Autonomous Docking of an Unmanned Surface Vehicle using Model Predictive Control

Sofia Kockum

LUND UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6164
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2022

Abstract

Autonomous docking of marine vessels presents challenges different from the ones faced when travelling at open sea or in an archipelago. External disturbances due to the varying environment and accurate positioning at low speed are examples of these kinds of challenges. The aim of this work has been to develop and implement an autopilot algorithm for docking. This was to be done for a specific marine vessel, using model predictive control (MPC). The vessel in question was Saab Kockums' Piraya, an unmanned surface vehicle (USV).

The problem formulation includes design and implementation of the MPC itself, as well as handling external disturbances and obstacle avoidance. Different software approaches to solving the optimization problem have been explored, as well as different solutions to the matter of advancing towards the desired position. Three approaches were implemented in an MPC framework.

The developed controllers have been successfully tested in simulations, in experiments using a small-scale model car dynamically representing the USV, as well as on the USV at Gränsö, Västervik.

Acknowledgments

I would like to thank the Department of Automatic Control at LTH and Saab Kockums for giving me the opportunity to work with this project and providing me with everything needed. Thanks to WARA-PS (Wallenberg AI, Autonomous Systems and Software Program Research Arena – Public Safety) for support with testing my work on the actual vessel. Thanks also to Jens-Olof Lindh, Saab Kockums, who contributed with extensive input and ideas during the process.

Also thank you to Alexander Persson, who took his time helping me to work with the Linux system. I would also like to express my gratitude to my examiner Karl-Erik Årzén for this opportunity to work with this technology.

At last, a special thank you to my supervisors Anders Robertsson, Birgitta Wingqvist, and Björn Olofsson, who were always quick to support. A special thanks to Anders who introduced me to this project.

Contents

1

Introduction

Docking of a surface vehicle is a challenging task. Not only due to the necessity of high precision when maneuvering close to a dock, but also to the need of taking external disturbances and obstacle avoidance into consideration. The present study addresses the main challenges one is faced by when performing docking of a surface vehicle.

Figure 1.1 illustrates possible problematics of docking a boat. These kinds of situations will not occur with autonomous docking.

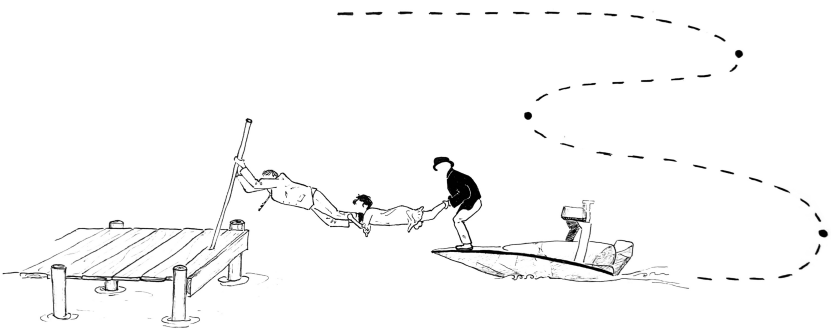

Figure 1.1 *Att angöra en brygga* or *Docking the boat*. At least this situation will not be a problem when considering unmanned vehicles. (Drawing loosely based on and inspired by the movie *Att angöra en brygga* [Danielsson, 1965]. Illustration made by the author.)

1.1 Related Work

Development of an autopilot for a marine vessel is described in [Voigt and Alkaysi, 2020]. The vessel in question is a personal watercraft's. Main focus is to be able to navigate at open sea with the specific vessel's dynamic taken in consideration.

Navigation was done with waypoint following. To perform docking of an autonomous surface vehicle (ASV) using nonlinear MPC, with focus on calculating a safe operation area, is described in [Martinsen et al., 2020]. Regarding the vessel in focus in this work, the Piraya, a previous project has explored autonomous docking [Wallenberg AI, Autonomous Systems and Software Program (WASP), 2021a]. Thanks to this, an efficient interface with sensors of the Piraya and Robot Operating System (ROS) had already been implemented in the USV's onboard computer.

Dynamic models for marine vehicles have been investigated, and they differ from those for ground vehicles as well as aeral vehicles. In [Fossen, 1994], models for marine vehicles are described. Most marine models are of the type greybox models. This means that they partially need experimental data to be fully described, as opposed to whitebox models which can be fully described based on theoretical knowledge, and blackbox models which are exclusively based on experimental data [Duun-Henriksen et al., 2013]. Once a greybox model has been described, it is necessary to identify the specific parameters, and an approach for doing this targeting USVs is described in [Ljungberg, 2020]. The greybox testing and modelling for the Piraya has been performed by Ljungberg, and the Piraya system model is described in [Ljungberg, 2021].

1.2 Problem Formulation

The purpose of this thesis is to explore autonomous docking of the USV Piraya. This includes motion planning and control, