the one in use, due to its completeness, optimality and efficiency [25].

The A*-algorithm works by representing all the possible vertices as *nodes*. The algorithm finds the shortest path between the start and end node by keeping track of the nodes

that have been discovered and the nodes that have been explored. When starting the algorithm, the only discovered node is the starting node. Then, for each iteration, a new node is selected for exploration from the list of discovered nodes. The selection of this node is what separates A* from Dijkstra's algorithm. In the A*-alorithm, the node n to explore is the one having the smallest cost

$$f(n) = g(n) + h(n) \tag{2.40}$$

consisting of the cost of the path leading to the node $g(n)$ plus the cost estimated for reaching the end node $h(n)$. With $h(n)$ being a problem-specific heuristic function, the algorithm offers great flexibility and is guaranteed to return a least-cost path from start to goal. When a node is explored, all of its non-explored neighbouring nodes are updated with new costs and added to the list of discovered nodes, if not previously discovered. Eventually, the end node will be discovered and the shortest path will be found by having the nodes keep track of their predecessor.

For the docking trajectory planner used in this thesis, the shortest path problem is solved by representing each integer NED-position as a node. The nodes that are inside obstacles are removed, and the discovery of nodes is done as shown in Figure 2.7. This discovery strategy allows for smoother and shorter paths than if only the 8 adjacent nodes were discovered.

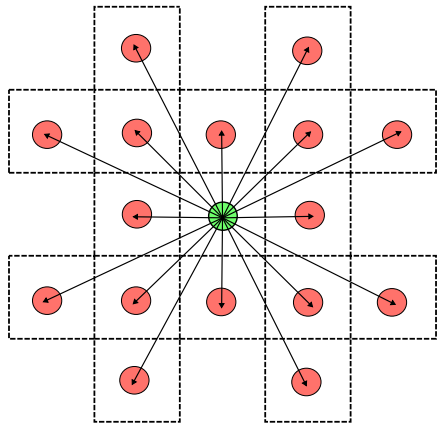


Figure 2.7: Discovery strategy of the A* algorithm used in this thesis. The explored node is illustrated as a green circle while the neighbouring nodes are in red.

2.8 Robot Operating System

The trajectory planner developed in this thesis is implemented as a part of the milliAmpere experimental vessel system. As milliAmpere uses the Robot Operating System (ROS) to organize and communicate between the different parts of the vessel, the following section will give a short introduction to the most important concepts in ROS.

As mentioned, ROS is a framework for robotic systems, allowing for the design of complex applications without knowing how every piece of hardware works. It is not an operating system, but it provides the tools and services to allow for similar functionality. The first version of ROS was released in 2010, and since then a number of new distributions have been released.

In ROS, the various processes are represented as *nodes* in a graph structure. Figure 2.8 gives an example of this structure. The nodes run as single processes, and communicate with other nodes through uniquely named *topics*. A node must *publish* a message to the topic to send information, and other nodes may *subscribe* to the topic in order to receive those messages. Topic examples in milliAmpere include the position of the vessel, motor states from the thrusters and desired docking waypoint. ROS also gives the nodes the possibility of having *services*. These services represent actions that gives results, and the nodes can advertise their own services or call services from other nodes. A service in milliAmpere is for instance the opportunity to change the docking mode of the vessel. The *parameter server* is also an important concept of ROS. It works as a shared database between the nodes where not frequently changed parameters can be stored. With this structure, ROS also provides a great way of creating simulators. By replacing the physical components with simulated nodes, the application can be safely tested without having to change the rest of the software. This has been done for milliAmpere, allowing new functionality to be tested in simulation and sea trials using the same code.

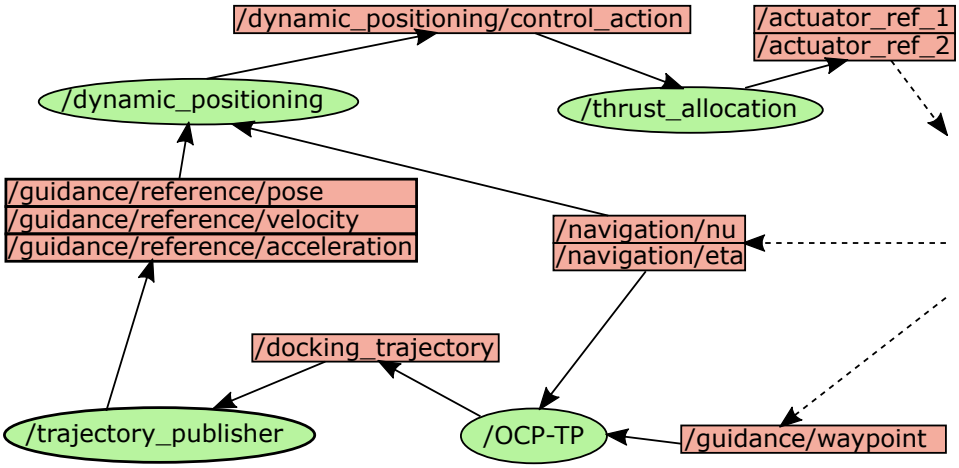


Figure 2.8: Example of the ROS graph structure, showing how the different nodes in the milliAmpere experimental vessel publish and subscribe to the various topics. Nodes are shown as green ellipses and topics as red rectangles. The arrow from node to topic indicates publishing and arrow from topic to node indicates subscription.

Chapter 3

The Trajectory Planner for Docking

In the preceding project report [15], an energy efficient optimization based trajectory planner for docking was proposed. In this thesis, this trajectory planner is further refined and realized as a docking method in a full-scale experimental vessel. This method will from now on be referred to as the Optimal Control Problem Trajectory Planner (OCP-TP). This chapter covers the development of the OCP-TP, from the mathematical equations of the OCP model, to how it is implemented in the actual vessel with warm-starting algorithms, obstacle creation from map polygons and trajectory publishing.

3.1 The OCP Model

The optimal control based trajectory planner presented in the preceding project report [15], is the heart of the OCP-TP docking method. The material in this section is taken from the preceding project report [15], and the recap covers the vessel model, the cost function, the obstacle constraints, and the transcription of the OCP into a solvable NLP.

3.1.1 Vessel Model

The vessel model of the OCP uses the state-space representation given in Section 2.1.1. It has the form

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{3.1}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_c(\boldsymbol{u}) \tag{3.2}$$

with rotation matrix

$$\boldsymbol{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

The pose vector $\boldsymbol{\eta} = [x, y, \psi]^\top$ is defined in the North-East-Down (NED)-frame and the velocity vector $\boldsymbol{\nu} = [u, v, r]^\top$ is defined in the body-frame.

The control force input is chosen as

$$\boldsymbol{u} = [F_{1,x}, F_{1,y}, F_{2,x}, F_{2,y}]^\top \tag{3.4}$$

having the forces from the two azimuth thrusters being decoupled in the x and y-directions as illustrated in Figure 2.3. With the corresponding thrust configuration matrix

$$\boldsymbol{T} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & L_x & 0 & -L_x \end{bmatrix} \tag{3.5}$$

the complete control input force vector $\boldsymbol{\tau}_c(\boldsymbol{u})$ is mapped as

$$\boldsymbol{\tau}_c(\boldsymbol{u}) = \boldsymbol{T}\boldsymbol{u} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & L_x & 0 & -L_x \end{bmatrix} \begin{bmatrix} F_{1,x} \\ F_{1,y} \\ F_{2,x} \\ F_{2,y} \end{bmatrix} = \begin{bmatrix} F_{1,x} + F_{2,x} \\ F_{1,y} + F_{2,y} \\ L_x F_{1,y} - L_x F_{2,y} \end{bmatrix} \tag{3.6}$$

Finally, by collecting the vessels states in the vector $\boldsymbol{x} = [\boldsymbol{\eta}^\top, \boldsymbol{\nu}^\top]^\top = [x, y, \psi, u, v, r]^\top$, a compact mathematical realization of the vessel model is found to be

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \boldsymbol{R}(\psi)\boldsymbol{\nu} \\ \boldsymbol{M}^{-1}(-\boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{\tau}_c(\boldsymbol{u})) \end{bmatrix} \tag{3.7}$$

3.1.2 Cost Function

The cost function of the OCP has the minimization of expended energy as the only objective and is defined as

$$J(\boldsymbol{u}(\cdot)) = \int_0^{t_f} \mathcal{L}(\boldsymbol{u}(\tau))d\tau \tag{3.8}$$

An energy effective method often gives very slow trajectories, so the end time t_f must be chosen carefully to limit the time usage. Next, minimizing the power consumption by controlling the control inputs $\boldsymbol{u}(\cdot)$ requires a mapping from thruster force to thruster power consumption. In [26], the equations for thrust T and power consumption P_{mech} are expressed by the shaft speed revolution n_s and the constants K_T, K_P dependent on the water density ρ and the propeller diameter D_p. These expressions are found to be

$$T = \text{sign}(n) K_T(\rho, D_p) n_s^2 \tag{3.9a}$$

$$P_{mech} = \text{sign}(n) K_P(\rho, D_p) n_s^3 \tag{3.9b}$$

and the power consumption can be expressed as a function of the thruster force

$$P_{mech} = K_{TP}|T|^{\frac{3}{2}} \tag{3.10}$$

which is proportional to

$$P_{mech} \propto |T|^{\frac{3}{2}} \tag{3.11}$$

when disregarding the approximately constant parameter K_{TP}.

With $\boldsymbol{u}$ as in (3.4), the momentary power consumption of the vessel can be described by

$$\mathcal{L}(\boldsymbol{u}(t)) = \sum_{i \in I} \left(F_{i,x}^2 + F_{i,y}^2 + \epsilon^2 \right)^{\frac{3}{4}}, \quad I = \{1, 2\} \tag{3.12}$$

where ϵ is a small variable enabling the cost function to be differentiable for all inputs $\boldsymbol{u}$.

3.1.3 Representation of Obstacle Constraints

For the functioning in complex harbour environments, mathematical rectangles are introduced in the problem to mimic obstacles. These rectangels are based on the equations given in Section 2.6.1 and are determined by the east position x_c, north position y_c, rectangle width a, rectangle length b and rectangle rotation α. By defining the function

$$\mathcal{C}(\boldsymbol{x}, x_c, y_c, a, b, \alpha) = \left| \frac{\bar{x}}{a} + \frac{\bar{y}}{b} \right| + \left| \frac{\bar{x}}{a} - \frac{\bar{y}}{b} \right| - 2 \tag{3.13}$$

the rectangle equations can be made into the desired obstacle constraints by including algebraic inequalities of the form

$$0 \leq \mathcal{C}(\boldsymbol{x}, x_c, y_c, a, b, \alpha) \leq \infty \tag{3.14}$$

3.1.4 From OCP to NLP

An OCP formulated according to (2.28) will have a solution giving a state trajectory $\boldsymbol{x}(\cdot)$ and desired inputs $\boldsymbol{u}(\cdot)$, that minimizes the cost function. By using the multiple shooting approach as described in section 2.5, the OCP is transformed into the nonlinear program (NLP):

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \boldsymbol{\phi}(\boldsymbol{w}) && \text{(3.15a)} \\ \text{s.t.} \quad & \boldsymbol{g}_{lb} \leq \boldsymbol{g}(\boldsymbol{w}) \leq \boldsymbol{g}_{ub} && \text{(3.15b)} \\ & \boldsymbol{w}_{lb} \leq \boldsymbol{w} \leq \boldsymbol{w}_{ub} && \text{(3.15c)} \end{aligned}$$

The dynamics are split into N timesteps of length $h = \frac{t_f}{N}$. This makes

$$\boldsymbol{w} = \begin{bmatrix} \boldsymbol{x}_0^\top & \boldsymbol{u}_0^\top & \boldsymbol{x}_1^\top & \dots & \boldsymbol{u}_{N-1}^\top & \boldsymbol{x}_N^\top \end{bmatrix}^\top \tag{3.16}$$

the combined vector of states and inputs for each timestep, with upper and lower bounds according to

$$\begin{aligned} \boldsymbol{w}_{lb} &= \begin{bmatrix} \boldsymbol{x}_{0,lb}^\top & \boldsymbol{u}_{lb}^\top & \boldsymbol{x}_{lb}^\top & \dots & \boldsymbol{u}_{lb}^\top & \boldsymbol{x}_{f,lb}^\top \end{bmatrix}^\top \\ \boldsymbol{w}_{ub} &= \begin{bmatrix} \boldsymbol{x}_{0,ub}^\top & \boldsymbol{u}_{ub}^\top & \boldsymbol{x}_{ub}^\top & \dots & \boldsymbol{u}_{ub}^\top & \boldsymbol{x}_{f,ub}^\top \end{bmatrix}^\top \end{aligned} \tag{3.17}$$

where

$$\boldsymbol{x}_{0,lb} = \boldsymbol{x}_{0,ub} = \boldsymbol{x}_0 \tag{3.18a}$$
$$\boldsymbol{x}_{f,lb} = \boldsymbol{x}_{f,ub} = \boldsymbol{x}_f \tag{3.18b}$$
$$\boldsymbol{x}_{lb} = [x_{lb}, y_{lb}, \psi_{lb}, u_{lb}, v_{lb}, r_{lb}]^\top \tag{3.18c}$$
$$\boldsymbol{x}_{ub} = [x_{ub}, y_{ub}, \psi_{ub}, u_{ub}, v_{ub}, r_{ub}]^\top \tag{3.18d}$$
$$\boldsymbol{u}_{lb} = [F_{1x,lb}, F_{1y,lb}, F_{2x,lb}, F_{2y,lb}]^\top \tag{3.18e}$$
$$\boldsymbol{u}_{ub} = [F_{1x,ub}, F_{1y,ub}, F_{2x,ub}, F_{2y,ub}]^\top \tag{3.18f}$$

The NLP cost function in (3.15a) is expressed as

$$\phi(\boldsymbol{w}) = J(t_N) \tag{3.19}$$

by calculating the accumulated cost $J(t_k)$ for each timestep as

$$J(t_k) = J(t_{k-1}) + h \cdot \mathcal{L}(\boldsymbol{u}(t_{k-1})) \tag{3.20}$$

using the Forward Euler method and the cost function given in (3.8).

The constraints in $\boldsymbol{g}(\boldsymbol{w})$ includes the shooting constraints of the model equation $\boldsymbol{g}_s(\boldsymbol{w})$, and the constraints imposed by the obstacles $\boldsymbol{g}_o(\boldsymbol{w})$.

Using the built-in integrator in the software package Casadi (see section 3.1.5), the discrete version of the dynamics in (3.7) is realized as

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{3.21}$$

giving the shooting constraints of

$$\boldsymbol{g}_s(\boldsymbol{w}) = \begin{bmatrix} \boldsymbol{x}_1 - \boldsymbol{F}(\boldsymbol{x}_0, \boldsymbol{u}_0) \\ \vdots \\ \boldsymbol{x}_N - \boldsymbol{F}(\boldsymbol{x}_{N-1}, \boldsymbol{u}_{N-1}) \end{bmatrix} \tag{3.22}$$

Forcing these equations to form a valid trajectory, corresponds to setting the bounds to

$$\boldsymbol{g}_{s,lb} = \boldsymbol{g}_{s,ub} = \boldsymbol{0} \tag{3.23}$$

Next, for each timestep *k*, the obstacle vector is defined as

$$\boldsymbol{g}_{o,k}(\boldsymbol{x}_k) = \begin{bmatrix} \mathcal{C}(\boldsymbol{x_k}, x_{c1}, y_{c1}, a_{c1}, b_{c1}, \alpha_{c1}) \\ \mathcal{C}(\boldsymbol{x_k}, x_{c2}, y_{c2}, a_{c2}, b_{c2}, \alpha_{c2}) \\ \vdots \\ \mathcal{C}(\boldsymbol{x_k}, x_{cn_o}, y_{cn_o}, a_{cn_o}, b_{cn_o}, \alpha_{cn_o}) \end{bmatrix} \tag{3.24}$$

with the number of different obstacles given by n_o. This results in all the timesteps being collected in the vector

$$\boldsymbol{g}_o(\boldsymbol{x}_k) = \begin{bmatrix} \boldsymbol{g}_{o,0}(\boldsymbol{x}_0) \\ \boldsymbol{g}_{o,1}(\boldsymbol{x}_1) \\ \vdots \\ \boldsymbol{g}_{o,N}(\boldsymbol{x}_N) \end{bmatrix} \tag{3.25}$$

where the bounds

$$\boldsymbol{g}_{o,lb} = 0, \quad \boldsymbol{g}_{o,ub} = \infty \tag{3.26}$$

enforce the obstacle constraint inequalities as define in (3.14).

After these definitions, the complete nonlinear constraints of (3.15b) will be captured by the equations

$$\boldsymbol{g}_{lb} = \begin{bmatrix} \boldsymbol{g}_{s,lb} \\ \boldsymbol{g}_{o,lb} \end{bmatrix}, \; \boldsymbol{g}(\boldsymbol{w}) = \begin{bmatrix} \boldsymbol{g}_s(\boldsymbol{w}) \\ \boldsymbol{g}_o(\boldsymbol{w}) \end{bmatrix}, \; \boldsymbol{g}_{ub} = \begin{bmatrix} \boldsymbol{g}_{s,ub} \\ \boldsymbol{g}_{o,ub} \end{bmatrix} \tag{3.27}$$

3.1.5 Solver and Building of Problem

In the preceding thesis [15], the problem and structure proposed in this section were realized in MATLAB by defining a MATLAB class. However, for this thesis, a class in Python is used. This class sets up the problem according to (3.15), by taking in the following variables:

- states $\boldsymbol{x}$
- control inputs $\boldsymbol{u}$
- cost function $\mathcal{L}(\boldsymbol{u})$
- system differential equations $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$
- obstacle restrictions $\mathcal{C}(\cdot)$
- upper- and lower bounds on states and inputs $\boldsymbol{x}_{lb}, \boldsymbol{x}_{ub}, \boldsymbol{u}_{lb}, \boldsymbol{u}_{ub}$
- initial and final states $\boldsymbol{x}_0, \boldsymbol{x}_f$
- desired end time t_f
- number of timesteps N

Using the optimal control software Casadi, an open-source tool for nonlinear optimization and algorithmic differentiation, the problem can be solved with the *IPOPT* method.

3.2 Obstacle Representation of Maps

By formulating the docking problem as in the previous section, the OCP-TP can avoid rectangular obstacles if they are given on the form (3.14). However, the task of creating such obstacles efficiently from a given map still remains. A suitable solution to this problem was developed as a part of this thesis, and the following section will explain how it was done. This section will also reveal implementation details that ensure efficient usage of the obstacles, both for creating the A*-initial guess and in the trajectory optimization process.

3.2.1 Polygon Map Representation

A common way of representing land objects in a map, is to use a number of polygons. The map used by the milliAmpere ferry is of this type. This map is stored as a JavaScript Object Notation (JSON)-file with polygons of different sizes making up the land masses. Figure 3.1 shows a section of this map plotted on a blue canvas. In the JSON-file, each polygon is stored as a list of points in the North-Eastern (NED) reference system, making up the total shape.

Figure 3.1: Polygon map representation of the harbour area of Trondheim, including the island of Munkholmen.

3.2.2 Polygons to Rectangle Obstacles: Concept

Knowing the polygon vertices still leaves the question of how to represent the whole polygon. One possibility would be to use a grid of various sized squares to ”fill” the polygons. This is done by Vestad [27], who suggests an automatic and practical route planning method for ships. He uses a recursive method of partitioning the allowed space of movement into squares of smaller and smaller resolution until a satisfying representation is made, as shown in Figure 3.2. Doing the same with the land polygons would be possible for the OCP-TP, but it would require a substantial amount of obstacles to achieve the desired resolution. Instead, a different approach is made here.

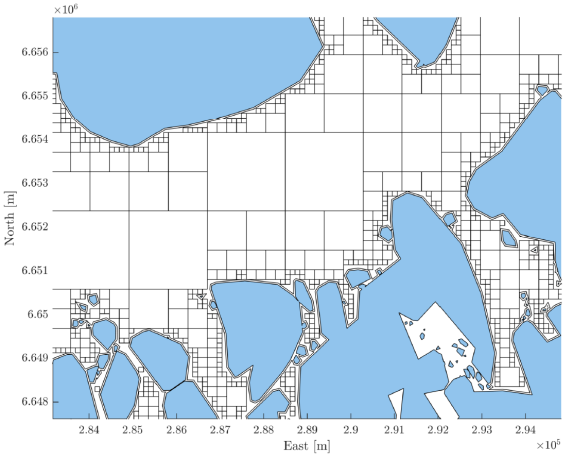


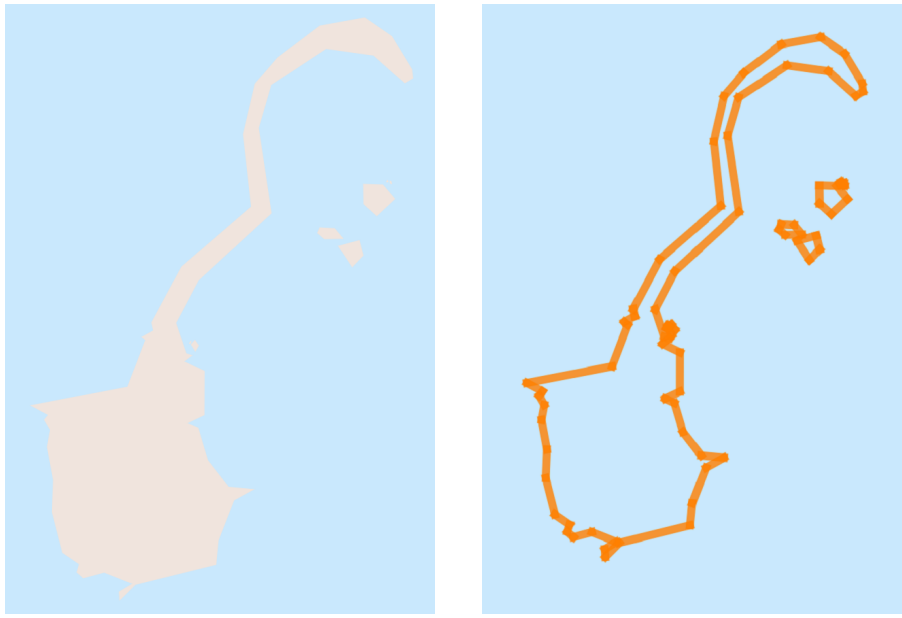
Figure 3.2: The allowed space of movement is partitioned into a number of squares of different sizes. ©2019 Vestad

The selected obstacle representation method is inspired by the works of Wu [28]. There, the idea of creating coastal constraints by using long, narrow ellipses is proposed. Instead of trying to represent the whole object, the ellipses simply ”fence” in the perimeter of the polygon. By replacing the ellipses with rectangles, this idea can be applied to the OCP-TP. This is illustrated in Figure 3.3, where the shape of the Munkholmen island is represented as 79 rectangular obstacles. If needed, the number of obstacles of can be reduced by using polygons with fewer vertices. Vestad [27] has developed such an algorithm, turning polygons into simpler versions with fewer edges, while still covering the original polygon.

3.2.3 Polygons to Rectangle Obstacles: Implementation

By representing the perimeter of the land polygons as a set of rectangular obstacles, the OCP-TP can generate trajectories that avoid the land masses. As each rectangular obstacle is determined by the parameters (x_c, y_c, a, b, α), an algorithm generating these parameters from the land polygons is developed. This algorithm works in the following way:

1. Take one of the remaining edges of the polygon



(a) **(b)**

Figure 3.3: Polygon representation **(a)** and rectangular obstacle representation **(b)** of the Munkholmen island.

2. Find the center point of the edge. This determines the parameters x_c and y_c.
3. Calculate the angle α of the edge (using atan2).
4. Determine the length of the obstacle a by using the euclidean distance between the two vertices that make up the edge.
5. Choose the desired width of the obstacle b.
6. Repeat until all edges have been represented.

By applying this algorithm to all the land polygons, the entire map can be represented by rectangular obstacles, as shown in Figure 3.4.

Although this way of representing obstacles is quite effective, it has some caveats. Most importantly, the obstacle width b must be selected with care, mainly for two reasons:

- The OCP-TP uses the center point of the ship when checking if the algebraic constraints are violated. The actual bow and aft of the vessel extends from this point in different directions, so the obstacle must be wide enough to prevent any part of the ship touching the actual land polygon.
- The rectangular obstacles serve as a ”fence” around the land polygon. As the trajectory is split into timesteps of length h, it is possible to plan a trajectory that ”jumps”

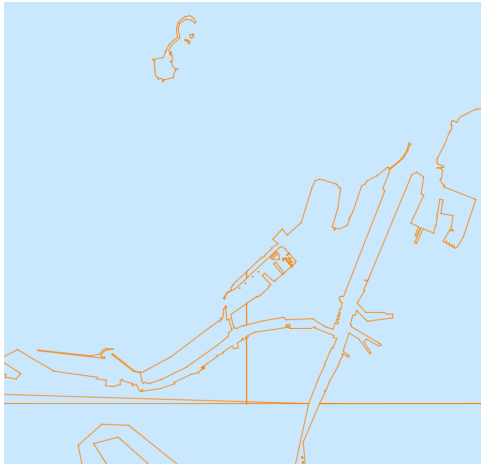


Figure 3.4: Rectangular obstacle representation of the Trondheim harbour area.

over the obstacle fence and gains access to the interior region of the polygon. This is highly unwanted, and can be prevented by choosing a sufficient large obstacle width. For instance, with the speed constraints set by the OCP-TP at 1 m/s and $h = 2$ s, the rectangles must be chosen at least 2 m wide to hold the trajectory outside of the polygon region.

With this in mind, $b = 3$ was chosen, corresponding to an obstacle width of 6 m. This keeps the 5 m long boat off the land and supports speed limits up to 3 m/s. The obstacle length a was also padded with 3 m to ensure that the same requirements are met in the corners. This extra padding keeps the vessel away from the obstacles, but unfortunately also makes it impossible to dock with a physical quay as the point of contact will be too close to the obstacle. This was solved by making the dock obstacle artificially smaller.

3.2.4 Remarks on Efficiency

With large and detailed polygon maps, the number of rectangular obstacle constraints created quickly reach the thousands. However, for the generation of a single trajectory, only around 50 of these are normally in the danger of being violated. To avoid the unnecessary obstacles slowing down the computation, only the closest obstacles are included when solving a trajectory planning problem. A sufficiently large rectangular sector is selected, covering the starting and ending point of the trajectory and only the rectangular obstacles within this sector are included in the problem. Fencing in this large sector with four more obstacles ensures that the optimization algorithm does not take any illegal shortcuts or roundtrips.

The A* algorithm that creates the initial guess can also take advantage of the created rectangular obstacle constraints. For this algorithm, a large portion of the computation time is spent checking whether a point is violating any obstacle. By only checking the points

in the rectangular obstacles making up the perimeter of the land polygons, the number of points to check is reduced substantially. Also, by inserting all illegal points into a set structure in advance, the cost of checking whether a point is within an obstacle is reduced to O(1) time.

3.3 A* Warm-starting Algorithm

As mentioned in the previous section, an A* warm-starting algorithm is implemented to create an initial guess for the trajectory planned by the OCP in Section 3.1. This is necessary as the OCP solver is only capable of converging to local minimums. When obstacles are introduced in the problem, the probability of getting stuck in one such local minimum increases, and finding a feasible path becomes quite computationally expensive, if possible at all. Therefore, an A*-algorithm has been implemented to find a feasible path between the start and the end point. An example of this method in usage is shown in Figure 3.5. The algorithm works according to the definition in Section 2.7 and uses the precomputed set of illegal points to avoid the restricted obstacle sections. After a possible path is found, it is postprocessed to make the path smoother. This processing includes waypoint reduction and path refinement using the algorithms proposed by Bhattacharya and Gavrilova [29]. An initial trajectory guess is then created, with suggested values $x_{ia}, y_{ia}, \psi_{ia}$ on north, east and heading for each timestep as shown in (3.28). The remaining states are set to zero.

$$\left[x_i, y_i, \psi_i, u_i, v_i, r_i, F_{1,xi}, F_{1,yi}, F_{2,xi}, F_{2,yi}\right] = \left[x_{ia}, y_{ia}, \psi_{ia}, 0, 0, 0, 0, 0, 0, 0\right] \quad (3.28)$$

The number of timesteps N is determined by using the length of the A* path multiplied with a scaling factor. For the simulations and sea trials performed in this thesis, this scaling factor is selected to give the docking operation an average speed of 0.5 m/s. With a timestep size of $h = 2$ s, this corresponds to a scaling factor of 1.

3.4 Trajectory Publisher

As shown in (3.16), the OCP splits the dynamics into N timesteps of length h. By choosing a limited amount of timesteps, the computational time can be reduced. However, this results in an optimal trajectory with insufficient resolution, as the optimal trajectory is tracked by the DP-controller in Section 3.6, which runs at a frequency of 10 Hz. Therefore, a trajectory publisher is created. By knowing the length of each timestep h in the optimal trajectory, the trajectory publisher is able to feed the DP-controller with reference values at a sufficient rate, using linear interpolation. The publisher also calculates the acceleration reference, needed in the feed-forward part of the controller.

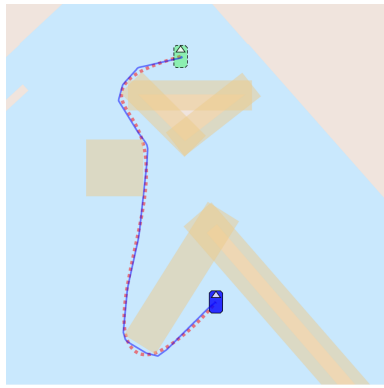


Figure 3.5: Warm-starting: The initial guess created by the A* warm-starting algorithm is shown in blue, while the resulting planned trajectory based on this guess is shown in dotted red. However, as this example illustrates, the shortest path is not necessarily the best. Simpler trajectories could be achieved by adding a cost of being too close to the obstacles both in the A* algorithm and the OCP.

3.5 Realization of the OCP-TP in the Vessel Computer System

With the algorithms and methods explained in the previous sections, the OCP-TP can be realized as an automatic docking method in the vessel system of the milliAmpere ferry. The actual implementation is done using ROS nodes as explained in Section 2.8, with topics and messages used for exchange of information. Figure 3.6 shows how the different components of the OCP-TP is organized, and the method works by the following steps:

1. The docking process starts by the operator selecting a desired end waypoint in the vessel GUI. If the vessel is in the OCP-TP docking mode, the process of planning an optimal docking trajectory is started.

2. Then, an initial guess of the trajectory, the end time t_f and the number of timesteps N is determined by the A* warm-starting algorithm. To create the initial guess, the algorithm uses the desired end waypoint and the initial pose of the vessel, along with the precomputed set of points that are in obstacles.

3. Next, the initial guess and the problem parameters are sent to the OCP module, along with the surrounding obstacles. The problem is constructed, then transcribed to an NLP using the Casadi-software and finally solved.

4. Furthermore, the optimal trajectory is found, containing pose, velocities and optimal control inputs for each timestep. This trajectory is sent to the trajectory publisher, which turns it into position, velocity and acceleration references.

5. Finally, the trajectory publisher sends the optimal references to the DP-controller at the desired rate.

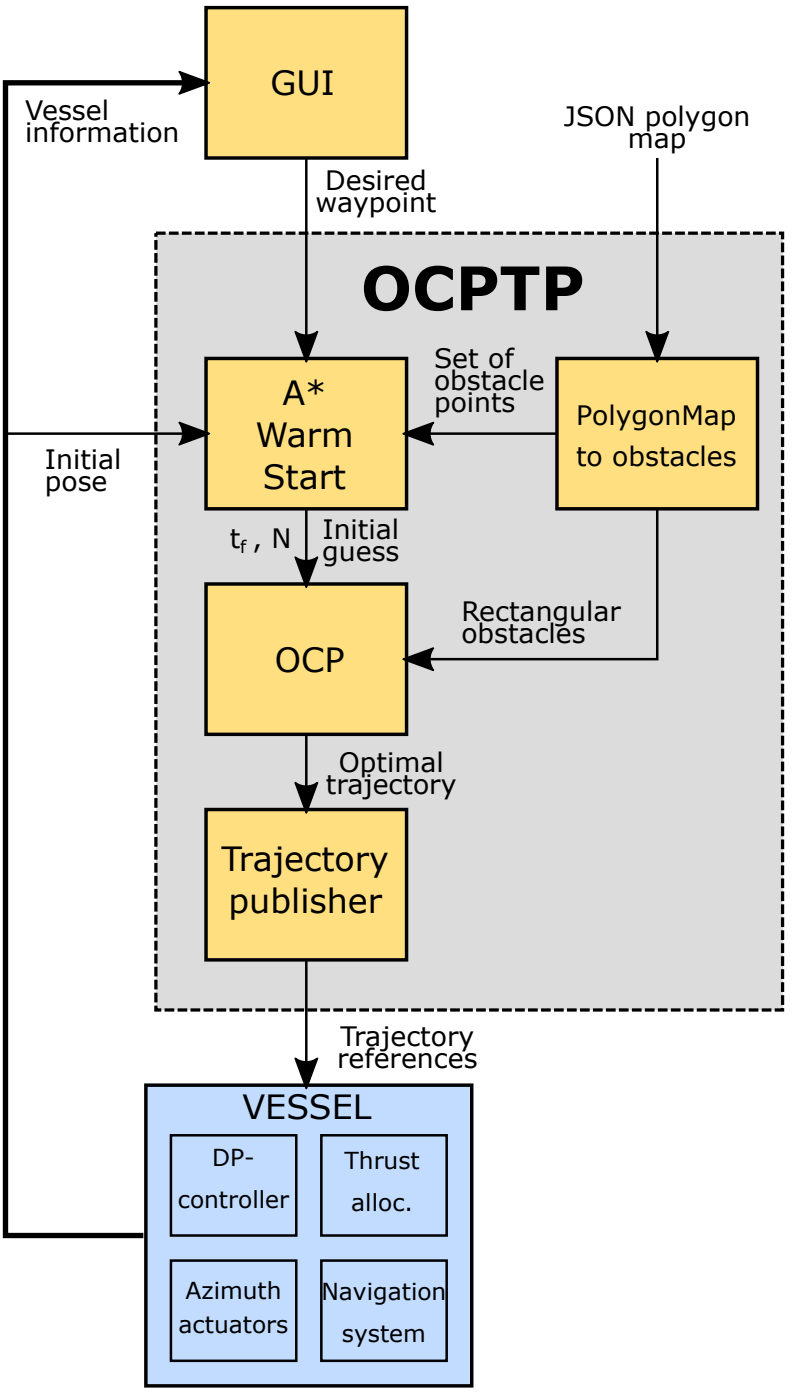


Figure 3.6: Flowchart showing how the OCP-TP is organized as a part of the vessel system.

3.6 Dynamic Positioning Controller

Although it was not developed as a part of this thesis, understanding the DP-controller that tracks the optimal trajectory is important. As the controller was already implemented on milliAmpere and had satisfying performance, developing a new one was deemed unnecessary. The DP-controller has been explained by Bitar et al. [30] in the following way:

The controller command $\boldsymbol{\tau}_c(t)$ is determined using Proportional-Integral-Derivative (PID) feedback, along with feed-forward terms from the velocity and acceleration references:

$$\boldsymbol{\tau}_c(t) = \boldsymbol{\tau}_{fb}(t) + \boldsymbol{\tau}_{ff}(t) \tag{3.29}$$

The feed-forward term is

$$\boldsymbol{\tau}_{ff}(t) = \boldsymbol{M}\dot{\boldsymbol{\nu}}_d(t) + \boldsymbol{D}(\boldsymbol{\nu}_d(t))\boldsymbol{\nu}_d(t) \tag{3.30}$$

where $\boldsymbol{\nu}_d$ and $\dot{\boldsymbol{\nu}}_d$ is the reference velocity and acceleration, while the system inertia matrix $\boldsymbol{M}$ and system damping matrix $\boldsymbol{D}(\boldsymbol{\nu}_d(t))$ follows from section 2.1.1. The PID feedback is

$$\boldsymbol{\tau}_{fb}(t) = -\boldsymbol{R}(\psi(t))^\top \left(\boldsymbol{K}_p \tilde{\boldsymbol{\eta}}(t) + \int_0^t \boldsymbol{K}_i \tilde{\boldsymbol{\eta}}(\tau) d\tau + \boldsymbol{K}_d \dot{\tilde{\boldsymbol{\eta}}}(t) \right) \tag{3.31}$$

with $\tilde{\boldsymbol{\eta}}(t) = \boldsymbol{\eta}(t) - \boldsymbol{\eta}_d(t)$ where $\boldsymbol{\eta}_d$ is the reference pose. The controller gains are $\boldsymbol{K}_p = \text{diag}\{100, 100, 200\}$, $\boldsymbol{K}_i = \text{diag}\{10, 10, 20\}$ and $\boldsymbol{K}_d = \text{diag}\{1000, 1000, 1500\}$. The integrator term in (3.30) has an anti-windup condition, limiting its contribution to $\pm[150\,\text{N}, 150\,\text{N}, 200\,\text{Nm}]^\top$. The desired control command $\boldsymbol{\tau}_c(t)$ is then sent to the thrust allocation system explained in Section 2.4.2, which translates the force commands into desired azimuth angles and propeller speeds.

Chapter 4

Development of the Graphical User Interface

Ultimately, the milliAmpere ferry is intended to be fully autonomous and operate nearly without any human intervention. However, before such a technology can be deployed, it must be thoroughly tested. The process of developing, testing, monitoring and debugging in this thesis, called for a Graphical User Interface (GUI). A user interface will connect the human developer to the behaviour, intentions and controls of the ferry. In this way, the developer can assess whether the new technology is working according to plan.

It is also important to distinguish between the interface used during the development phase of the ship and the user interface that will be presented to an operator when the vessel is put in regular, passenger traffic. The latter interface must be in compliance with international standards and requirements, meeting demands on e.g. colour usage, resolution, display scale and reference frames [31]. However, the interface developed in this thesis will not be used in such environments and therefore does not have to follow these regulations. Instead, a variety of colours, objects and maps are used to give the developer the necessary vessel information, and to highlight whether the vessel behaves as planned.

For the milliAmpere ferry, a preliminary interface has already been developed as illustrated in Figure 4.1. This interface has been developed using Hypertext Markup Language (HTML) and JavaScript (JS), with a bridge node in the ROS system that feeds the interface with vessel data. The interface includes:

1. A map showing the ferry's position and the surrounding land
2. Mode selection
3. Waypoint selection and storage
4. Velocity graphs

5. Thruster outputs and azimuth visualizations
6. Global Navigation Satellite System (GNSS) status

In this thesis, the preliminary interface has been augmented with:

7. Graphical reference monitor
8. New layout and trajectory visualization in the map
9. Power usage plot

This new interface is shown in Figure 4.2 and the additions will be further explained in the following sections.

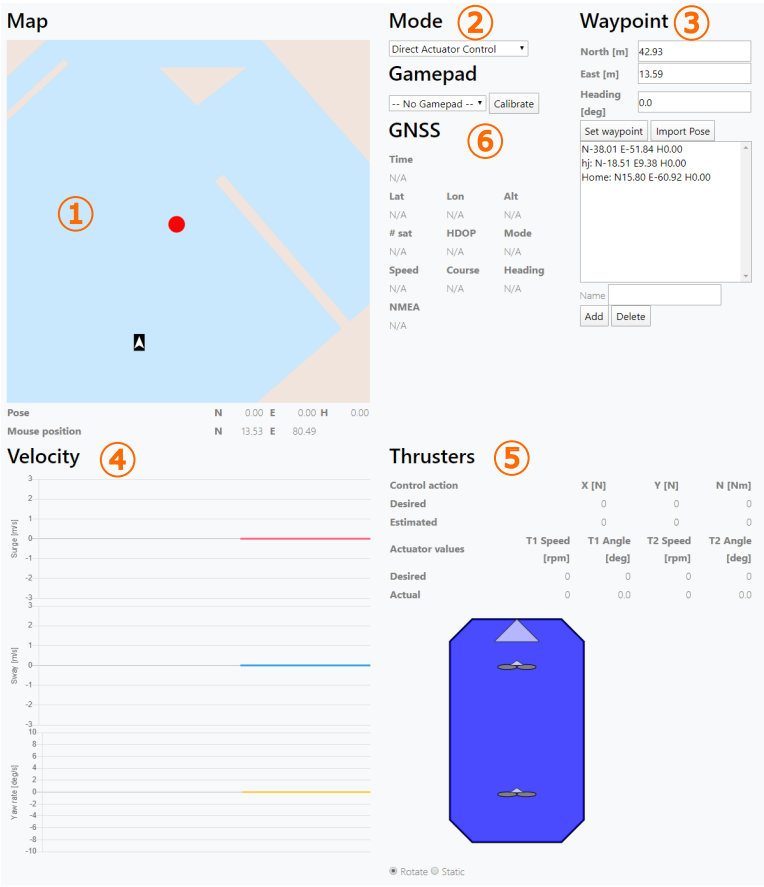


Figure 4.1: Illustration of the preliminary GUI. The various components are marked with numbers according to the list.

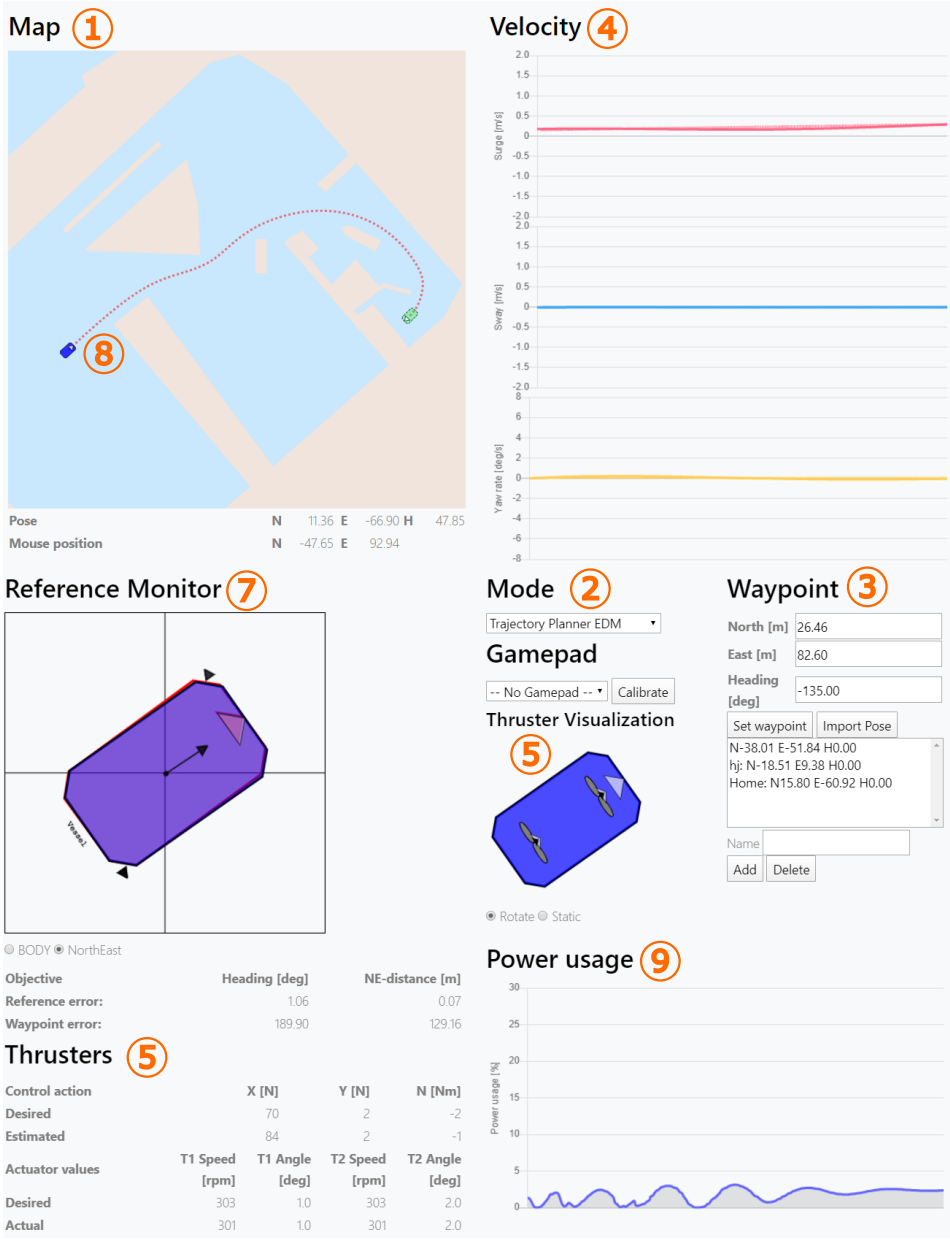
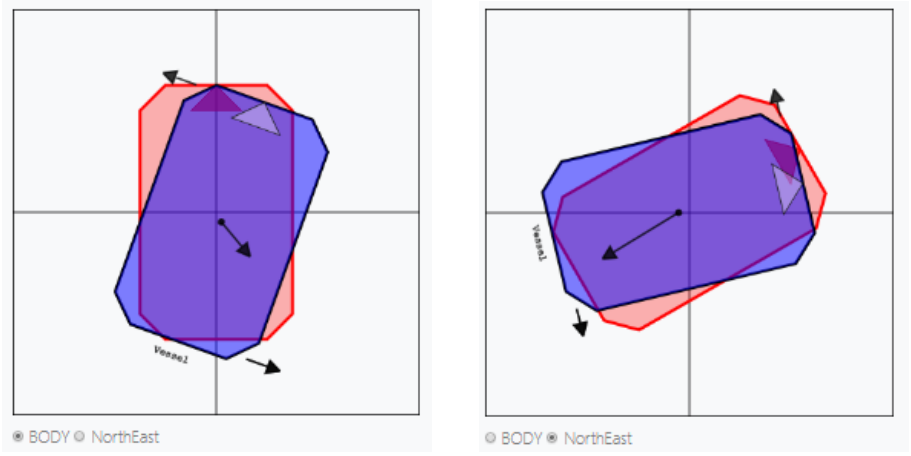


Figure 4.2: The new interface developed in this thesis. The various components have been rearranged and are numbered according to the list. Trajectory visualization, reference monitoring and a power usage plot has been added to the interface.

4.1 Reference Monitor

To get the ferry from a starting position to the desired waypoint, a reference trajectory is created. The ferry moves by feeding the reference positions, velocities and accelerations to the control system where they are transformed to thruster inputs. Because of this, the vessel must keep up with the reference to operate as intended. Motivated by this desire to monitor the deviations from the reference, a graphical reference monitoring system was added to the user interface. In addition to being useful for an operator when validating the intention and status of the vessel during operation, this monitoring system has proven quite valuable in the process of controller tuning and testing of the trajectory planner.

An illustration of the reference monitoring system is given in Figure 4.3. The monitoring system is reference-centred, having the red shape of the reference in the middle of the plot, while the actual pose of the ferry is given in blue. The octagonal shape is chosen to resemble the actual shape of the ferry along with a triangle indicating the forward direction. Having this setup, the amount of deviation from the reference can be seen in both surge, sway and yaw simultaneously. The deviation is also given in number values below the plot, to give actual numerical meaning to the relative differences. The actual control force and moments being made by the vessel are given in the plot as black arrows. They indicate the direction of the forces and increase in length according to how the forces change in magnitude. The inclusion of these forces give the operator further knowledge about the vessel behaviour, and provides a notion on how the vessel is trying to correct for the reference deviation.



(a) Reference monitor in body-mode.

(b) Reference monitor in north east coordinates.

Figure 4.3: Reference monitor displays: The red shape indicates the desired pose, while the blue shape is the actual pose.

The reference monitor comes with the option of choosing either body-coordinates or NED-coordinates as Figure 4.3a and Figure 4.3b show. The body-mode has the reference heading fixed at zero and is especially helpful when doing trajectory monitoring with operators being on the actual ferry. The NED-mode rotates all poses according to north and is simpler to understand when comparing the reference monitor with a map.

The desired waypoint is included in the reference plot as a similar shape, but in green. As Figure 4.4 exemplifies, this gives the operator a notion of how far the vessel is from the desired end position. In combination with the red shape of the reference, it is easy to see whether the vessel is converging to the waypoint in the correct manner.

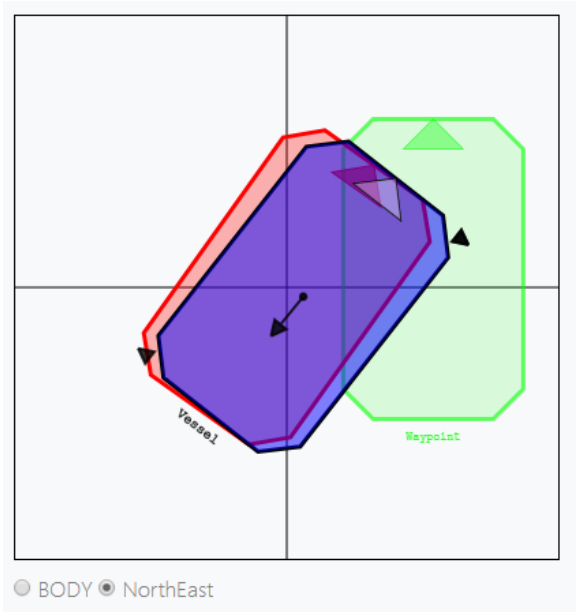


Figure 4.4: Reference monitor with waypoint indication.

By using the mouse wheel, the operator can also scale the reference monitor plot to the desired size as illustrated in Figure 4.5. This gives the operator the ability to see where the waypoint is located and to settle on a desired level of detail.

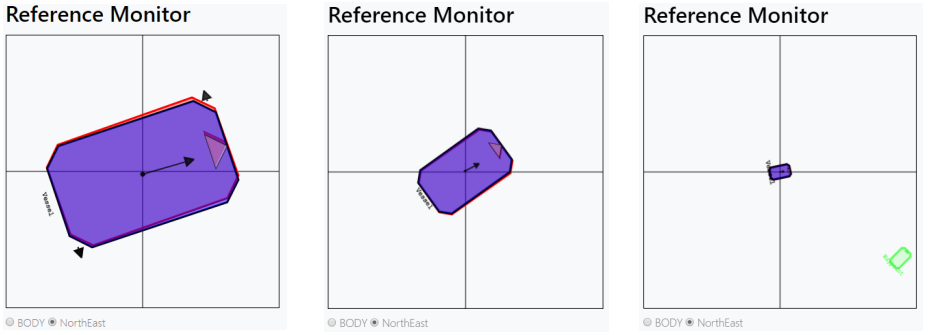


Figure 4.5: Reference monitor in different scale levels.

4.2 Trajectory Map Visualization

A visualization of the planned trajectory in a map is useful for the operator, if he is to trust the ferry during automatic trajectory tracking. In this way, the planned trajectory can be verified in terms of feasibility, length, time duration and obstacle avoidance. Additionally, it should be possible to interact with the map, for instance by selecting waypoints.

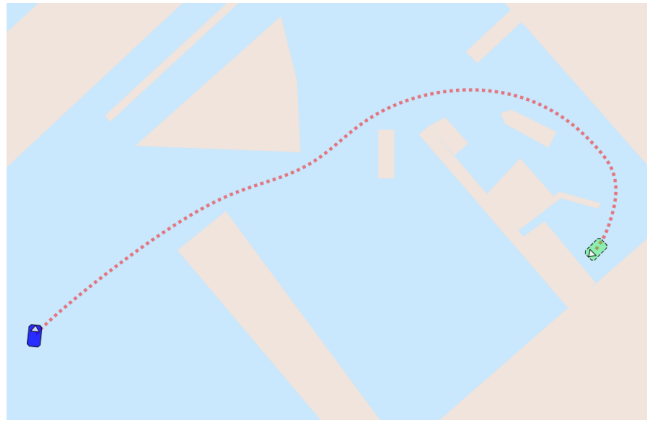
In this thesis, such a map has been developed. It builds on the existing map from the preliminary interface and has been augmented with:

- New ship and waypoint shape and color
- Waypoint heading selection by right-clicking with the mouse
- Trajectory visualizations and real-time update

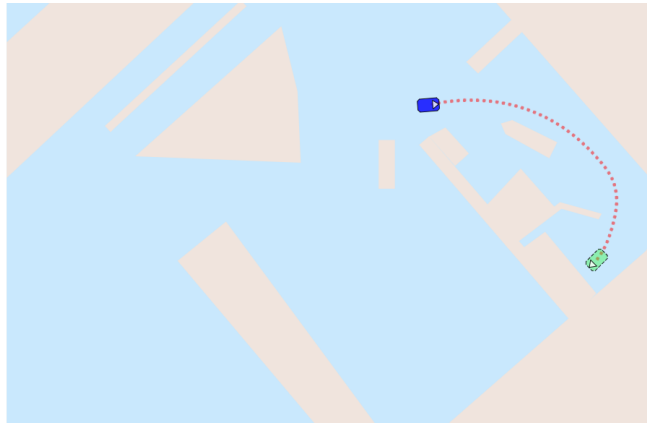
As illustrated in Figure 4.6, this map uses the blue octagonal shape to indicate the current location of the milliAmpere vessel. By clicking on the map, the green shape representing the selected waypoint emerges, and its heading can be altered by right-clicking with the mouse.

When a waypoint is selected and the trajectory planner starts the calculations, the first path to be visualized is the A*-guess as illustrated in Figure 4.6a. As some complex paths can be computationally expensive, plotting the A*-guess early helps to assure the operator that the algorithm is on the right track. After the algorithm has converged to an optimal solution, the A*-guess is replaced with the actual calculated path as shown in Figure 4.6b. Then, as the vessel starts following the trajectory, the map visualization is continuously updated and only shows the remaining part to travel as seen in Figure 4.6c.

(a) Initial A* guess.



(b) Initial trajectory.



(c) Updated remaining trajectory.

Figure 4.6: Different stages of the map trajectory visualization.

4.3 Power Usage Plot

Many of the docking methods seek to develop energy efficient-trajectories. The validation of this efficiency is done by post-processing the data, but it is still interesting to see how much power is being consumed while operating. Therefore, a power usage plot has been developed, as illustrated in Figure 4.7. This plot shows the current power usage in percent of maximum usage and how it has changed over the last 90 seconds. As the motor current and voltage are not present in the simulator, the propeller revolution speed is the size used to generate this power usage estimate employing the assumptions explained in Section 3.1.2. The plot will give a quick overview of the energy-efficiency of the method, and it also helps to highlight periodic control inputs or sudden large spikes.

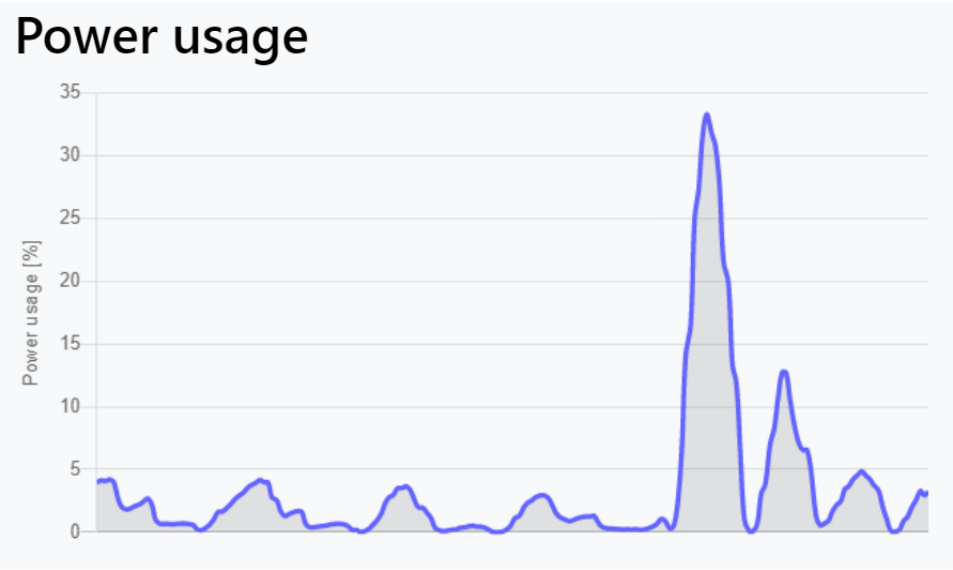


Figure 4.7: Power usage plot as seen in the interface.

Chapter 5

Simulation and Experimental Results

To assess the performance of the OCP-TP, it was tested both in simulation and in sea experiments. In this chapter, the results from these simulations and experiments are presented. However, firstly, methods for comparison, performance metrics and test scenarios will be defined.

5.1 Trajectory Planning Methods For Comparison

To asses the quality of the trajectory planner, it is compared with other trajectory generating methods. In the simulations and experiments that follow, it will therefore be compared with the following methods:

- Trajectory Reference Filter (TRF)
- Model Predictive Control (MPC)
- A* Path Planner (A-PP)
- Manual

Here, TRF, MPC and A-PP are automatic methods, and they all work by creating trajectory references that are tracked by the same controller as for the OCP-TP. This is the PID feed-forward DP-controller presented in Section 3.6. A more detailed explanation of the comparison methods follows below.

5.1.1 TRF

This is an existing method on the milliAmpere ferry that can be used for automatic docking, developed by Tobias Torben. The method accepts a requested waypoint and generates

a feasible trajectory using a 3rd order reference filter in north, east and yaw. This ensures that smooth reference positions and velocities are given between the starting position and the desired final waypoint. The implementation does not support obstacle avoidance.

5.1.2 MPC

This method is also an existing docking method for the milliAmpere ferry suggested by Bitar et al. [30]. The process of docking is solved as an optimal control problem with recalculation of the trajectory at 0.1 Hz making it an MPC controller. The optimization problem is solved in Casadi, using the same vessel model as the trajectory planner from Chapter 3. However, the cost functional is on the form

$$\min_{\boldsymbol{x}_d(\cdot),\boldsymbol{u}_d(\cdot),\boldsymbol{s}(\cdot)} \int_{t_0}^{t_o+T} \Big(F(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)) + \boldsymbol{k}_{\boldsymbol{s}}^\top \boldsymbol{s}(t) \Big) dt \tag{5.1}$$

Here, $F(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t))$ is a cost-to-go-function, while $\boldsymbol{k}_{\boldsymbol{s}}^\top \boldsymbol{s}(t)$ is a cost-to-go on the slack variables. The planned states are denoted $\boldsymbol{x}_d = [\boldsymbol{\eta}_d^\top, \boldsymbol{\nu}_d^\top]^\top$, consisting of the Earth-fixed pose $\boldsymbol{\eta}_d = [x_d, y_d, \psi_d]^\top$ and the body-fixed velocity vector $\boldsymbol{\nu}_d = [u_d, v_d, r_d]^\top$. With the input vector $\boldsymbol{u}_d = [f_{x1}, f_{y1}, f_{x2}, f_{y2},]^\top$ and the final docking pose defined as $\boldsymbol{\eta}_f = [x_f, y_f, \psi_f]^\top$ the cost-to-function is

$$\begin{aligned} F(\boldsymbol{x}_d(t),\boldsymbol{u}_d(t)) = \\ H\Big(\begin{bmatrix} x_d(t) - x_f \\ y_d(t) - y_f \end{bmatrix} \Big) + \\ 20(1 - \cos(\psi_d(t) - \psi_f)) + \\ 10 v_d(t)^2 + 10 r_d(t)^2 + \\ \boldsymbol{u}_d(t)^\top \boldsymbol{u}_d(t) / m_{11}^2 \end{aligned} \tag{5.2}$$

where the pseudo-Huber function

$$H(\boldsymbol{a}) = \delta^2 \Bigg(\sqrt{1 + \frac{\boldsymbol{a}^\top \boldsymbol{a}}{\delta^2}} - 1 \Bigg) \tag{5.3}$$

with $\delta = 10$ m uses the positioning error to provide a penalty that is quadratic near zero and turns linear when the error gets large.

As (5.2) indicates, this MPC controller seeks to minimize the time to reach the final position while using as little control input as possible. This gives a fast and energy-efficient docking alternative, with robustness to disturbances provided by the MPC loop.

The MPC method does also support obstacle avoidance. As the vessel is moving, it considers the surrounding land constraints and creates a feasible, convex, allowed space of movement as illustrated in Figure 5.1. By doing so, any waypoint selected inside this region can be reached without violating the obstacle constraints. However, if selected, waypoints outside of this region are not guaranteed to be reached. Thus, the method always avoids crashes, but it does not necessarily lead the vessel to the desired waypoint if the harbor environment is too complex.

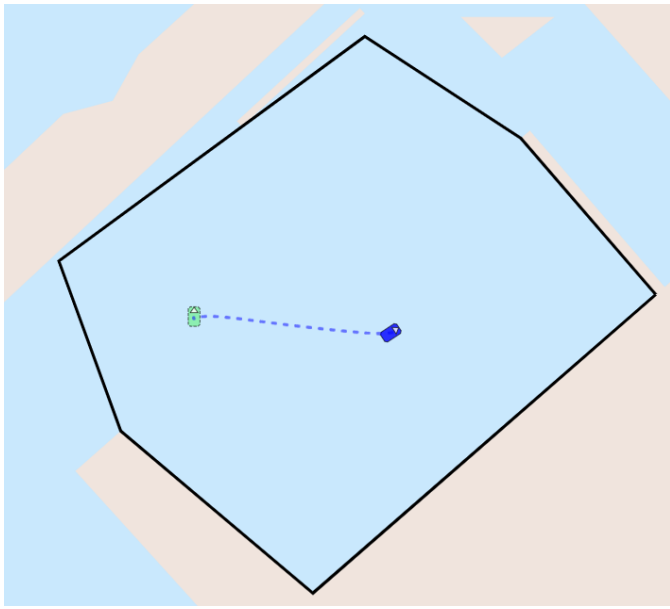


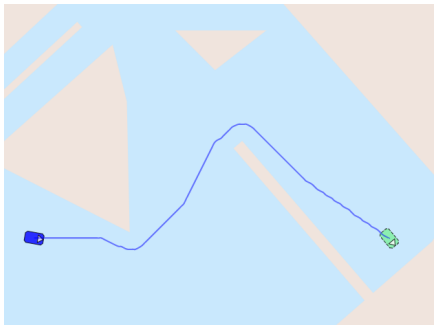
Figure 5.1: Convex feasible region of the Docking MPC method.

5.1.3 A-PP

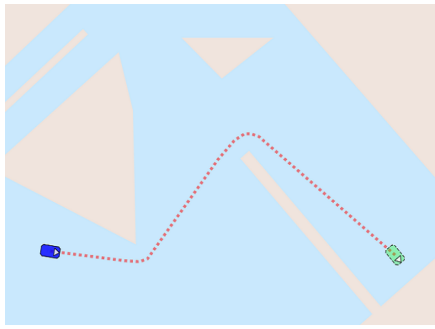
Comparing the OCP-TP with a method that also supports obstacle avoidance is of great interest. As neither of the two previous methods can guarantee a feasible path to any waypoint, such a method has been developed as a part of this thesis.

The method has been named the A* Path Planner as it is based on the initial path given by the A* algorithm from Section 3.3. In Figure 5.2a, an example of one such initial path is given. At the given stage, this path is unsuitable for tracking as it is only parameterized in NED-coordinates and obviously lacks smoothness. To introduce this smoothness and create an actual shortest path, the number of points is reduced by using path reduction algorithms from Bhattacharya and Gavrilova [29]. Additional waypoints are also introduced in an attempt to shorten the path, resulting in a path as illustrated in Figure 5.2b.

Finally, a trajectory must be created from the path. This is done by assigning a total travel time to the tracking operation, and then splitting the path into a corresponding number of reference positions. To allow sufficient time for acceleration and deceleration, these positions are shifted slightly in the start and the end of the trajectory. Desired headings for the boat are also introduced, and the path-tangential angle of the path is used for each trajectory position. To avoid jumps in the heading reference, a maximum heading rate of change is enforced. This ensures that a smooth and feasible heading reference is available, from the start heading all the way to the end. This trajectory can now be sent to the controller and followed by the vessel.



(a) Initial path generated by the A*-algorithm.



(b) Final trajectory from the A-PP

Figure 5.2: Illustration of the different steps of the A-PP. Green vessel silhouette indicates the selected waypoint.

5.1.4 Manual

The milliAmpere vessel allows for operator control through an RC-controller. In this mode, the operator decides the sway and surge force and yaw moment by using the joysticks. Comparing the OCP-TP with the vessel movements performed by an operator will indicate if the automated process outperforms a skilled captain. A human operator will also be able to avoid land constraints, allowing to compare difficult docking scenarios.

5.2 Performance Metrics

To asses and compare the different docking methods, a set of performance metrics were defined:

5.2.1 Power Consumption

With the goal of creating an energy-efficient docking method, it is important to have a way of measuring how much power the vessel is using at any time instance. For the physical experiments conducted on the milliAmpere ferry, this data can be obtained from the motor state of the two azimuth thrusters. Here, the voltage V_i and current I_i demands of the two propellers $i \in \{1, 2\}$ are logged. In order to use these values, a signal filtering algorithm had to be made. This process is described in Appendix A. Using the standard power equation from electrical engineering

$$P = VI \tag{5.4}$$

the momentary power used for propulsion by the vessel can be expressed as

$$P_{elec}(t) = V_1(t)I_1(t) + I_2(t)V_2(t) \tag{5.5}$$

giving the total energy consumption to be equal to

$$E_{elec}(t) = \int_0^t P_{elec}(\tau)d\tau \tag{5.6}$$

For the vessel simulations, the current and voltage measurements are not available. Thus, the exact power cannot be known, but by using the propeller shaft speed a proportional size can be computed. As derived in (3.9), the equation for power consumption P_{mech} can be found proportional to

$$P_{mech} \propto |n_s^3| \tag{5.7}$$

Thus, by defining

$$P_{mech}(t) = |n_s(t)^3| \tag{5.8}$$

the size E_{mech} proportional to the total energy consumption can be defined as

$$E_{mech}(t) = \int_0^t P_{mech}(\tau) d\tau \tag{5.9}$$

Ideally, the power used by the servos to change the azimuth thruster angles should be incorporated in the power consumption metric. However, measurements of this is not available in the simulation or in the actual vessel. Not including these measurements in the metric can still be accepted, as the amount of power consumed by the angle servos can be assumed negligible compared to the amount consumed by the propeller motors.

5.2.2 Trajectory Tracking Capability

The Integral of the Absolute Error (IAE) as suggested by Sørensen and Breivik [32] is used to evaluate how well the vessel manages to follow the computed trajectory. The position and heading errors are normalized between 0 and 1, using the parameters given in Table 5.1.

Name	Value
$p_{e,max}$	3 m
$\psi_{e,max}$	30 deg

Table 5.1: Normalizing parameters for the IAE metric. These values were selected by looking at the collected data and finding the maximum values.

With the combined positional error in north and east defined as the Euclidean distance

$$p_e(t) = \sqrt{(x(t) - x_d(t))^2 + (y(t) - y_d(t))^2} \tag{5.10}$$

the normalized errors are selected as

$$\bar{p_e}(t) = \frac{p_e(t)}{p_{e,max}}, \qquad \bar{\psi_e}(t) = \frac{|\psi(t) - \psi_d(t)|}{\psi_{e,max}} \tag{5.11}$$

Thus, by defining the error $\bar{e}(t)$ as

$$\bar{e}(t) = \bar{p_e}(t) + \bar{\psi_e}(t) \tag{5.12}$$

the IAE can be stated as

$$IAE(t) = \int_0^t |\bar{e}(\tau)| d\tau \tag{5.13}$$

It is also interesting to see which docking method has the best tracking performance versus energy consumption. Using the power consumption P_{elec} as defined in section 5.2.1, the Integral of the Absolute Error multiplied by the energy consumption (IAEW) is stated as

$$IAEW(t) = \int_0^t |\bar{e}(\tau)| d\tau \int_0^t P_{elec}(\tau) d\tau \tag{5.14}$$

5.2.3 Obstacle Avoidance

Being able to steer clear of the harbour obstacles is essential for an automatic docking method. To evaluate how well the method is functioning, the Distance to the Nearest Obstacle (DNO) can be computed at any point in the trajectory by using a list of obstacle points. Finding the distance to the nearest obstacle corresponds to finding

$$\begin{aligned} DNO(t) = \min \Bigg(& \sqrt{(x(t) - x_{o,1})^2 + (y(t) - y_{o,1})^2}, ..., \\ & \sqrt{(x(t) - x_{o,N})^2 + (y(t) - y_{o,N})^2} \Bigg) \end{aligned} \tag{5.15}$$

where the total list of obstacle points is given as

$$\boldsymbol{L_o} = [x_{o,1}, y_{o,1}, ..., x_{o,N}, y_{o,N}] \tag{5.16}$$

The points in $\boldsymbol{L_o}$ are generated by adding the xy-values of the points making up the perimeter of every rectangular obstacle in the problem. The points have 0.1 m of spacing between each other.

5.2.4 Passenger Comfort

As the ferry is intended to carry passengers, a body acceleration metric is defined to quantify passenger comfort. With the normalizing parameters in Table 5.2 being limits for what is considered as comfortable accelerations for passengers, the passenger comfort metric $C(t)$ is defined as:

$$C(t) = \int_0^t \left(\frac{|\dot{u}(\tau)|}{\dot{u}_{lim}} + \frac{|\dot{v}(\tau)|}{\dot{v}_{lim}} + \frac{|\dot{r}(\tau)|}{\dot{r}_{lim}} \right) d\tau \tag{5.17}$$

5.2.5 Actuator Wear and Tear

With two azimuth thrusters available, it is interesting to see how much stress they are exposed to. To quantify the amount of wear and tear, a combined metric $AWT(t)$ using

Name	Value
$\dot{u}_{lim}$	1.1 m/s^2
$\dot{v}_{lim}$	1.1 m/s^2
$\dot{r}_{lim}$	0.2 rad/s^2

Table 5.2: Normalizing parameters for the body acceleration metric. Numerical values for these parameters are given in [33].

the change in propeller shaft revolution rate $\dot{n}_{s,i}$ and azimuth angle turning rate $\dot{\alpha}_i$ for the two azimuths $i \in \{1, 2\}$ is defined as

$$AWT(t) = \int_0^t \left(\frac{|\dot{n}_{s,1}(\tau)|}{\dot{n}_{s,max}} + \frac{|\dot{n}_{s,2}(\tau)|}{\dot{n}_{s,max}} + \frac{|\dot{\alpha}_1(\tau)|}{\dot{\alpha}_{max}} + \frac{|\dot{\alpha}_2(\tau)|}{\dot{\alpha}_{max}} \right) d\tau \tag{5.18}$$

The normalizing parameters represents the maximum change in propeller shaft rate and azimuth angle as defined in Table 5.3.

Name	Value
$\dot{n}_{s,max}$	650 rpm/s
$\dot{\alpha}_{max}$	55 deg/s

Table 5.3: Normalizing parameters for the IAE metric. Numerical values for the parameters are selected by looking at the data.

5.3 Scenarios

In order to validate the behaviour, efficiency and limitations of the trajectory planner from Chapter 3, a set of testing scenarios have been determined. These are selected to test various aspects of the docking procedure and will focus on:

- Energy efficiency
- Trajectory feasibility
- Obstacle avoidance
- Actuator wear and tear

The alternative docking methods presented in the previous section were developed for different purposes, and all of them cannot be used in every scenario. The following subsections will explain each scenario in detail, how they are tested and the docking methods that will be used.

5.3.1 Scenario 1

The first scenario is defined according to Table 5.4 and aims at moving the vessel 31.85 meters north and 4.66 meters east while rotating 93.5 degrees as illustrated in Figure 5.3.

This is a simple scenario that can be performed by all docking methods as indicated by Table 5.5, but still demanding enough to create differences in trajectories, power usages and tracking capabilities.

Scenario	$\boldsymbol{x}_i$	$\boldsymbol{x}_f$	Max time [s
1	$[48.15, 45.34, 48.50°, 0, 0, 0]$	$[80, 50, -45°, 0, 0, 0]$	66

Table 5.4: Parameters for docking scenario 1.

Method / **Environment**	*OCP-TP*	*TRF*	*MPC*	*A-PP*	*Manual*
Simulation	YES	YES	YES	YES	NO
Sea experiment	YES	YES	YES	YES	YES

Table 5.5: Docking methods used in Scenario 1.

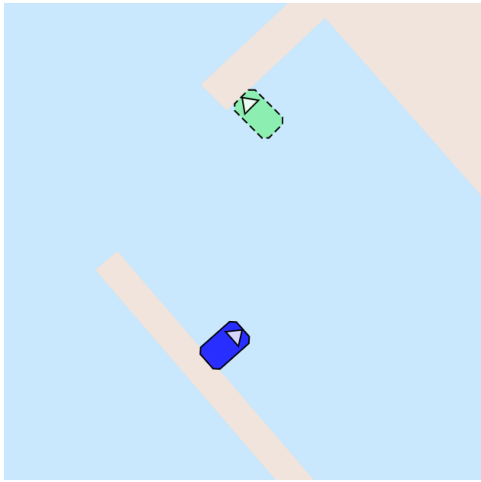


Figure 5.3: Scenario 1: Initial and final pose in blue and green, respectively.

5.3.2 Scenario 2

In scenario 2, the goal is to move the vessel 67 meters north and 27 meters west while rotating clockwise 90 degrees as Table 5.6 indicates. In addition, the harbour environment includes several obstacles as illustrated in Figure 5.4. As Table 5.7 shows, this means that not all docking methods are applicable. This scenario aims at showing the obstacle avoidance capabilities of the docking methods, and how their chosen strategies affect power consumption and other evaluation metrics.

Scenario	$\boldsymbol{x}_i$	$\boldsymbol{x}_f$	Max time [s]
2	$[27, 75, 45°, 0, 0, 0]$	$[94, 48, 135°, 0, 0, 0]$	226

Table 5.6: Parameters for docking scenario 2.

Method / **Environment**	*OCP-TP*	*TRF*	*MPC*	*A-PP*	*Manual*
Simulation	YES	NO	NO	YES	NO
Sea experiment	YES	NO	NO	YES	YES

Table 5.7: Docking methods used in Scenario 2.

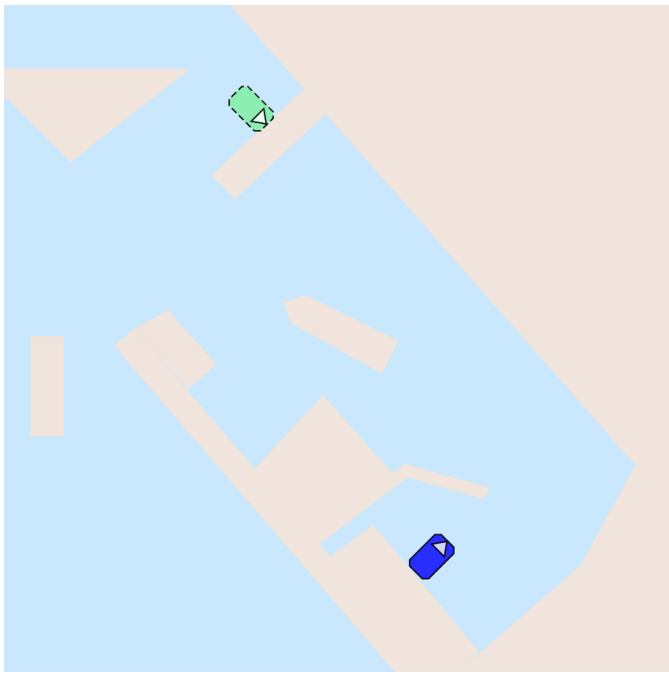


Figure 5.4: Scenario 2: Initial and final pose is separated by obstacles.

5.3.3 Scenario 3

Scenario 3 is designed to resemble an actual docking operation. As Table 5.8 shows, the vessel approaches the starting point for the docking operation at 1 m/s in surge direction. A suitable docking trajectory must then be calculated from this starting state to the final docking position. Table 5.8 shows the docking methods that are applicable for this scenario, while Figure 5.5 gives a visual representation of the scenario.

Scenario	$\boldsymbol{x}_i$	$\boldsymbol{x}_f$	Max time [s]
3	$[20, -75, 45°, 1, 0, 0]$	$[48.15, 45.34, 48.5°, 0, 0, 0]$	306

Table 5.8: Parameters for docking scenario 3.

Method / Environment	*OCP-TP*	*TRF*	*MPC*	*A-PP*	*Manual*
Simulation	YES	NO	NO	YES	NO
Sea experiment	YES	NO	NO	YES	YES

Table 5.9: Docking methods used in Scenario 3.

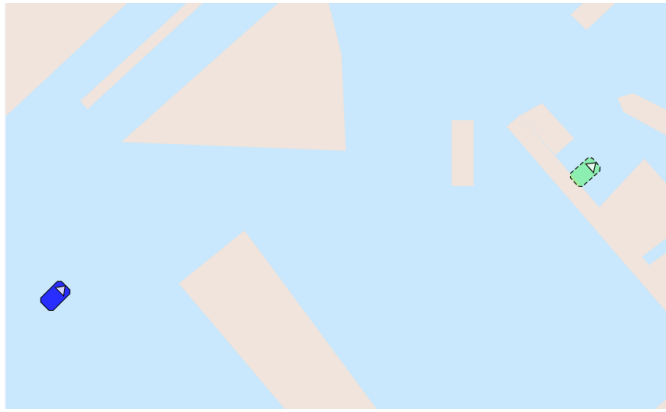


Figure 5.5: Scenario 3: The vessel starts following the docking trajectory from the blue starting position.

5.4 Simulation Results and Discussion

Before running the scenarios in sea trials, they were tested in the milliAmpere simulator. The metrics

- Estimated total energy E_{mech}
- Distance to the Nearest Obstacle (DNO)
- Integral of the Absolute Error (IAE)

were used to evaluate the simulation results. Later, when presenting the experimental results, the remaining metrics will be included as well. The following section presents the results from the simulations:

5.4.1 Scenario 1

This section contains the simulation results from Scenario 1. The scenario was performed as defined in Table 5.4 with the following docking methods:

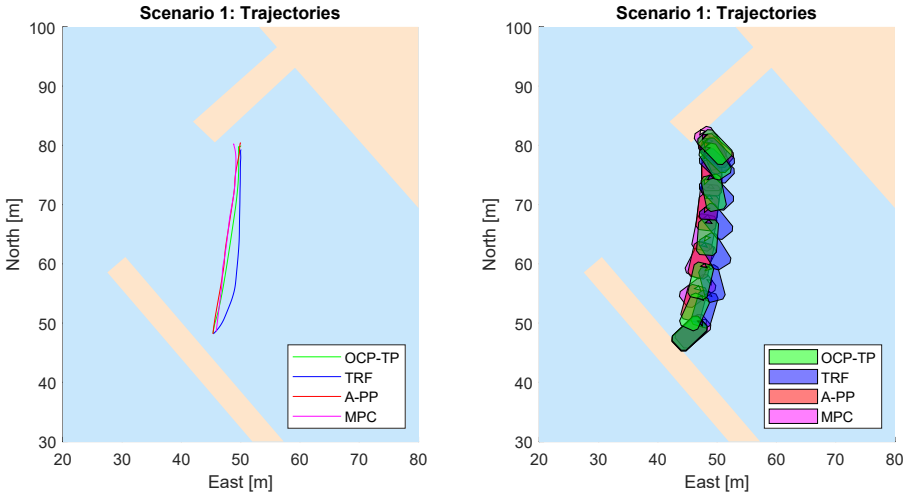
- OCP-TP
- TRF
- A-PP
- MPC

The end values of the selected metrics are found in Table 5.10 and an explanation of the results is given below.

Method	E_{mech}	IAE
OCP-TP	**0.56**	9.5
TRF	0.68	**6.6**
A-PP	1.16	13.7
MPC	1.07	11.4

Table 5.10: Scenario 1 Simulation: End values of metrics with the best method in green and bold.

As Figure 5.6 and Figure 5.7 show, the docking methods were, in simulation, able to place the vessel at the desired waypoint. The TRF changes the heading angle earlier than the other methods, but aside from this, the methods follow approximately the same trajectory.



(a) Scenario 1: Followed path.

(b) Scenario 1: Path poses at 10 s intervals.

Figure 5.6: Scenario 1 Simulation: Trajectories of the different docking methods.

The estimated power consumption from the methods during the simulation of Scenario 1 is found in Figure 5.8. These results indicate that the OCP-TP method has the lowest energy

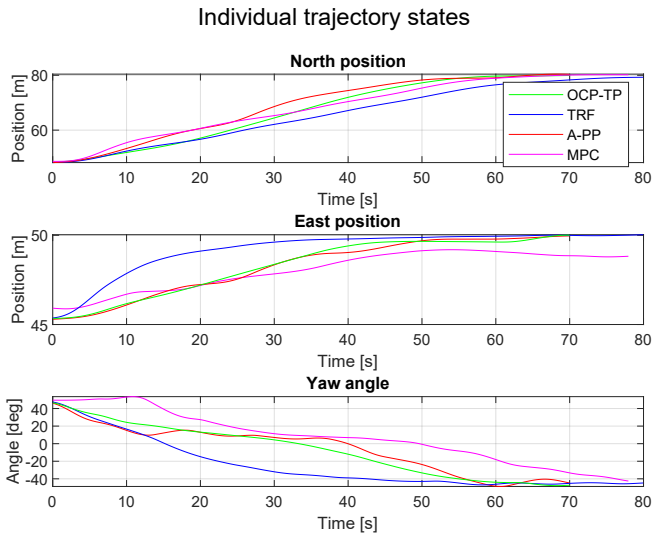


Figure 5.7: Scenario 1 Simulation: Pose trajectories.

consumption, closely followed by the TRF method. In addition, it is seen from Figure 5.9 that the methods are able to avoid the harbour obstacles.

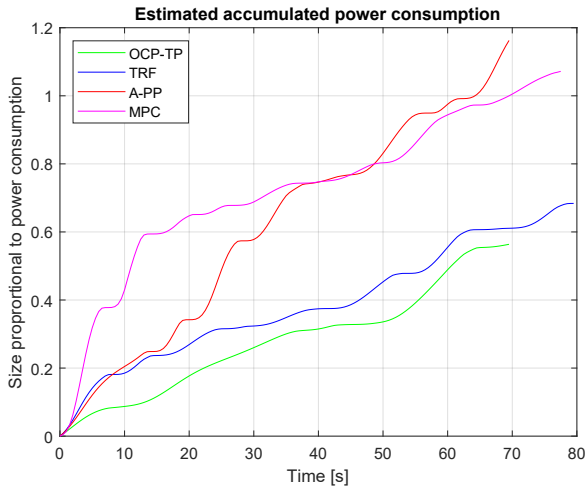


Figure 5.8: Scenario 1 Simulation: Accumulated power consumption.

Finally, Figure 5.10 shows that the TRF method has the lowest tracking error, while the OCP-TP is the second best method in this regard.

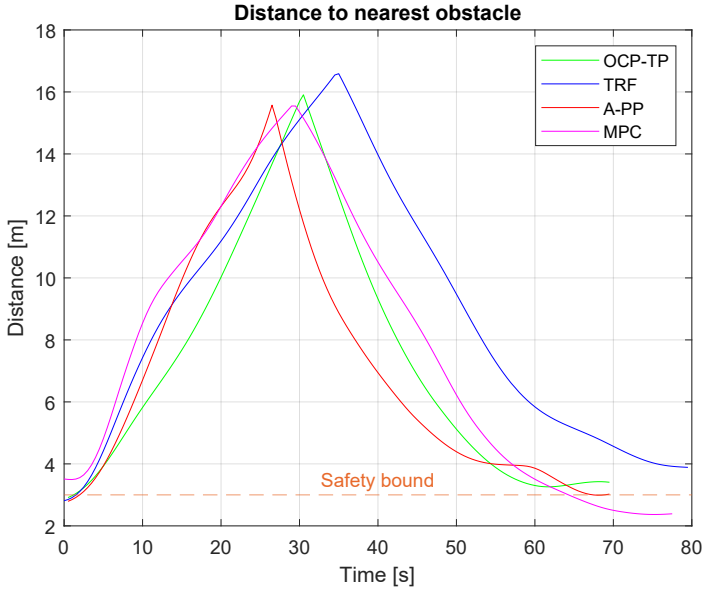


Figure 5.9: Scenario 1 Simulation: Distance to nearest obstacle.

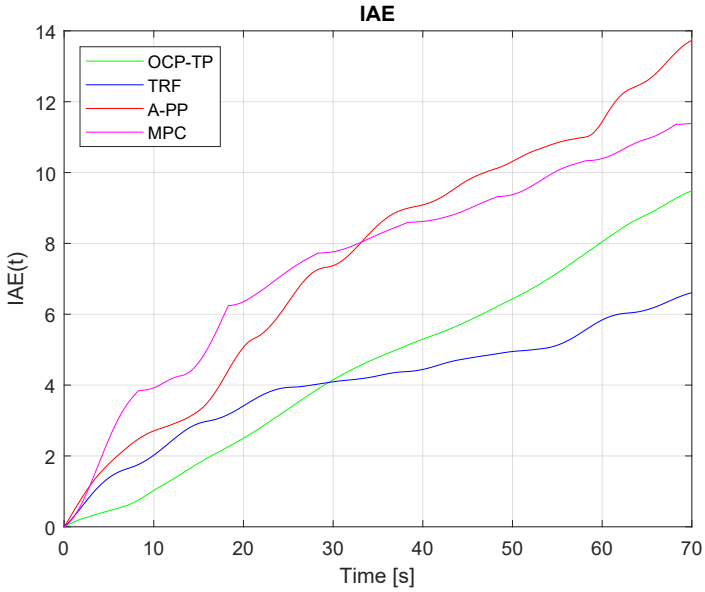


Figure 5.10: Scenario 1 Simulation: Integral Absolute Error from the trajectory tracking.

5.4.2 Scenario 2

This section contains the simulation results from Scenario 2. The scenario was performed as defined in Table 5.6 with the following docking methods:

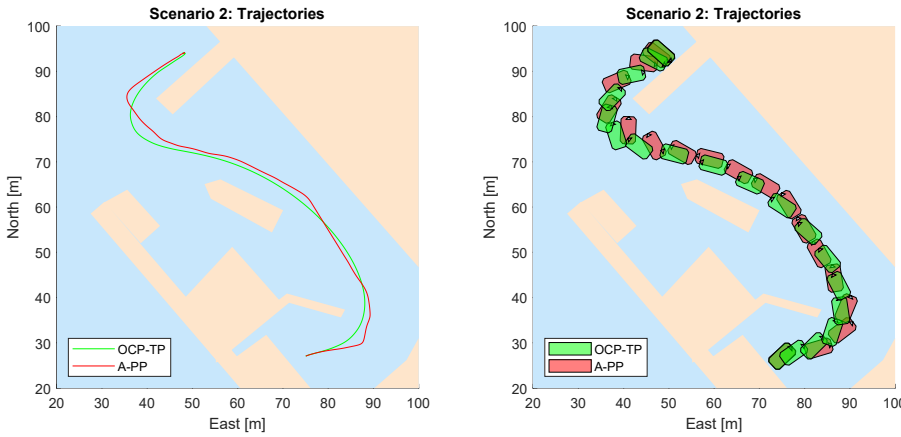
- OCP-TP
- A-PP

The end values of the selected metrics are found in Table 5.11 and an explanation of the results is given below.

Method	E_{mech}	IAE
OCP-TP	**1.2**	**21.4**
A-PP	2.6	33.2

Table 5.11: Scenario 2 Simulation: End values of metrics with the best method in green and bold.

The docking paths and individual state trajectories obtained in simulation for Scenario 2, are found in Figure 5.11 and Figure 5.12, respectively. Both the OCP-TP and the A-PP method successfully dock the ship, with the OCP-TP method appearing a bit more smooth in the edges. As Figure 5.14 shows, the harbour obstacles are always kept at a safe distance.



(a) Scenario 2: Followed path.

(b) Scenario 2: Path poses at 12 s intervals.

Figure 5.11: Scenario 2 Simulation: Trajectories of the different docking methods.

As in Scenario 1, Figure 5.13 indicates that the OCP-TP method has a significantly lower power consumption than the comparing A-PP method. The tracking error is also much smaller, as presented in Figure 5.15.

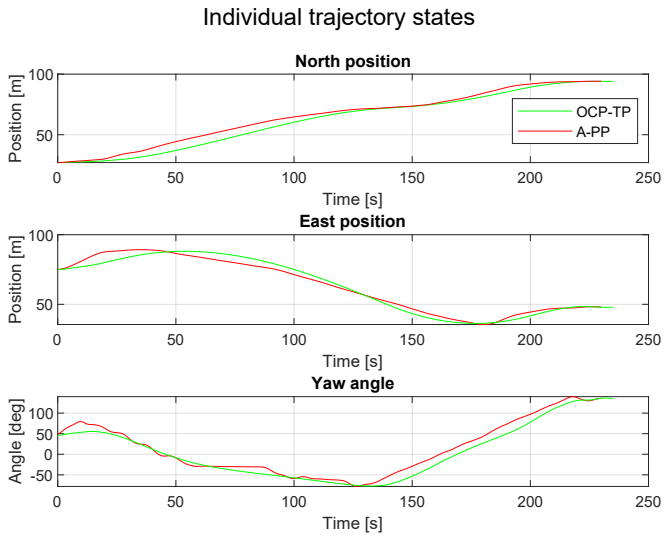


Figure 5.12: Scenario 2 Simulation: Pose trajectories.

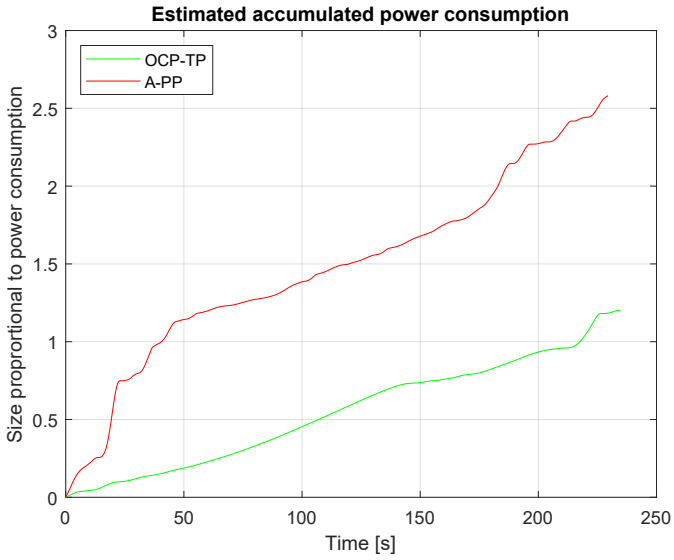


Figure 5.13: Scenario 2 Simulation: Accumulated power consumption.

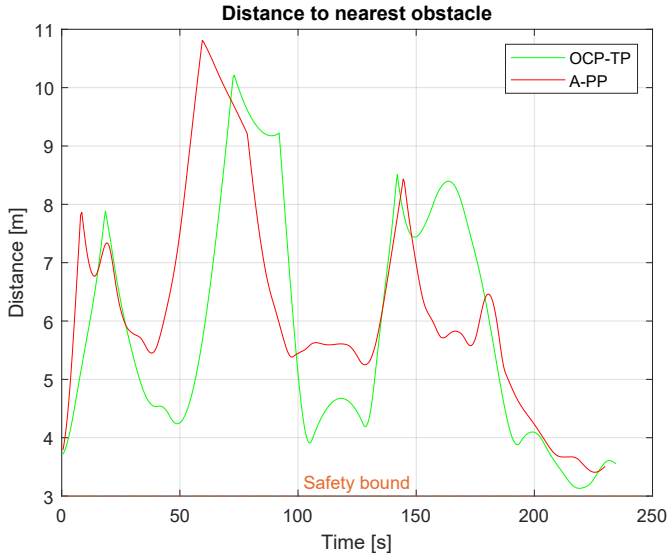


Figure 5.14: Scenario 2 Simulation: Distance to nearest obstacle.

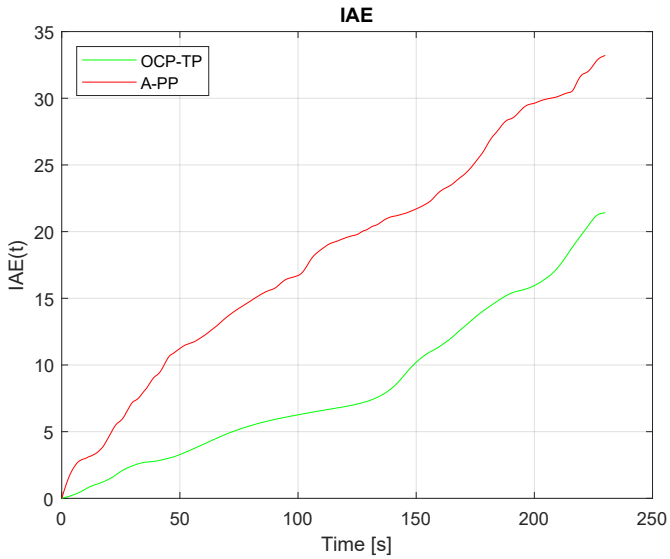


Figure 5.15: Scenario 2 Simulation: Integral Absolute Error from the trajectory tracking.

5.4.3 Scenario 3

This section contains the simulation results from Scenario 3. The scenario was simulated as defined in Table 5.8 with the following docking methods:

- OCP-TP
- A-PP

The end values of the selected metrics are found in Table 5.12 and an explanation of the results is given below.

Method	E_{mech}	**IAE**
OCP-TP	**1.45**	**27.3**
A-PP	2.30	30.7

Table 5.12: Scenario 3 Simulation: End values of metrics with the best method in green and bold.

Both the OCP-TP and the A-PP method successfully dock the vessel in the simulation of Scenario 3. As shown in Figure 5.16 and Figure 5.17, the OCP-TP method exploits the initial surge velocity of 1 m/s and calculates a smarter trajectory for the first 60 m of the docking operation. The individual state trajectories are presented in Figure 5.18, while Figure 5.20 shows that the harbour obstacle restrictions are never violated.

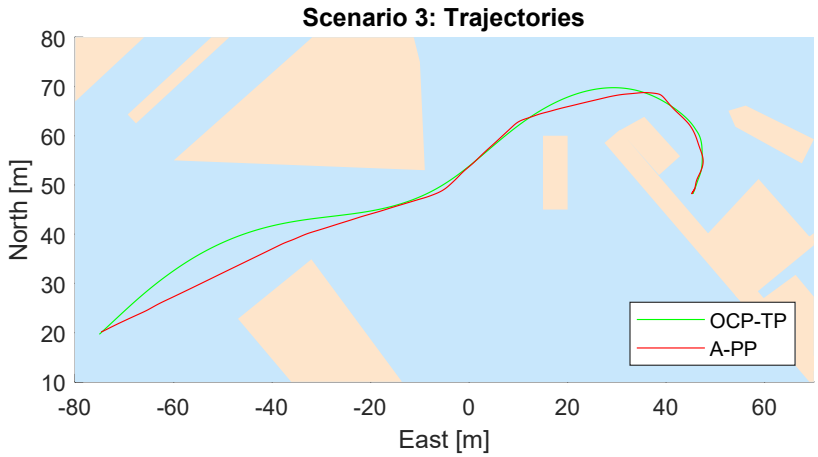


Figure 5.16: Scenario 3 Simulation: Followed path.

When it comes to the power consumption of the methods, Figure 5.19 once again shows that the OCP-TP uses the least amount of energy during the docking operation. However, the tracking error, shown in Figure 5.21, is more or less equal for the methods in this scenario, and actually greater for the OCP-TP in the first half of the operation.

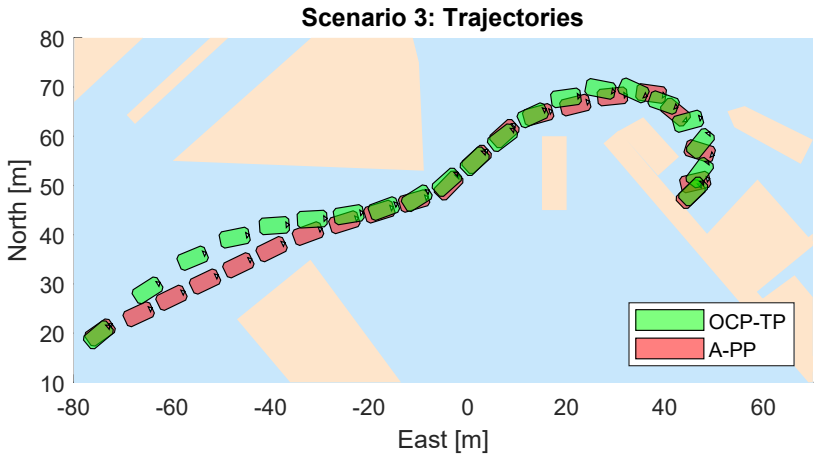


Figure 5.17: Scenario 3 Simulation: Path poses at 15 s intervals.

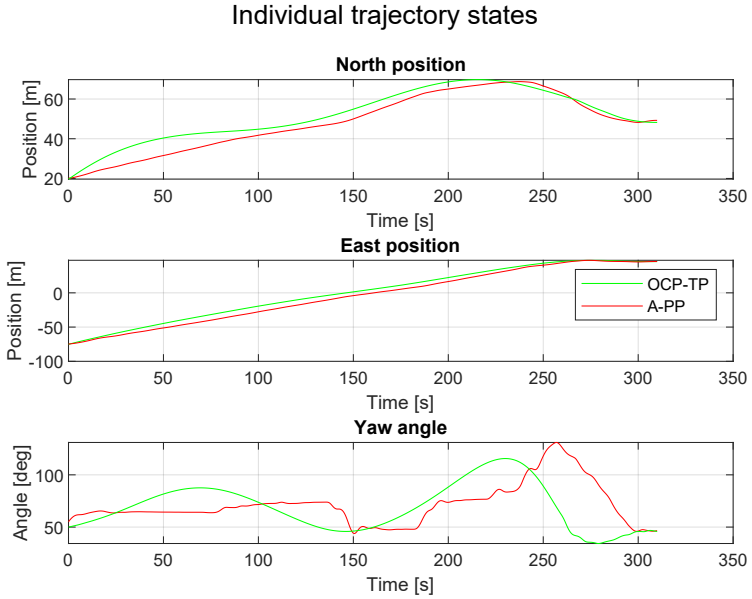


Figure 5.18: Scenario 3 Simulation: Pose trajectories.

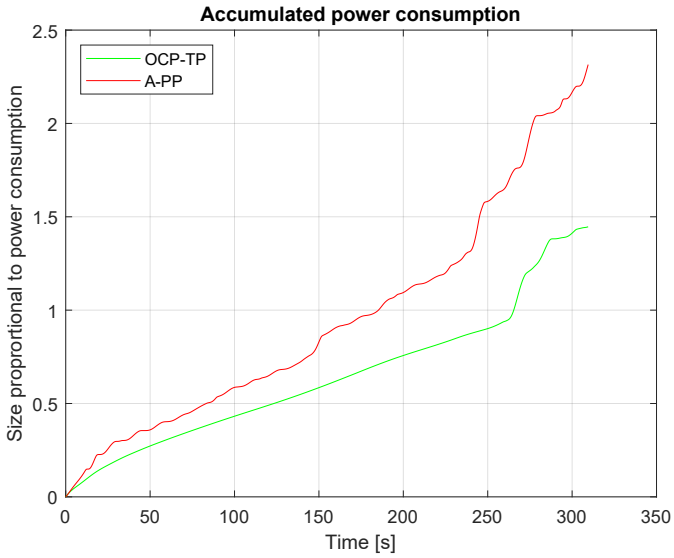


Figure 5.19: Scenario 3 Simulation: Accumulated power consumption.

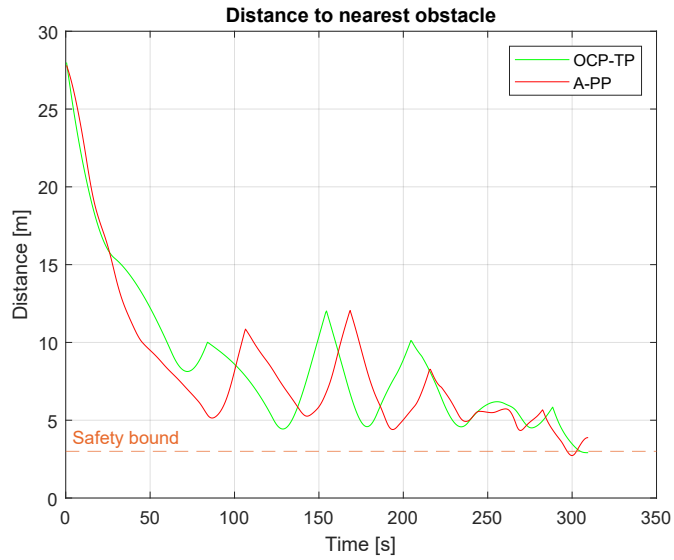


Figure 5.20: Scenario 3 Simulation: Distance to nearest obstacle.

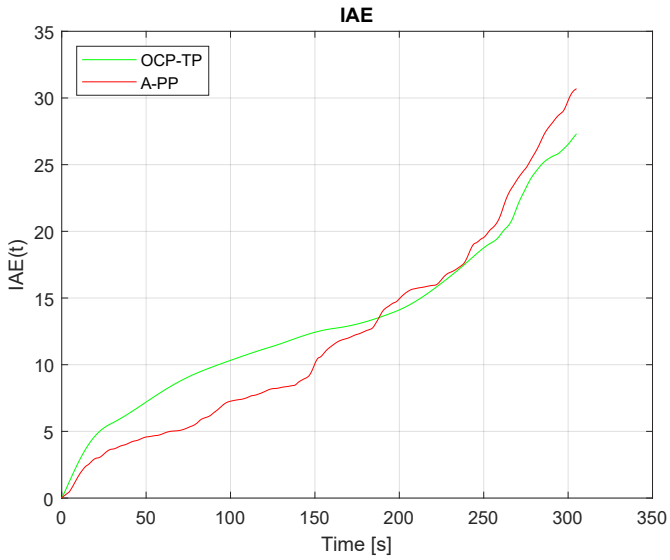


Figure 5.21: Scenario 3 Simulation: Integral Absolute Error from the trajectory tracking.

5.4.4 Remarks on the Simulation Results

The simulation results show that the OCP-TP is a promising automatic docking alternative. The planned trajectories are smooth and reasonably planned, OCP-TP has the lowest energy consumption of all methods in all scenarios and the tracking of the trajectory is satisfying, being mostly influenced by the tuning of the DP-controller. The next section will reveal if this is also the case in sea trials.

5.5 The milliAmpere Experimental Vessel

In order to test, compare and validate the OCP-TP in sea trials, a suitable vessel platform must be chosen. In this thesis, the prototype milliAmpere vessel is used, as it serves as an experimental platform for autonomous marine vessel research at NTNU. The milliAmpere vessel provides opportunities both for simulation and sea trials, as previous students have tuned the control system, developed thrust allocation algorithms, performed parameter estimation and created a realistic simulator.

Figure 5.22: The experimental vessel milliAmpere at rest at Fosenkaia in Trondheim in April 2020.

5.5.1 Vessel Characteristics

The development of the ferry started in 2017. It was built in aluminium in 1:2 scale of the desired final size. The vessel has an overall length of 5 m, a beam of 2.8 m and weighs 1670 kg. Like most ferries, milliAmpere has a symmetric design, but the hull is flat-bottomed with no keel. This makes the vessel quick to turn and very maneuverable, but also quite unstable in yaw and vulnerable to wind. The vessel is equipped with two electric azimuth thrusters capable of delivering 2 kW each, giving the vessel a top speed of 4.7 knots. The power source is six 24V DC lead-acid batteries with a total capacity of 600 Ah. A corner-view of the ferry can be seen in Figure 5.22

5.5.2 Navigation Sensors

The navigation system of the vessel relies on the following sensors:

- Real-Time Kinetic (RTK) Global Navigation Satellite System (GNSS) (Vector^{TM} VS330 GNSS Receiver)
- Inertial Measurement Unit (IMU) (Xsens MTi20)

The navigation system of the vessel uses an Error State Kalman Filter (ESKF) for sensor fusioning of the GNSS and IMU data, and with an RTK lock on the GNSS measurements, centimeter accuracy of the position can be achieved.

5.5.3 Computer and Software

The Onboard Computer (OBC) of the vessel is an Axiomtek eBOX670-883-FL with an Intel Core I7 processor. It runs the Ubuntu OS with the Kinetic ROS distribution and is accessed through a screen and keyboard as seen in Figure 5.23. The Radio Control (RC) controller, allowing for manual wireless control of the ship, is also visible in this image.

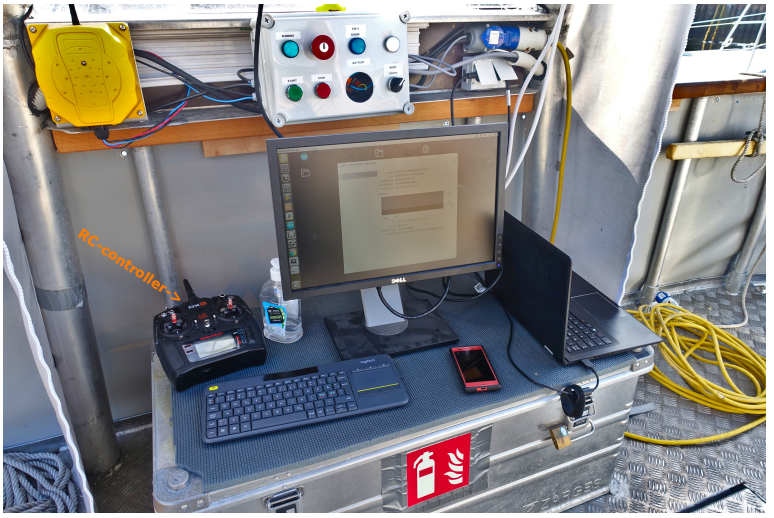


Figure 5.23: The control station of the ship, with screen, keyboard and RC controller.

5.6 Experimental Results and Discussion

With a functioning OCP-TP, a number of comparing methods, defined performance metrics and a suitable experimental vessel, sea trials can be performed. In this section, these experimental results will be presented and discussed. All experiments were performed between April 27 and 30, 2020, in the harbour area of Brattørkaia. A satellite image of this area is found in Figure 5.24. An extensive test plan was created in advance, to ensure that the limited time with milliAmpere was used efficiently. This test plan is found in Appendix B, and defines safety measures, how data will be collected and a detailed plan for how, where and when each scenario shall be performed.

5.6.1 Weather Conditions

During the three first days of testing, the weather conditions were stable and nice with no precipitation, calm waters and a light breeze from north-east. These conditions are

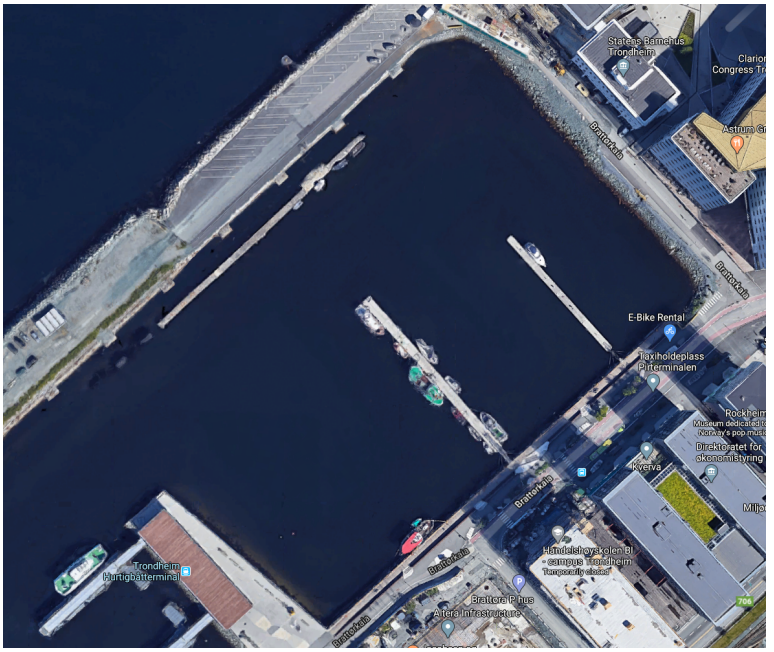


Figure 5.24: Satellite image of the location for the sea experiments.

depicted in Figure 5.25. However, for the last day of testing, April 30, the conditions worsened. The wind changed to a moderate breeze from the south, with occasional gusts up to 15 m/s. The only data collected that day were the ones labeled ”OCP-TP windy”.

Figure 5.25: The milliAmpere vessel during sea trials, April 27. As the fake hawk decoy in the top right corner indicates, the sea conditions were nice, with calm waters and wind speeds limited to light breeze.

5.6.2 Scenario 1

This section contains the experimental results from Scenario 1. The scenario was performed as defined in Table 5.4 with the following docking methods:

- OCP-TP
- TRF
- A-PP
- MPC
- Manual

The end values of the defined metrics are found in Table 5.13 and a detailed explanation of the results is given below.

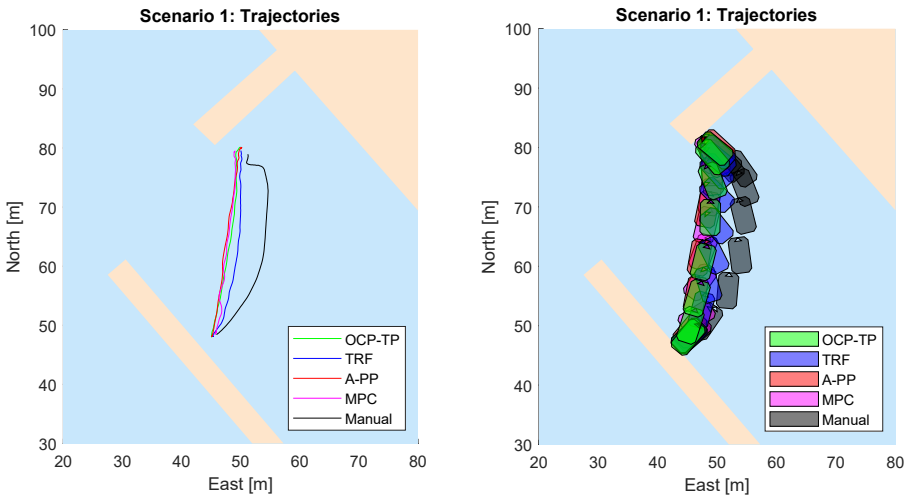
Method	E_{elec}	IAE	IAEW	C	AWT
OCP-TP	8.3	24.9	204.9	9.0	129.7
TRF	10.5	**14.8**	**140.4**	8.5	144.3
A-PP	13.1	15.2	197.8	9.1	131.6
MPC	11.8	15.1	179.1	10.1	147.0
Manual	**5.7**	–	–	**5.7**	**127.2**

Table 5.13: Scenario 1 Experimental: End values of metrics with the best method in green and bold.

The resulting paths and trajectories from this scenario is found in Figure 5.26. It is seen that the four automatic methods follow approximately the same path, with some differences in yaw angle for the TRF method. This is also evident when looking at the individual state trajectories given in Figure 5.27. The expected oscillations in the yaw angle caused by the lack of keel is also visible in this figure. The manually controlled run follows a much more curved path than the others, and also struggles more to settle in the final position.

Figure 5.28 illustrates how close to the nearest obstacle the different methods are during the scenario run. This figure shows that all of the methods steer clear of the obstacles during operation, and only comes close in the starting and ending phase of the mission, as expected.

The difference in energy usage for the scenario is shown in Figure 5.29. The OCP-TP is the automatic method using the least amount of energy, only beaten by the manual operator. While the automatic methods have a steadily increasing energy usage, the manual run consists of step increases. This is in compliance with the manual operating style in low speed, where small inputs are given and the correction of the boat's position is not happening constantly. This can explain the low energy usage.



(a) Scenario 1: Followed path.

(b) Scenario 1: Path poses at 10 s intervals.

Figure 5.26: Scenario 1 Experimental: Trajectories of the different docking methods.

In Figure 5.30, the IAE of the different runs is presented. This metric can only be computed for the automatic methods, as it measures how well the reference trajectory is followed. The figure indicates that OCP-TP has a substantially larger IAE than the other automatic methods for this scenario. When delving deeper into the data, an explanation to this observation seemed to be that the method struggled to converge to the desired waypoint when laying sideways. This was not present in the simulation run of Scenario 1, so it could be due to sudden wind gusts for this specific run. Still, errors in the model used in the OCP-TP or insufficient tuning of the DP-controller can be the reason. When looking at the tracking error vs power consumption given in Figure 5.31 the methods are more levelled due to the OCP-TP's low energy usage, but the TRF method is the one with the lowest IAEW.

During operation, the automatic methods created a lot more noise from the ship's actuators. This view is confirmed by Figure 5.33 which shows the amount of wear and tear on the actuators. However, the difference is not as large as expected. As indicated in the second plot of the figure, the change in the azimuth angles is substantially lower for the manual run. This change is what causes noise onboard, as some slack is present in the motors responsible for angular change. Still, the manual operator uses a more on-off strategy for controlling the ship, causing the change in propeller rate to be quite high. This contribution levels the wear and tear, and make all of the five methods approximately equal in this respect.

The passenger comfort metric, based on the accumulated acceleration of the vessel, is found in Figure 5.32. The OCP-TP method comes in third, undergoing more acceleration than the TRF method and the manual operator. This is not surprising, as the TRF method

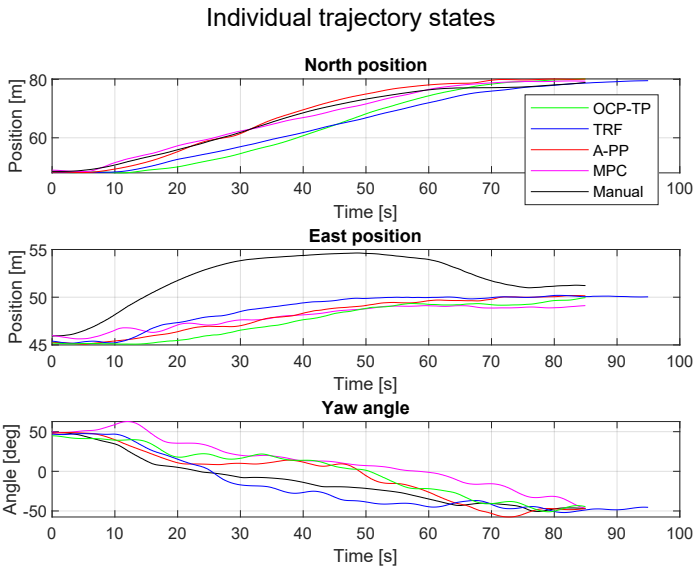


Figure 5.27: Scenario 1 Experimental: Pose trajectories.

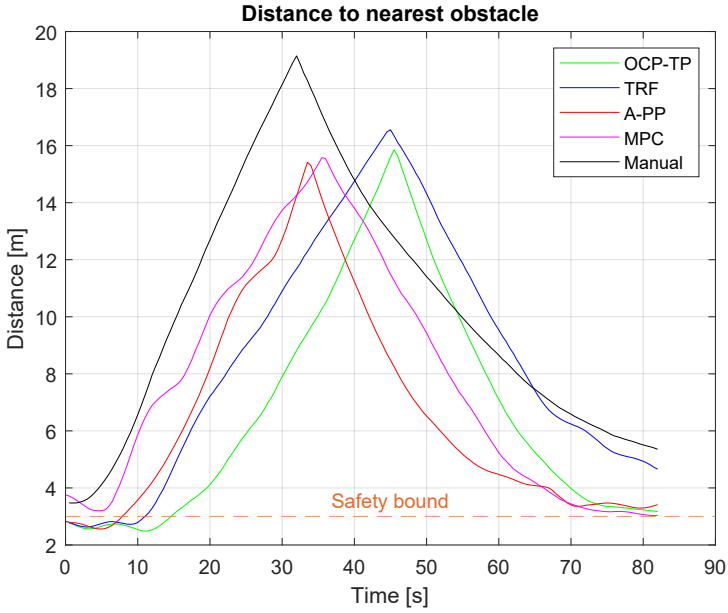


Figure 5.28: Scenario 1 Experimental: Distance to nearest obstacle.

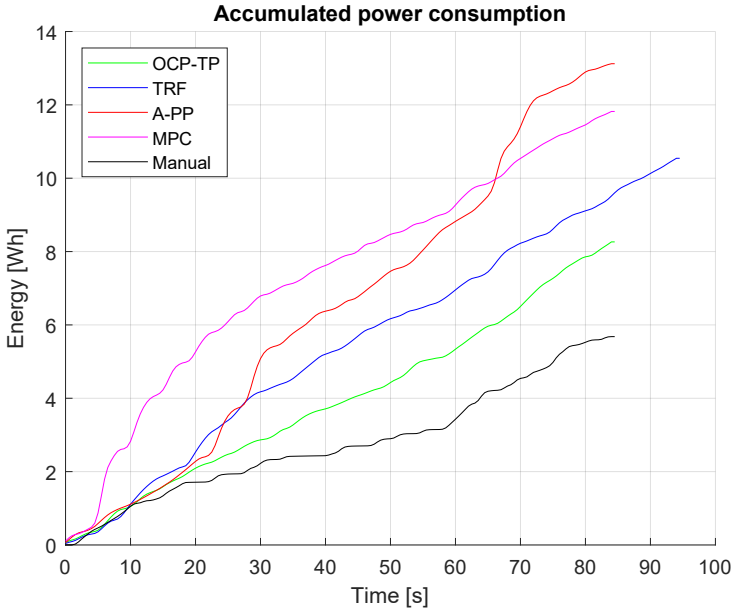


Figure 5.29: Scenario 1 Experimental: Total energy used by the different docking methods.

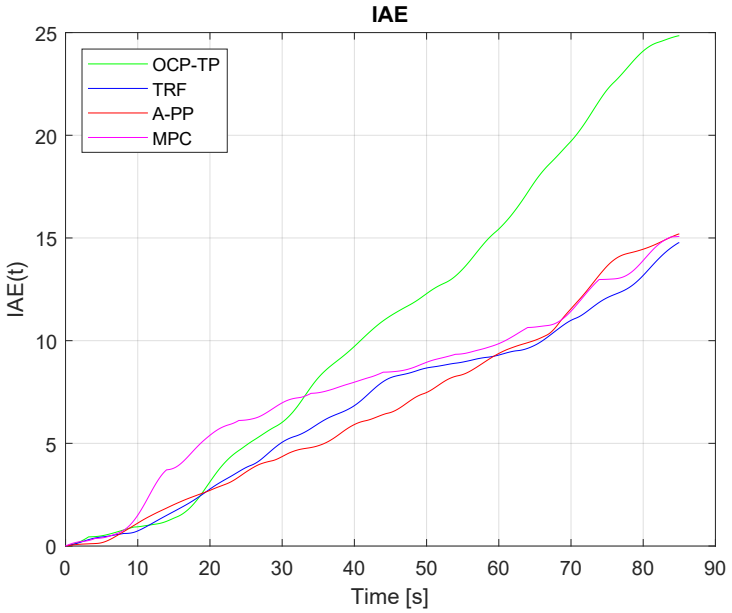


Figure 5.30: Scenario 1 Experimental: IAE.

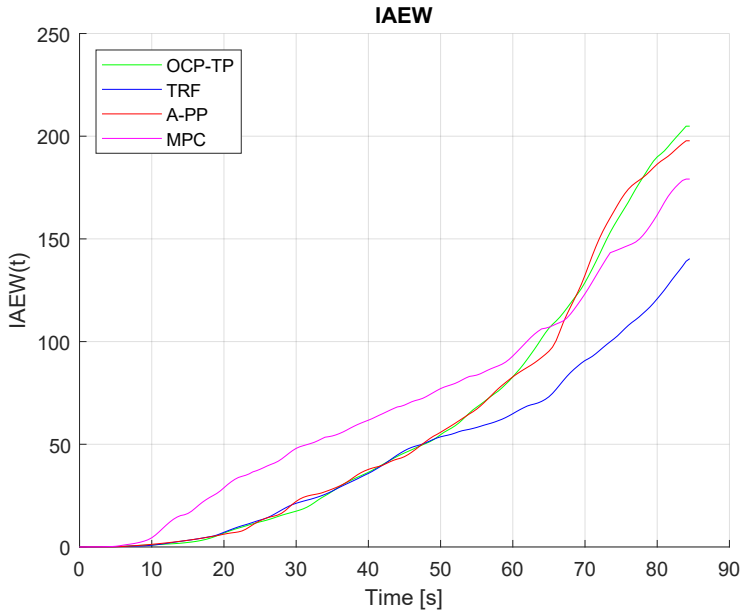


Figure 5.31: Scenario 1 Experimental: IAE multiplied by power consumption.

with its third order filter creates a very smooth reference trajectory and the manual operator has large periods of only drifiting, giving almost zero acceleration.

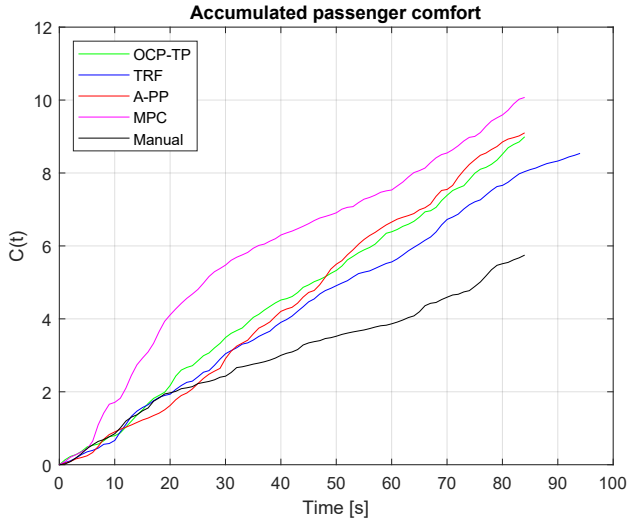


Figure 5.32: Scenario 1 Experimental: Experienced passenger comfort.

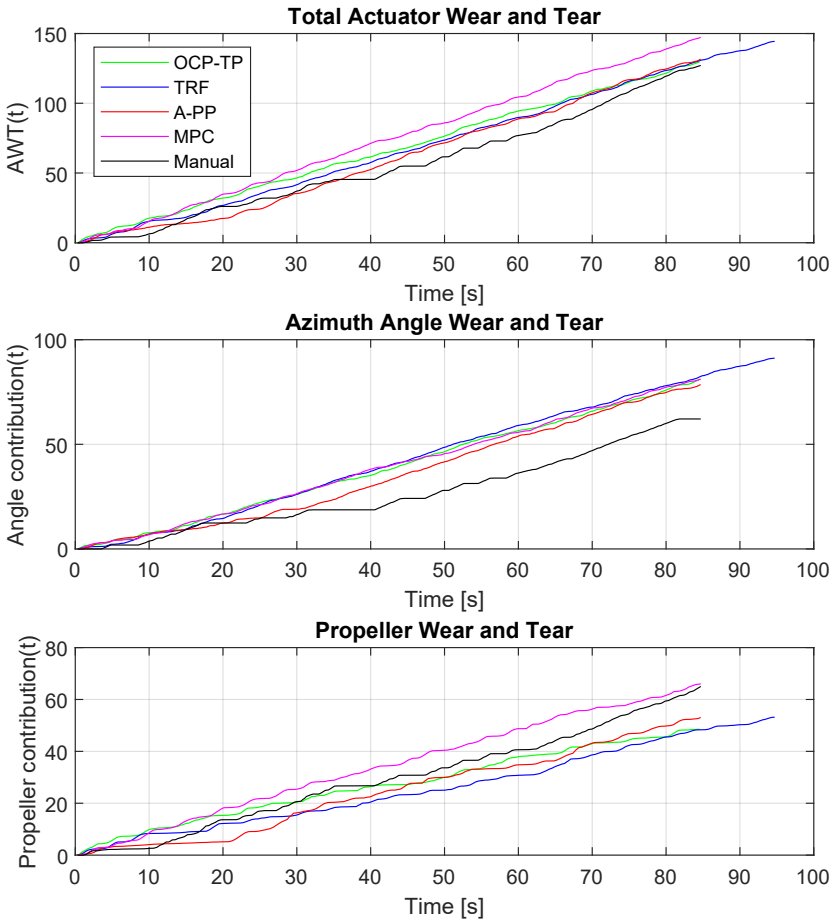


Figure 5.33: Scenario 1 Experimental: Total actuator wear and tear, followed by the separate wear and tear on azimuth angle motor and the propeller.

5.6.3 Scenario 2

This section contains the experimental results from Scenario 2. The scenario was performed as defined in Table 5.6 with the following docking methods:

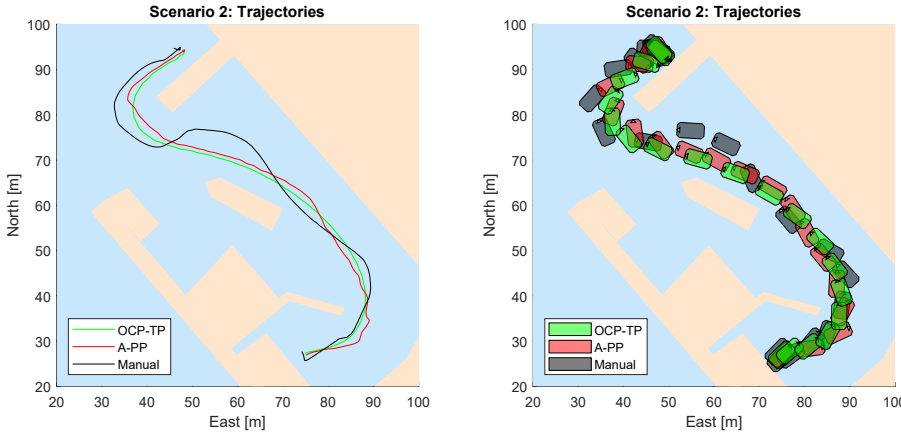
- OCP-TP
- A-PP
- Manual

The end values of the defined metrics are found in Table 5.14 and a detailed explanation of the results is given below.

Method	E_{elec}	IAE	IAEW	C	AWT
OCP-TP	**17.0**	**42.4**	**728.2**	**17.1**	373.1
A-PP	29.9	51.3	1541.0	28.1	396.2
Manual	24.8	–	–	24.7	**355.9**

Table 5.14: Scenario 2 Experimental: End values of metrics with the best method in green and bold.

The resulting paths and trajectories from this scenario is found in Figure 5.34. This scenario is far more complex than the previous, and the difference in the paths are more obvious. From Figure 5.34a it is clear that the OCP-TP method results in a smooth and precise path, while the red A-PP path and the black manual operator path are more prone to oscillations and unnatural movements. This view is further confirmed by the individual state results found in Figure 5.35.



(a) Scenario 2: Followed path.

(b) Scenario 2: Path poses at 12 s intervals.

Figure 5.34: Scenario 2 Experimental: Trajectories for the different docking methods.

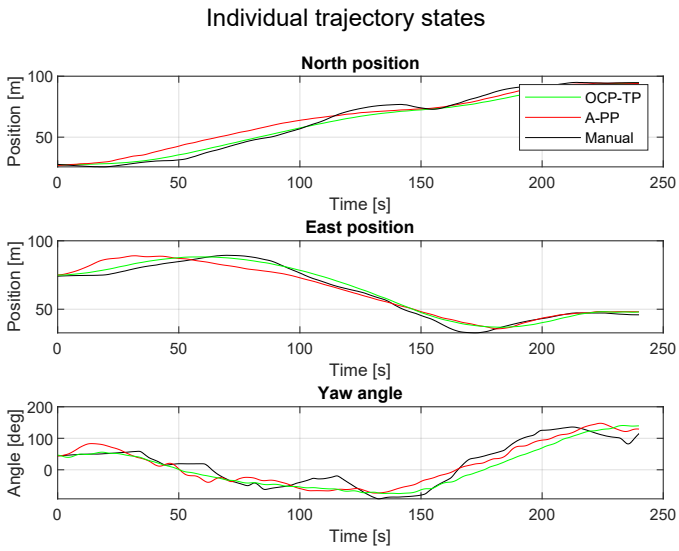


Figure 5.35: Scenario 2 Experimental: Pose trajectories.

The smooth and efficient docking trajectory created by the OCP-TP is also present in Figure 5.36. Here, it has the lowest total energy consumption among the methods tested in the scenario, and during the run the power consumption is increasing in a slow and steadily manner. For the manual and the A-PP method, energy consumption is substantially higher and also consists of more step-like increases.

Next, Figure 5.37 shows that most of the methods are able to steer clear of the obstacles during operation. However, the manual operator came quite close to the ship obstacle at time $t = 110$.

The IAE result found in Figure 5.38 demonstrates a slightly different situation than what was given in the previous scenario. Now, the OCP-TP method has the lowest IAE value and when multiplying with the energy usage, Figure 5.49 indicates that the IAEW value of OCP-TP is only half of that of the A-PP method.

With regards to wear and tear on the actuators, Figure 5.40 provides the same impression as in Scenario 1. The total wear and tear for the methods is approximately equal, but the manual operator changes less on the azimuth angles and more on the propeller rate.

Finally, the metric describing the passenger comfort is presented in Figure 5.41. Not surprisingly, the energy-efficient docking trajectory of the OCP-TP method provides the lowest amount of acceleration, while the manual operator and the A-PP method has 44% and 64% more acceleration, respectively.

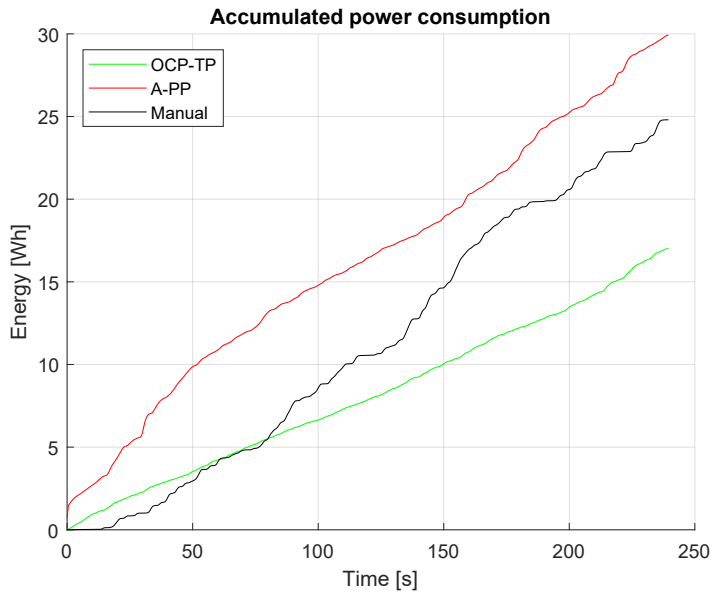


Figure 5.36: Scenario 2 Experimental: Total energy used by the different docking methods.

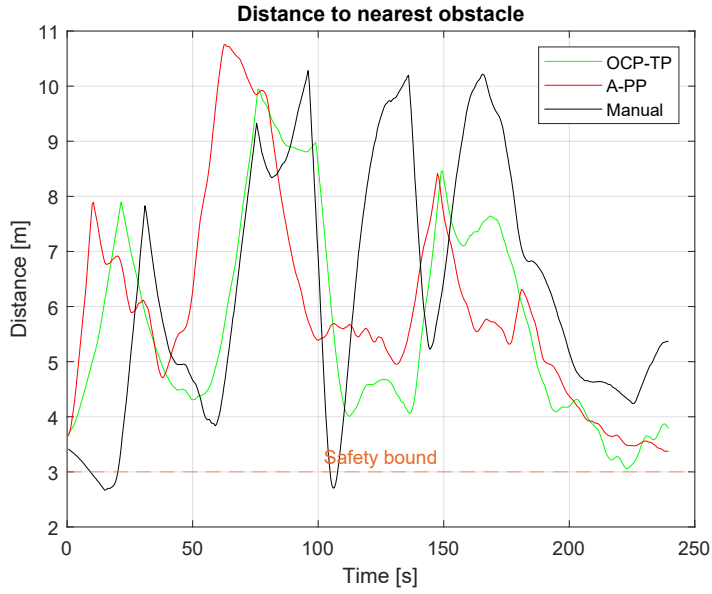


Figure 5.37: Scenario 2 Experimental: Distance to nearest obstacle.

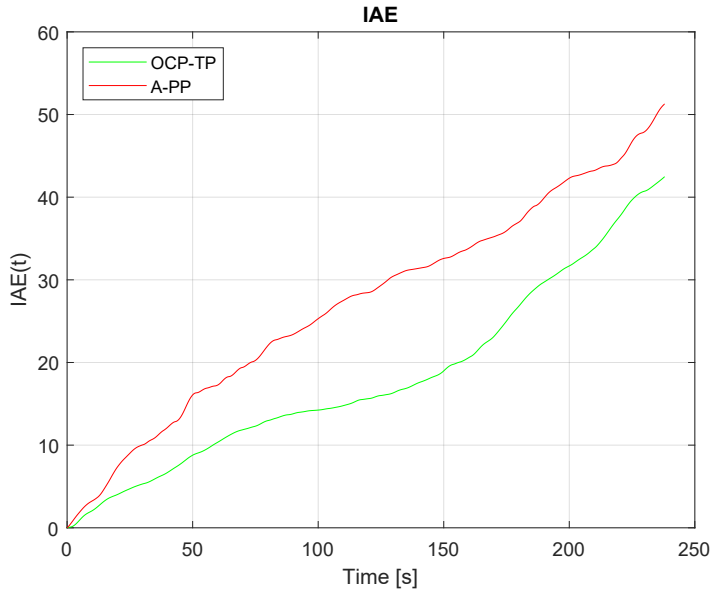


Figure 5.38: Scenario 2 Experimental: IAE.

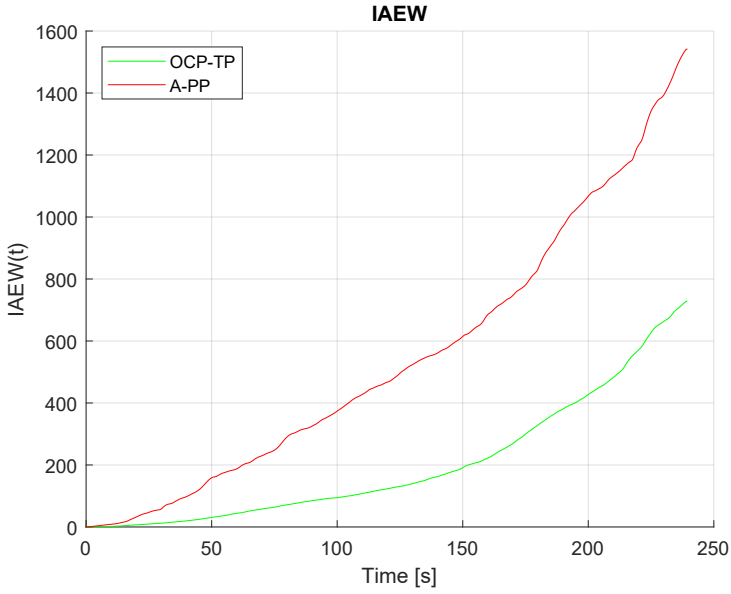


Figure 5.39: Scenario 2 Experimental: IAE multiplied by power consumption.

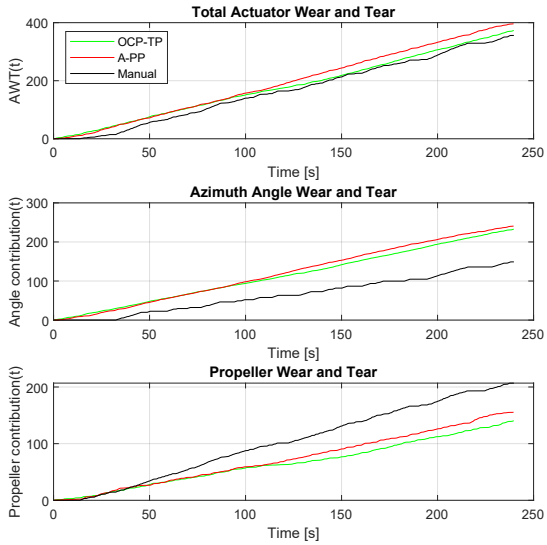


Figure 5.40: Scenario 2 Experimental: Total actuator wear and tear, followed by the separate wear and tear on azimuth angle motor and the propeller.

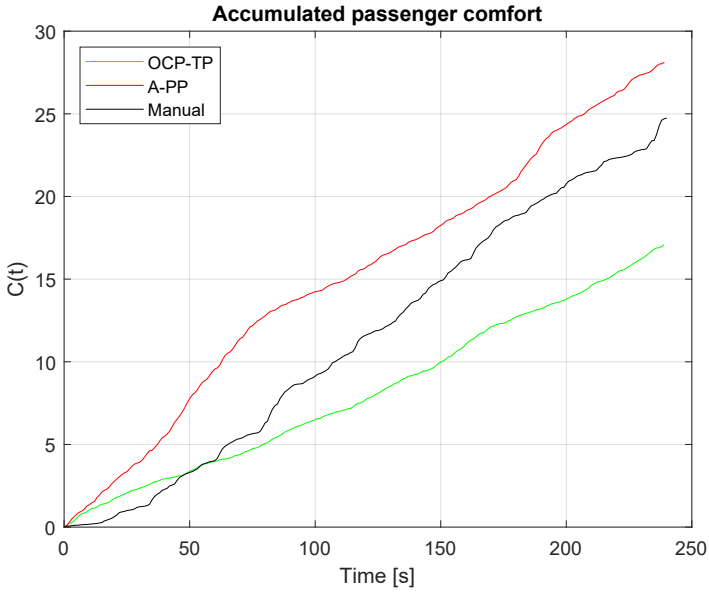


Figure 5.41: Scenario 2 Experimental: Experienced passenger comfort.

5.6.4 Scenario 3

This section contains the experimental results from Scenario 3. The scenario was performed as defined in Table 5.8 with the following docking methods:

- OCP-TP
- A-PP
- Manual

A second run of the OCP-TP method in windy conditions was also made in order to examine the robustness of the method. The end values of the defined metrics are found in Table 5.15, and a detailed explanation of the results is given below.

Method	E_{elec}	IAE	IAEW	C	AWT
OCP-TP	**25.4**	**60.3**	**1429.0**	**26.9**	522.1
A-PP	30.7	65.1	1987.0	32.4	509.5
Manual	28.9	–	–	28.3	557.2
OCP-TP windy	35.1	78.0	2570.0	36.4	**459.9**

Table 5.15: Scenario 3 Experimental: End values of metrics with the best method in green and bold.

While the previous scenarios only docked to virtual quays, Scenario 3 was selected to resemble what an actual automatic docking operation would look like. This included coming in from a transit speed of 1 m/s, letting the docking method take control of the ship, navigating between the obstacles, and eventually docking to the actual physical quay. The resulting paths and trajectories from this scenario are found in Figure 5.42 and Figure 5.43. When looking at the figures and the individual state trajectories in Figure 5.44 it is clear that all methods manage to dock the ship. The paths of the OCP-TP and the A-PP method are both quite smooth, while the manual operator path does not seem as sturdy. On the other hand, the automatic methods still have yaw angle oscillations present. It is also worth noting how well the final phase of the docking was performed. With accurate and precise pose measurements, all methods managed to get the vessel to rest at the end waypoint. As seen in Figure 5.45, the vessel made slow and controlled contact with the quay, allowing for passengers to get off.

The total energy used by the different methods is seen in Figure 5.46. It is shown that the OCP-TP method in has the lowest energy consumption in calm conditions. The manual operator follows closely behind, but the delicate positioning near the quay costs a lot of energy. Not surprisingly, the OCP-TP method is the most energy consuming process in windy conditions, spending 38% more power than in calm conditions.

Figure 5.47 shows that all methods are able to avoid the obstacles of the harbour environment. Still, the manual operator has the least impressive performance, coming quite close to the safety zone at some occasions.

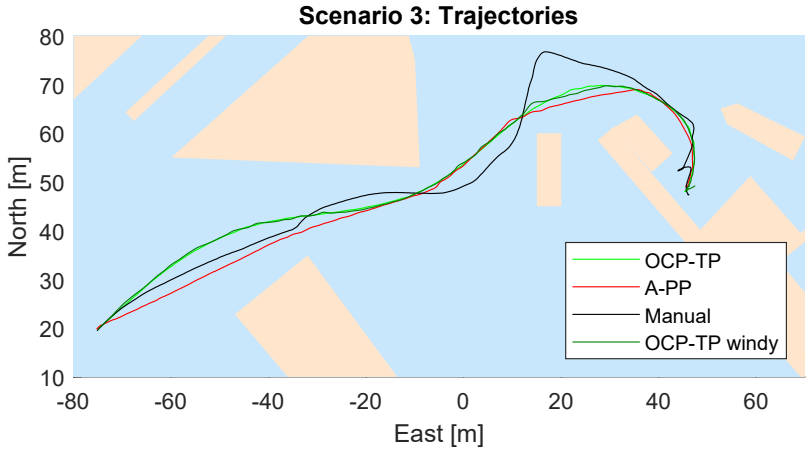


Figure 5.42: Scenario 3 Experimental: Followed path.

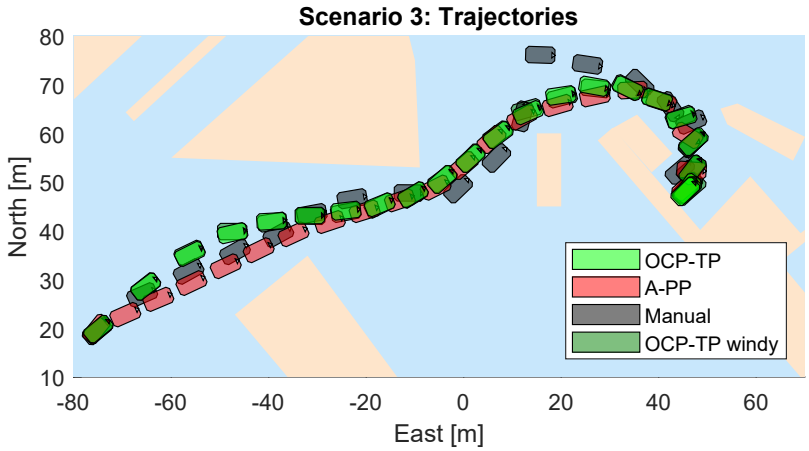


Figure 5.43: Scenario 3 Experimental: Path poses at 12 s intervals.

In Figure 5.48, the tracking capabilities of the methods are given. As expected, the OCP-TP has the worst tracking in wind, as sudden gusts and disturbances knock the vessel out of the desired position. In more calm conditions, the OCP-TP method has the best performance, closely followed by the A-PP method. Multiplying in the energy consumption, Figure 5.49 shows that the OCP-TP method comes out with the best IAEW value.

When it comes to wear and tear on the actuators, Figure 5.50 gives the same impression as in Scenario 1 and Scenario 2. The total wear and tear for the methods is approximately equal, but the manual operator changes less on the azimuth angles and more on the propeller rate. Still, it is interesting to see that the OCP-TP method appears to have the lowest

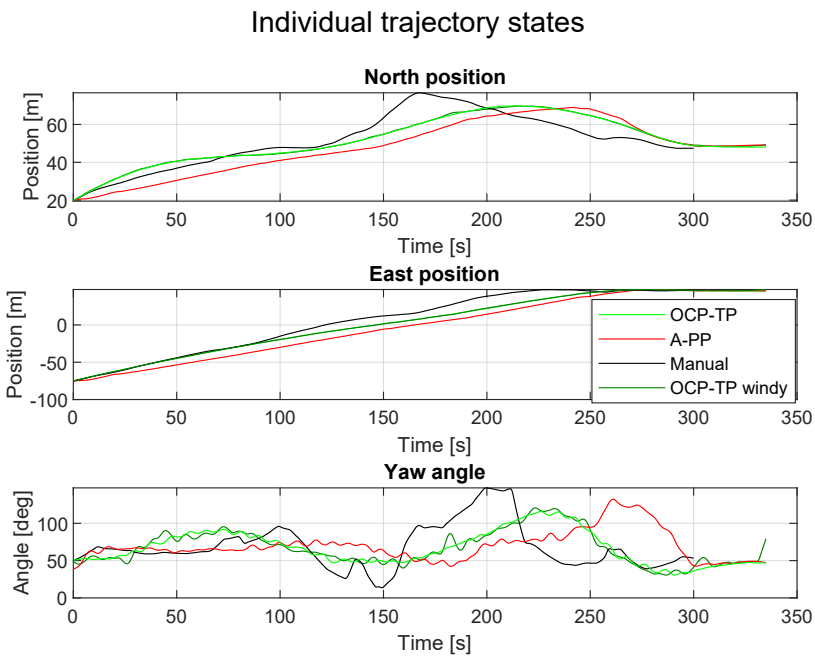
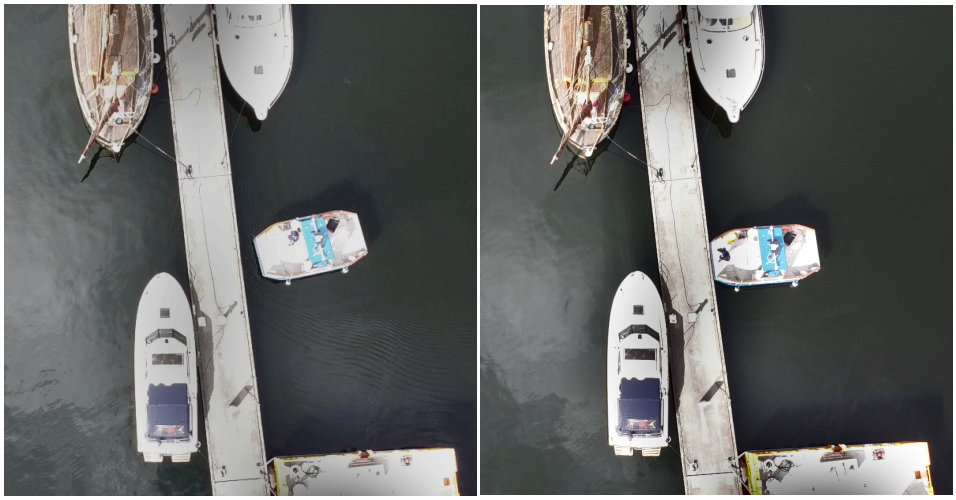


Figure 5.44: Scenario 3 Experimental: Pose trajectories.



(a) Final angle adjustment.

(b) Contact is made 15 s later.

Figure 5.45: Scenario 3: Actual aerial photographs of the final docking phase where contact is made with the quay, using the OCP-TP method in calm conditions.

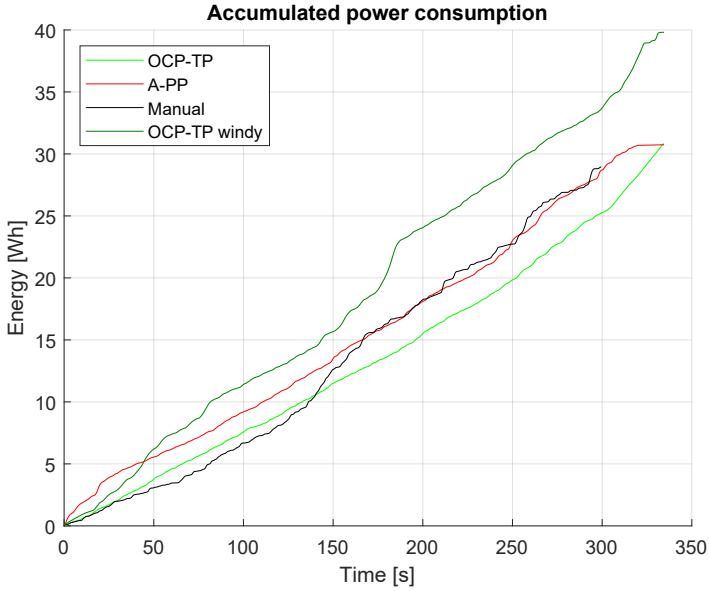


Figure 5.46: Scenario 3: Total energy used by the different docking methods. For the OCP-TP methods, the contact with the quay is made at $t = 308$ s, causing a substantial increase in the power consumption as the final angle adjustment is made while pushing against the quay.

total actuator wear and tear in wind. A possible explanation to this can be that great wind disturbances cause the propellers to work on a constantly high rate. Thus, they do not change their speed of revolution as much as they would in more calm waters.

Next, the metric describing the passenger comfort is presented in Figure 5.51. The energy-efficient docking trajectory of the OCP-TP method in calm waters provides the lowest amount of acceleration, followed by the human operator and the A-PP method. As expected, the OCP-TP experiences the most acceleration in wind, mainly due to sudden wind gusts.

A final remark should also be made in regards of the robustness of the OCP-TP method. As shown in this section, the windy conditions caused greater energy consumption, a larger tracking error and more acceleration. However, as seen in Figure 5.52, the vessel was still able to follow the reference trajectory with great precision. With a well-planned trajectory, good position estimates, smart thrust allocation and quick actuators, the only differences in the trajectory were some minor angle deflections.

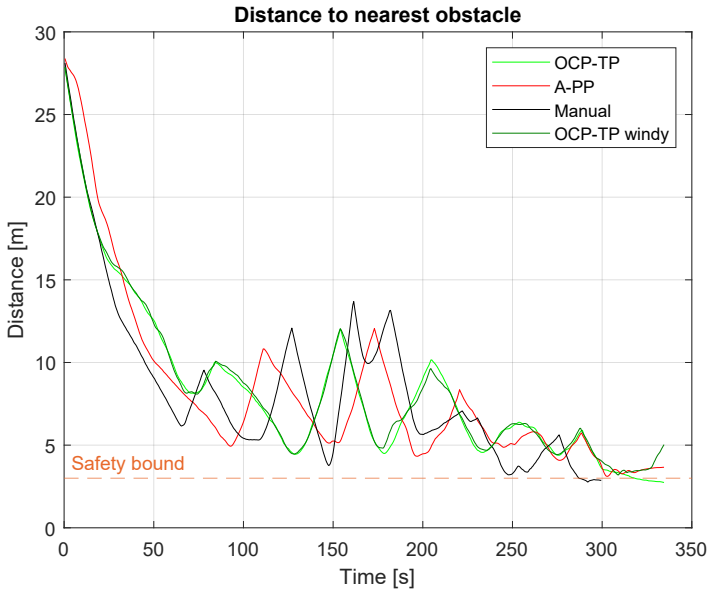


Figure 5.47: Scenario 3: Distance to nearest obstacle.

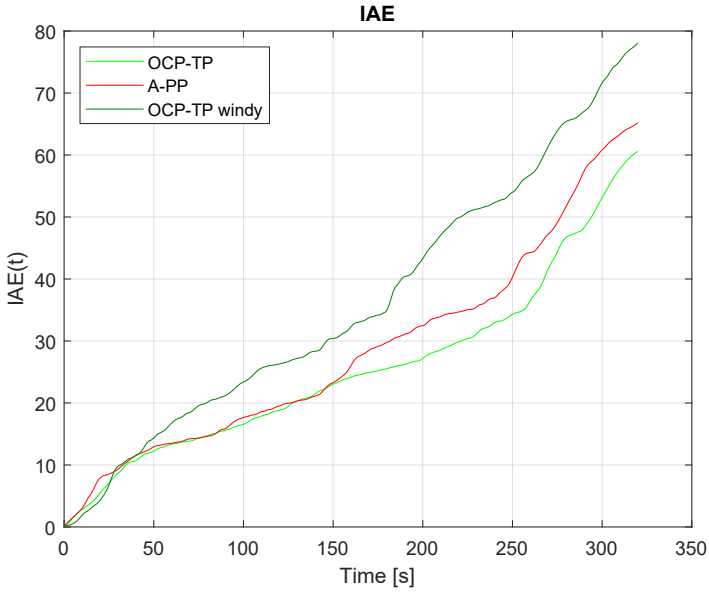


Figure 5.48: Scenario 3 Experimental: IAE.

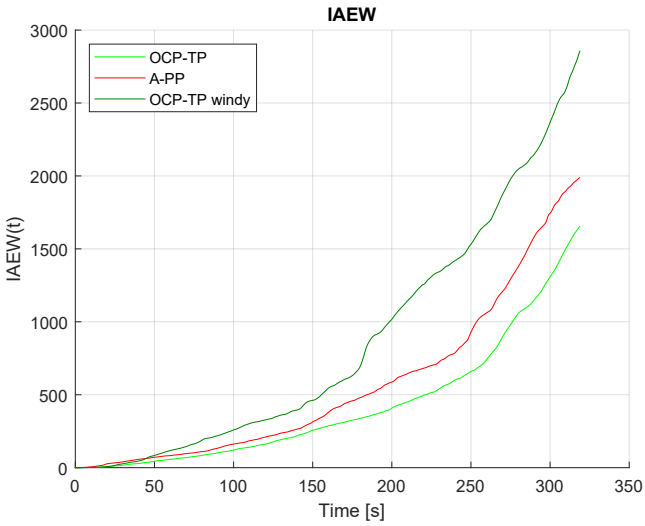


Figure 5.49: Scenario 3 Experimental: IAE multiplied by power consumption.

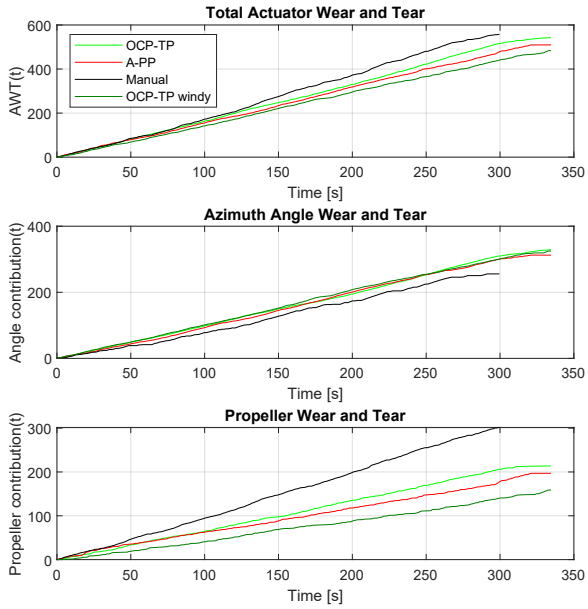


Figure 5.50: Scenario 3 Experimental: Total actuator wear and tear, followed by the separate wear and tear on azimuth angle motor and the propeller.

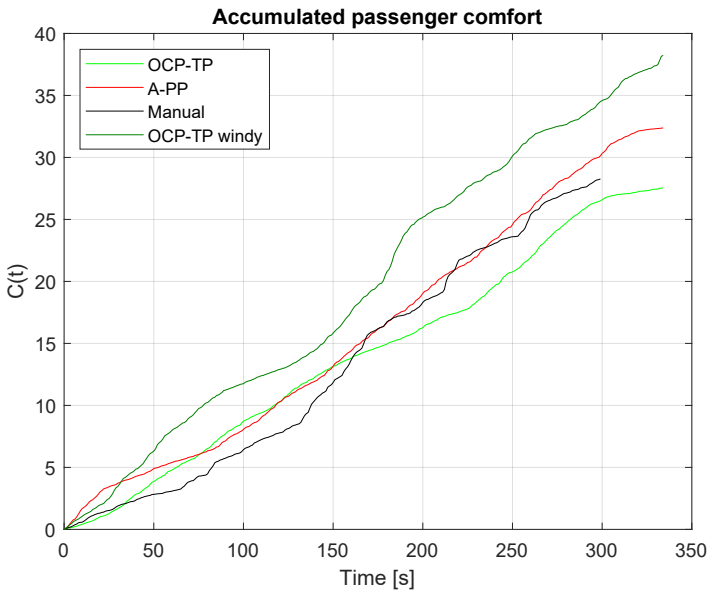


Figure 5.51: Scenario 3 Experimental: Experienced passenger comfort.

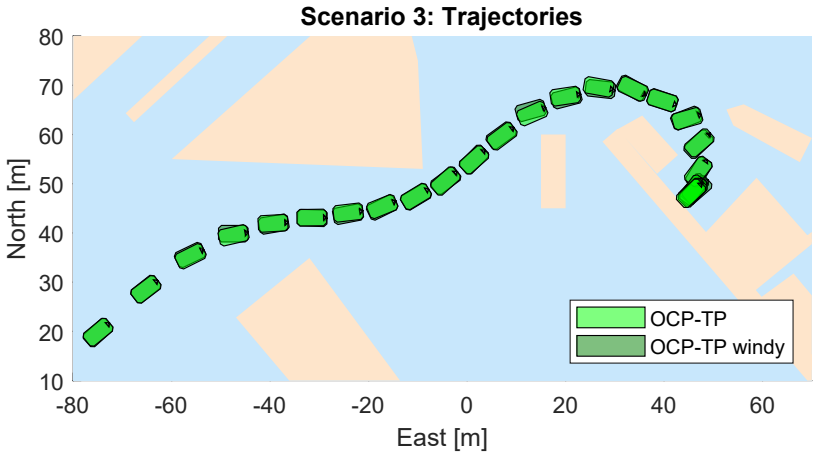


Figure 5.52: Scenario 3 Experimental: The OCP-TP docking method manages to follow the desired trajectory despite facing different wind loads.

5.6.5 Concluding Remarks

Having examined each of the scenarios in detail, some general conclusions can be made. In all scenarios, the different docking methods were able to control the ship to the designated end point while avoiding the various harbour obstacles. For the automatic methods, the instability in yaw became a source of heading oscillations. The trajectories proposed by the OCP-TP were quite smooth and precise, although the vessel struggled to converge completely in the final phase of Scenario 1. When being controlled by a manual operator, the movements of the ship were jumpier. In addition, the delicate task of settling at the final waypoint, proved more difficult for the manual operator than for the automatic methods.

In terms of energy efficiency, the OCP-TP method performs well. It has the lowest energy consumption in Scenario 2 and Scenario 3, while the manual operator has a better performance in Scenario 1. As suspected, a well-trained operator can outperform the OCP-TP by using small inputs and not constantly readjusting the position of the vessel. However, this comes with the cost of precision. In an obstacle-filled harbour like in Scenario 2, this might not be acceptable.

When it comes to wear and tear, the automatic methods come out approximately equally. This is probably because they use the same DP-controller and thrust allocation algorithm. These modules are what set the azimuth angles and choose the propeller revolution rates, and they probably have more influence on the wear and tear of the actuators than the reference trajectories created by the docking methods. The wear and tear when steering manually is slightly better than the automatic methods in Scenario 1 and Scenario 2, but perhaps not as large as one would think. Although changes in the azimuth angles do not happen as frequently, this is almost outweighed by constant changes in the propeller revolution speed. Ultimately, if one is to select a superior method in terms of wear and tear, the cost of angle changes and propeller revolution speeds must be weighted in some way.

Considering the accumulated acceleration providing a metric for passenger comfort, the OCP-TP comes out quite well. As the amount of acceleration is closely coupled to the power consumption, it is no surprise that this method has the lowest value in Scenario 2 and Scenario 3. Still, it must be noted that the maximum amount of acceleration for any method in any scenario never exceeded $0.13\,\mathrm{m/s^2}$, $0.17\,\mathrm{m/s^2}$ and $0.06\,\mathrm{rad/s^2}$ in surge, sway and yaw, respectively. These values are far lower than the limit values for what is considered comfortable, as stated in Table 5.2.

All in all, the experimental results have shown that the OCP-TP is an energy-efficient and reliable automatic docking alternative. The generated trajectories are smooth, with sufficient clearing to the given obstacles. With accurate pose estimates, a properly tuned controller and thrust allocation algorithm, the OCP-TP can lead the vessel from transit speed, through an obstacle-filled environment, and put it safely at rest at the quay, even in windy conditions.

Chapter 6

Conclusion and Further Work

This thesis covers the development and implementation of the Optimal Control Problem Trajectory Planner (OCP-TP), an energy-efficient, trajectory planning method for automatic docking operations.

First, the mathematical vessel notation is defined, and important docking theory is presented. This includes a decoupled 3 Degree of Freedom (DOF) vessel model, optimization theory, motion control and various strategies for control allocation. Introductions to obstacle representation, A* path planning and the Robot Operating System (ROS) are also made.

Next, the implementation details of the OCP-TP are given. The equations forming the optimizable vessel model from the preceding project thesis [15] are shown, and the transcription of the Optimal Control Problem (OCP) into a solvable Nonlinear Program (NLP) is explained. A new method, capable of turning map polygons into optimizable rectangular obstacles is proposed. This method represents the perimeter of the land polygons as a set of long, narrow rectangular obstacles. By choosing a sufficiently large obstacle width, the interior of the land polygon becomes unavailable and the desired obstacle avoidance capability is achieved. To ensure feasibility and efficiency, the OCP-TP contains an A*-based warm-starting algorithm that creates an initial trajectory guess. The trajectory publisher, the DP tracking controller, and the organization of the whole OCP-TP system are also described and illustrated.

Then, the improved developer GUI, created in JavaScript (JS) and Hypertext Markup Language (HTML) is presented. The interface is augmented with a graphical reference monitor, to give the operator a quick way of telling how well the vessel follows the current reference pose. This reference monitor comes with different zoom levels, supports body and North-East-Down (NED) frames and also provides numerical values for the tracking errors. To be in control of the intentions of the vessel, trajectory plotting capabilities are added to the GUI map. With real-time updates, the remaining vessel trajectory is shown in

the map and the feasibility can be verified manually by the operator. A power usage plot is also added, to quantify the energy-efficiency of various docking methods.

Further, to asses the OCP-TP, the four docking methods Trajectory Reference Filter (TRF), Model Predictive Control (MPC), A* Path Planner (A-PP) and Manual are defined. The two first are automatic docking methods designed by others, while the A-PP is created specifically for this thesis. The manual method lets the operator take control of the vessel through the sticks of the RC-controller.

To evaluate the performance of the docking methods, six metrics are selected. These are chosen to be the power consumption E_{elec}, trajectory tracking IAE, trajectory tracking with respect to power consumption IAEW, obstacle avoidance DNO, passenger comfort C and actuator wear and tear AWT. The mathematical definitions of these metrics are provided, along with numerical values for the normalization parameters. Also, three different docking scenarios are defined. These increase in complexity, with the third and final scenario simulating an actual docking operation by entering the harbour at transit speed, navigating through an obstacle-filled environment before docking at the physical quay.

Finally, the results from the simulations and sea trials of the OCP-TP and the comparing methods are presented. The simulation results indicate that the OCP-TP is more energy-efficient than the comparing methods, and the results from the sea trials confirm this hypothesis.

During sea trials, the OCP-TP is found to have the lowest energy consumption in two of the scenarios, only beaten by the manual operator in the first one. The planned trajectories of the OCP-TP are smooth and precise, and the vessel successfully manages to dock at the desired end points in all scenarios. The harbour obstacles are also taken into account and kept at a distance at all times. The tracking capabilities and the amount of wear and tear of the OCP-TP is found to be similar to the comparing methods, as they all rely on the same DP-controller and thrust allocation algorithm. The passenger comfort metric is closely related to the energy consumption, thus, the OCP-TP performes well also in this regard. It is unmatched by the other automatic methods and even beats the manual operator in two of the scenarios. Finally, results from a windy run using the OCP-TP demonstrates that the method is reliable and successfully docks the vessel, despite facing significant environmental disturbances.

Still, the OCP-TP has room for extensions, modifications and improvements. The following topics related to the OCP-TP could be further explored:

- Recalculation of the docking trajectory with dynamic obstacle avoidance. As of now, the OCP-TP only plans the docking trajectory once, using a predefined map for obstacle avoidance. If the vessel was equipped with sensors able to sense harbour obstacles and the OCP-TP was made sufficiently fast, the obstacle map could be updated in real-time and the OCP-TP could be run as an MPC. In addition, if the future positions of surrounding vessels are known, the OCP structure makes it possible to create moving obstacles to increase the flexibility of the docking trajectory planner.

- Moment-of-impact controller. Although the OCP-TP successfully docked the vessel, it would be desirable with a different control strategy for the final stage where contact is made with the quay. This controller should rely less on the Global Navigation Satellite System (GNSS) measurements, and rather use information from proximity sensors or cameras to obtain the relative distance to the quay. In addition, the physical impact with the quay should be dealt with, and the vessel should end up maintaining its final docking pose by pushing towards the quay with a suitable force.

- Global vessel trajectory planning. In this thesis, the OCP-TP was used for automatic docking of a vessel. But the method could also be applicable for the entire vessel operation, including undocking from point A, sea transit and final docking at point B. Realising such a planner would require changes in the OCP model, switching between different controllers and a more efficient way of computing trajectories, as the problem will increase substantially in size.

- Tuning of the DP tracking controller for milliAmpere. During sea trials, the controller was not able to control the heading of milliAmpere precisely, resulting in yaw angle oscillations. Improved controller tuning would likely result in lower energy usage and less actuator wear and tear.

Bibliography

[1] Tore Stensvold. Verdens første autonome skip i drift skal erstatte 40.000 vogntogturer i året. https://www.tu.no/artikler/verdens-forste-autonome-skip-i-drift-skal-erstatte-40-000-vogntogturer-i-aret/382717, 2017. Accessed on 05.12.2019.

[2] DNV-GL. The ReVolt - a new inspirational ship concept. https://www.dnvgl.com/technology-innovation/revolt/index.html, 2019. Accessed on 05.12.2019.

[3] Unni Skoglund. Førerløse ferger kan erstatte gangbruer. https://gemini.no/2018/06/forerlose-ferger-kan-erstatte-gangbruer/, 2018. Accessed on 15.05.2020.

[4] Adrian Søgnen. Nå skal fergene krysse fjorden automatisk. https://e24.no/hav-og-sjoemat/i/y3jWra/naa-skal-fergene-krysse-fjorden-automatisk, 2018. Accessed on 15.05.2020.

[5] Volvo Penta. Volvo penta unveils pioneering self-docking yacht technology. https://www.volvopenta.com/marineleisure/en-en/news/2018/jun/volvo-penta-unveils-pioneering-self-docking-yacht-technology.html, 2018. Accessed on 10.09.2019.

[6] H. Yamato. Automatic berthing by the neural controller. In *Proceeding of Ninth Ship Control Systems Symposium. Bethesda, Maryland, USA*, volume 3, pages 3.183–3.201, September 1990.

[7] Karim Djouani and Yskandar Hamam. Ship optimal path planning and artificial neural nets for berthing. In *Proceedings of oceans'94. Brest, France*, September 1994.

[8] George Rae and Samuel Smith. A fuzzy rule based docking procedure for autonomous underwater vehicles. In *OCEANS '92 Mastering the Oceans Through Technology. Newport, RI, USA*, 1992.

[9] Minh-Duc Le, Hai nam Nguyen, and Khac-Bao Nguyen. Control of large ship motions in harbor maneuvers by applying sliding mode control. In *IFAC Proceedings Volumes PSYCO 04. Yokohama, Japan*, volume 37, pages 309–314, September 2004.

[10] Kouichi Shouji, Kohei Ohtsu, and Sumitoshi Mizoguchi. An automatic berthing study by optimal control techniques. In *IFAC Proceedings Volumes CAMS '92. Genova, Italy*, volume 25, pages 185–194, April 1992.

[11] Karim Djouani and Yskandar Hamam. Minimum time-energy trajectory planning for automatic ship berthing. *IEEE Journal of Oceanic Engineering*, 20:4–12, 1995.

[12] Naoki Mizuno, Yosuke Uchida, and Tadatsugi Okazaki. Quasi real-time optimal control scheme for automatic berthing. In *IFAC Proceedings MCMC'2015. Copenhagen, Denmark*, volume 48, pages 305–312, August 2015.

[13] Andreas B. Martinsen, Anastasios M. Lekkas, and Sebastien Gros. Autonomous docking using direct optimal control. In *Proceedings of the 12th IFAC Conference on Control Applications in Marine Systems, Robotics and Vehicles (CAMS). Daejeon, South Korea*, 2019.

[14] Glenn Bitar, Vegard N. Vestad, Anastasios M. Lekkas, and Morten Breivik. Warm-started optimized trajectory planning for asvs. *CAMS 2019. Daejeon, South Korea*, 2019.

[15] Eivind Duus Molven. Energy-optimized automatic docking for marine vessels. Project report: Norwegian University of Science and Technology (NTNU). December 2019.

[16] The Society of Naval Architects and Marine Engineers. Nomenclature for treating the motion of a submerged body through a fluid. *Technical and Research Bulletin*, 3, 1950.

[17] Thor Inge Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, 2011.

[18] Jorge Nocedal and S. Wright. *Numerical Optimization*. Springer, 2006.

[19] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57, 2006.

[20] Asgeir J. Sørensen. *Lecture notes: Propulsion and Motion Control of Ships and Ocean Structures*. Department of Marine Technology, Norwegian University of Science and Technology (NTNU), 2013.

[21] Tor A. Johansen, Thor I. Fossen, and Svein P. Berge. Constrained nonlinear control allocation with singularity aviodance using sequential quadratic programming. *IEEE Transactions on Control Systems Technology*, 12:211–216, 2004.

[22] Tobias R. Torben, Astrid H. Brodtkorb, and Asgeir J. Sørensen. Control allocation for double-ended ferries with full-scale experimental results. In *18th IFAC World Congress. Daejeon, South Korea)*, 2019.

[23] David D. Morrison, James D. Riley, and John F. Zancanaro. Multiple shooting method for two-point boundary value problems. *Communications of the ACM*, 5: 613–614, 1962.

[24] Qi Gong and L. R. Lewis annd I. Michael Ross. Pseudospectral motion planning for autonomous vehicles. *Journal of Guidance, Control and Dynamics*, 32:1039–1044, May 2009.

[25] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions and Systems Science and Cybernetics*, 4:100–107.

[26] John Carlton. *Marine Propellers and Propulsion*. Butterwoth-Heinemann, 1994.

[27] Vegard Nitter Vestad. Automatic and practical route planning for ships. Master's thesis, Norwegian University of Science and Technology (NTNU), June 2019.

[28] David H. Wu. Proactive maritime collision avoidance based on historical AIS data. Master's thesis, Norwegian University of Science and Technology (NTNU), June 2019.

[29] Priyadarshi Bhattacharya and Marina L. Gavrilova. Roadmap-based path planning - using the voronoi diagram for a clearance-based shortest path. *IEEE Robotics & Automation Magazine*, 15:58–66, 2008.

[30] Glenn Bitar, Andreas B. Martinsen, Anastasios M. Lekkas, and Morten Breivik. Trajectory planning and control for automatic docking of ASVs with full-scale experiments.

[31] Norsk Elektroteknisk Komite. *Maritime navigation and radiocommunication equipment and systems - ECDIS - Operational and performance requirements, methods of testing and required results*, 2015.

[32] Mikkel Eske Nørgaard Sørensen and Morten Breivik. Comparing nonlinear adaptive motion controllers for marine surface vessels. In *Proceedings of the 10th IFAC Conference on Manoeuvring and Control of Marine Craft MCMC: Copenhagen, Denmark 24–26 August*, 2015.

[33] Emil Hjelseth Thyri. A path-velocity decomposition approach to collision avoidance for autonomous passenger ferries. Master's thesis, Norwegian University of Science and Technology (NTNU), June 2019.

Appendix A

milliAmpere Current and Voltage Filtering

Before using the current and voltage measurements from the milliAmpere thrusters, the signals must be filtered. As Figure A.1 shows, the original measurements suffer from occasional wild points. These points must be detected and removed to give good power estimates. With the batteries delivering a voltage of 24 V and each thruster being capable of delivering 2 kW, the threshold values in Table A.1 was chosen. By removing the values outside of these regions, and replacing them with an interpolated value based on the measurements before and after, the better estimates of Figure A.2 can be obtained.

Name	Threshold Value
V_{max}	30 V
V_{min}	20 V
C_{max}	100 A
C_{min}	-100 A

Table A.1: Threshold values for the current and voltage filtering.

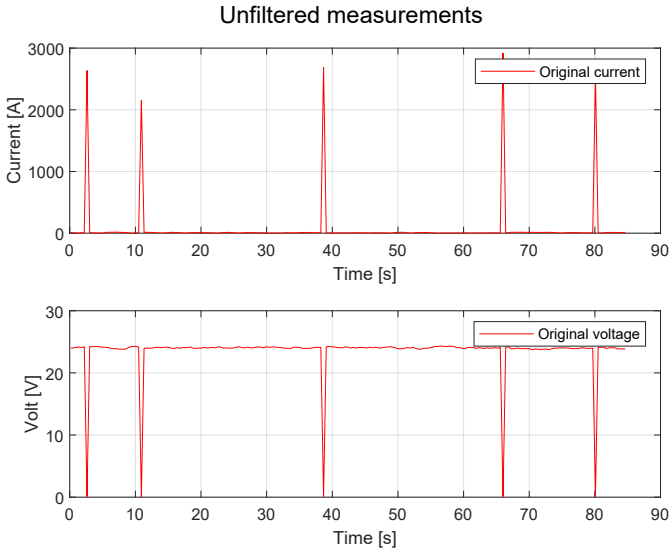


Figure A.1: Example of the original current and voltage measurements from one of the milliAmpere thrusters.

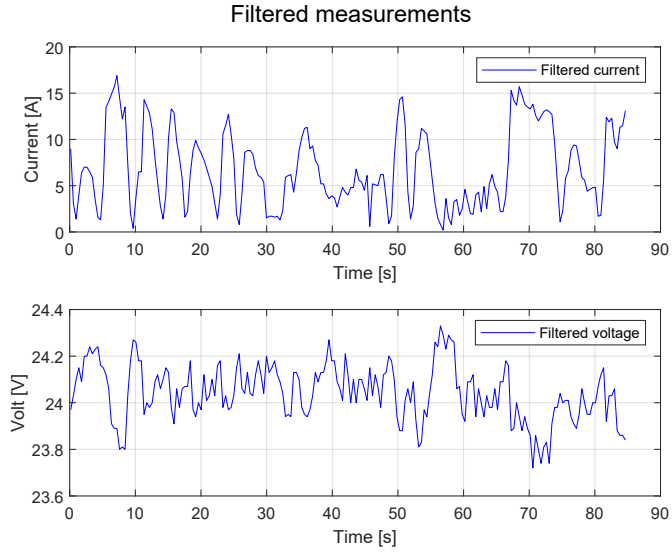


Figure A.2: The current and voltage measurements after filtering

Appendix B

Test Plan for Experiments on milliAmpere

Docking Experiments for the milliAmpere Ferry
TEST PLAN

Eivind Duus Molven

27. April- 1. May
2020

Safety

When operating the milliAmpere vessel at sea, HMS-regulations must be followed. This includes having at least two persons on board, and being aware of the placing of emergency stops, emergency exits and fire extinguisher. In addition, all operators must be taught in advance how to act in case of unexpected events such as loss of propulsion, power loss, fire, crash etc. As the vessel experiments are taking place during the COVID19-lockdown, steps to minimize contact between the vessel personnel must also be taken. These include keeping 1 meters distance to each other and washing contact points with antibac between user sessions. The following procedures must be followed every day:

- Before testing:
  - At least two persons are on board, both wearing life jackets
  - Contact points on the vessel have been washed with antibac
  - Battery status on the ferry and the emergency switches has been checked
  - The weather is found suitable and we are aware of any other activity in the harbour environment
- During testing:
  - We are aware of other vessels and activity in the area.
  - Emergency stops and remote control is always within reach
  - At least 1 m distance is kept between the personnel on board
  - The vessel is maneuvered responsibly with sufficient margin for errors
- After testing:
  - Contact points are washed with antibac
  - The boat is properly tied to shore
  - The vessel is connected to power from shore
  - The main power switch is turned off and locked

Test goal

The main goal for these experiments on milliAmpere is to collect state data from the ferry as it is performing three different docking scenarios with 5 different docking methods.

Handling of test data

In order to ensure that the correct experimental data is obtained throught the test week, the process of collecting, verifying, organizing and storing data must be done properly:

Collection

The control system on board runs ROS, which allows for collection of data in a file called a *rosbag*. Ferry information sent from the different nodes of the system during operation will be logged in this file format and is stored for later processing. For this experiment, the relevant ferry data is:

- Pose $\eta = [x, y, \psi]$
- Velocity $\nu = [u, v, r]$
- DP-controller pose reference $[x_r, y_r, \psi_r]$
- DP-controller velocity reference $[u_r, v_r, r_r]$
- Azimuth propeller speed
- Azimuth motor current
- Azimuth motor voltage

Verification

When a scenario has been performed and a new rosbag is made, this file is immediately transferred on a memory stick to a separate computer on board the vessel. Here, a Matlab script runs the file and plot figures of the desired values. Looking at them manually, the data can be verified as valid (or invalid).

Organization and storage

When a valid data file has been made, it is renamed and stored on the computer. The filename already includes the date and time, and is augmented with the scenario number and the docking method that is used. For scenario 3, method OCP, a valid file name will eg. be: `ma_2020-04-15-08-13-07_OCP_Scen3.bag`

The valid file names are then mode to a folder for the specific scenario on the plotting computer, and a copy is stored on the memory stick for redundancy.

Tentative schedule

Table 1 shows a tentative schedule for the vessel experiments performed at Brattørkaia.

Location

The experiments will take place at Brattørkaia in the city of Trondheim, Norway. A satellite image of this harbour area is found in Figure 1

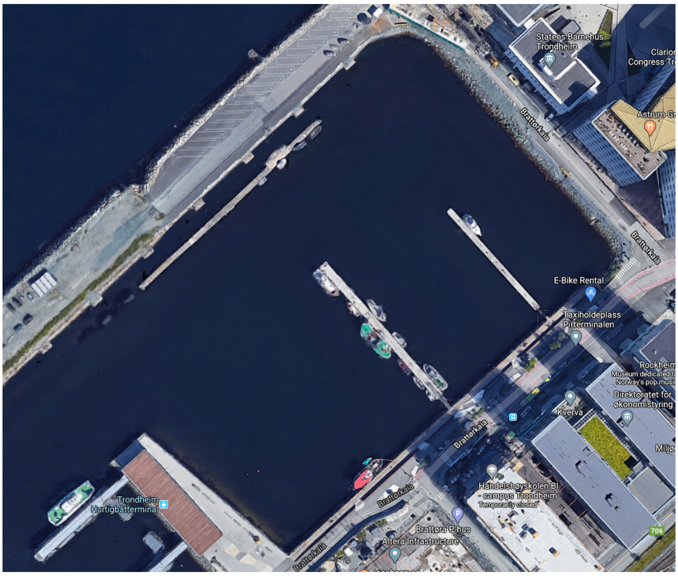


Figure 1: Satellite image of Brattørkaia harbour area

Date	Time	Test plan section number	Task description	Estimated Duration
27. April 2020	09.00-11.30	1.1	Heading experiments and controller tuning	2:30
27. April 2020	12.00-13.00	2.1.1	Scenario 1 with OCP method	1:00
27. April 2020	13.00-14.00	3.1.1	Scenario 2 with OCP method	1:00
27. April 2020	14.00-15.00	4.1.1	Scenario 3 with OCP method	1:00
27. April 2020	15.00-15.30	2.1.3	Scenario 1 with MPC method	0:30
27. April 2020	15.30-16.00	2.1.2	Scenario 1 with DP method	0:30
28. April 2020	09.00-10.30	2.1.4	Scenario 1 with A* method	1:30
28. April 2020	10.30-13.00	2.1.5	Scenario 1 with Operator Control	2:30
28. April 2020	13.00-14.30	3.1.2	Scenario 2 with A* method	1:30
28. April 2020	14.30-16.00	3.1.3	Scenario 2 with Operator Control	1:30
29. April 2020	09.00-10.30	4.1.3	Scenario 3 with A* method	1:30
29. April 2020	10.30-12.00	4.1.2	Scenario 3 with Operator Control	1:30
29. April 2020	12.00-16.00		Available for additional testing	4:00
30. April 2020	09.00-16.00		Available for additional testing	7:00

Table 1: Tentative schedule for the vessel experiments

1 Heading controller testing

To get accurate and reliable data when testing the docking methods, it is vital that the DP-controller found in milliAmpere is working correctly. The ferry is known to be quite unstable in the heading angle, and the heading controller must be tuned precisely to guarantee sufficient heading control. Thus, the first experiment will be to verify that the heading controller is sufficiently stable.

1.1 Experiments

To test the heading controller two experiments will be performed. The first experiment tests the ability to change heading when staying at rest, while the second involves both movement and rotation. **Both** experiments should be tested in **DP** and **Trajectory Planner EDM** mode.

1.1.1 Ex 1: Change of heading at rest

1. Make sure the boat is at rest on the water away from the shore with heading angle at 0 degrees
2. Change the heading in the display to 180 degrees and set waypoint.
3. Pay attention to the boats movement by looking at the reference and actual pose monitor. A succsessful test has the boat converging to the desired heading without overshooting more than 22.5 degrees and the oscillations damping out quickly (<20s)

1.1.2 Ex 2: Change of heading with velocity

1. Make sure the boat is at rest on the water approximately close to the point (40,53). Heading angle should be approximately 45 degrees.
2. Set the new waypoint (55,65,180).
3. Pay attention to the boats movement by looking at the reference and actual pose monitor. A succsessful test has the boat converging to the desired heading without overshooting more than 22.5 degrees and the oscillations damping out quickly (<20s)

2 Scenario 1

The first scenario shall be performed by all five docking methods, and simulates a simple docking maneuver. This scenario serves as a benchmarking experiment as the convex scenario allows for testing of all the methods.

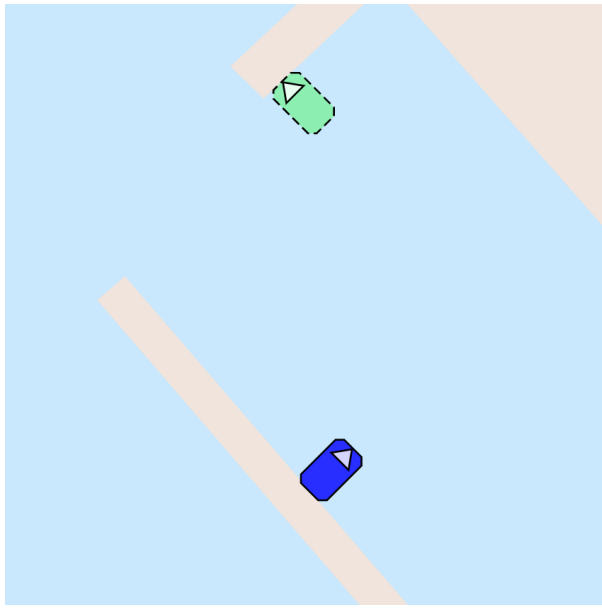


Figure 2: Starting and ending position of scenario 1

2.1 Experiments

For all docking methods, make sure that the correct MAP is used and a new rosbag is created and that the starting position of the boat is approximately around:

$$[x_0, y_0, \psi_0] = [40, 53, \frac{\pi}{4}] \tag{1}$$

Also, make sure that the planned movement space is free of obstacles and hinders.

2.1.1 OCP tp scenario 1 test:

1. Make sure the boat is laying at rest at $[40, 53, \frac{\pi}{4}]$ in the mode: **Trajectory Planner EDM** with settings set for OCP-trajectory planning.
2. Select the waypoint at $[80, 50, -\frac{\pi}{4}]$ and press *Set waypoint*
3. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 90 seconds have passed.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note the time.

2.1.2 DP scenario 1 test:

1. Make sure the boat is laying at rest at $[40, 53, \frac{\pi}{4}]$ in the mode: **DP**
2. Select the waypoint at $[80, 50, -\frac{\pi}{4}]$ and press *Set waypoint*
3. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 90 seconds have passed.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note the time.

2.1.3 MPC scenario 1 test:

1. Make sure the boat is laying at rest at $[40, 53, \frac{\pi}{4}]$ in the mode: **Docking**
2. Select the waypoint at $[80, 50, -\frac{\pi}{4}]$ and press *Set waypoint*
3. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 90 seconds have passed.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note the time.

2.1.4 A* scenario 1 test:

1. Make sure the boat is laying at rest at $[40, 53, \frac{\pi}{4}]$ in the mode: **Trajectory Planner EDM** with settings set for A*-trajectory planning.
2. Select the waypoint at $[80, 50, -\frac{\pi}{4}]$ and press *Set waypoint*
3. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 90 seconds have passed.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note the time.

2.1.5 Operator controlled scenario 1 test:

1. Make sure the boat is laying at rest at $[40, 53, \frac{\pi}{4}]$ in the mode: **Joystick**.
2. Start a timing device and drive towards the waypoint at $[80, 50, -\frac{\pi}{4}$
3. Try to spend approximately 90 seconds on the docking operation and to steer as efficient as possible.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note.
5. Repeat step 1-4 three times in order to get a representative amount of data.

3 Scenario 2

The second scenario and will try to simulate a docking operation where the harbour environment is more complex and obstacles must be avoided. This scenario will be performed by the docking methods:

- OCP tp
- A* tp
- Operator controlled

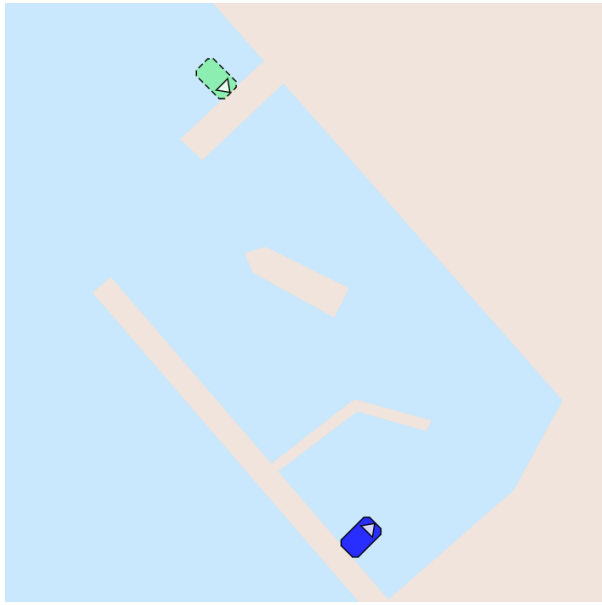


Figure 3: Starting and ending position of scenario 2

3.1 Experiments

For all docking methods, make sure that the correct MAP is used and a new rosbag is created and that the starting position of the boat is approximately around:

$$[x_0, y_0, \psi_0] = [18, 72, \frac{\pi}{4}] \tag{2}$$

Also, make sure that the planned movement space is free of obstacles and hinders.

3.1.1 OCP tp scenario 2 test:

1. Make sure the boat is laying at rest at $[18, 72, \frac{\pi}{4}]$ in the mode: **Trajectory Planner EDM** with settings set for OCP-trajectory planning.
2. Select the waypoint at $[94, 48, \frac{3}{4}\pi]$ and press *Set waypoint*
3. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 250 seconds have passed.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note the time.

3.1.2 A* scenario 2 test:

1. Make sure the boat is laying at rest at $[18, 72, \frac{\pi}{4}]$ in the mode: **Trajectory Planner EDM** with settings set for A*-trajectory planning.
2. Select the waypoint at $[94, 48, \frac{3}{4}\pi]$ and press *Set waypoint*
3. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 250 seconds have passed.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note the time.

3.1.3 Operator controlled scenario 2 test:

1. Make sure the boat is laying at rest at $[18, 72, \frac{\pi}{4}]$ in the mode: **Joystick**.
2. Start a timing device and drive towards the waypoint at $[94, 48, \frac{3}{4}\pi]$. PAY ATTENTION TO THE MAP AND STEER CLEAR OF THE VIRTUAL OBSTACLES
3. Try to spend approximately 250 seconds on the docking operation and to steer as efficient as possible.
4. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note.
5. Repeat step 1-4 three times in order to get a representative amount of data.

4 Scenario 3

The third scenario and will try to simulate a complete docking operation where the boat comes in with transit speed at 1 m/s and the automatic docking method takes over. This scenario will be performed by the docking methods:

- OCP tp
- Operator controlled
- A* tp

Figure 4: Starting and ending position of scenario 3

4.1 Experiments

For all docking methods, make sure that the correct MAP is used and a new rosbag is created and that the starting position of the boat is approximately around:

$$[x_0, y_0, \psi_0] = [-10, -105, \frac{\pi}{4}] \tag{3}$$

Also, make sure that the planned movement space is free of obstacles and hinders.

4.1.1 OCP tp scenario 3 test:

1. Make sure the boat is laying at rest at $[-10, -105, \frac{\pi}{4}]$ in the mode: **Trajectory Planner EDM** with settings set for OCP-trajectory planning and the transit-docking mode enabled.
2. Select the waypoint at $[20, -75, \frac{3}{4}\pi]$ and press *Set waypoint* to set the ending point for the transit phase.

3. As soon as the vessel starts moving, select the waypoint at $[18, 72, \frac{5}{4}\pi]$ as the ending point of the docking operation.

4. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 440 seconds has passed.

5. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note the time.

4.1.2 Operator controlled scenario 3 test:

1. Make sure the boat is laying at rest at $[-10, -105, \frac{\pi}{4}]$ in the mode: **Joystick**.

2. Drive towards the waypoint at $[20, -75, \frac{3}{4}\pi]$. Try to have 1 m/s speed in surge direction and the correct heading angle as the waypoint is passed.

3. Start a timing device and steer towards the waypoint at $[18, 72, \frac{5}{4}\pi]$. PAY ATTENTION TO THE MAP AND STEER CLEAR OF THE VIRTUAL OBSTACLES

4. Try to spend approximately 440 seconds on the docking operation and to steer as efficient as possible.

5. Drive back to the starting point and restart the vessel system. Make sure to store the rosbag file with correct scenario name and note.

6. Repeat step 1-4 two times in order to get a representative amount of data.

4.1.3 A* scenario 3 test:

The A* method does not support a smooth transition from 1m/s to docking, but a similar experiment can still be performed

1. Make sure the boat is laying at rest at $[-10, -105, \frac{\pi}{4}]$ in the mode: **Joystick**.

2. Drive towards the waypoint at $[20, -75, \frac{3}{4}\pi]$. Try to have 1 m/s speed in surge direction and the correct heading angle as the waypoint is passed.

3. Quickly switch to the mode: **Trajectory Planner EDM** with settings set for A*-trajectory planning and the transit-docking mode DISABLED.

4. Select the waypoint at $[18, 72, \frac{5}{4}\pi]$ as the ending point of the docking operation.

5. Monitor the docking operation and wait till the vessel has come to rest at the new waypoint and at least 440 seconds has passed.

6. Quit the vessel operation. Make sure to store the rosbag file with correct scenario name and note the time.

Eivind Duus Molven

Optimal Control-based Docking for Autonomous Ferries

NTNU
Kunnskap for en bedre verden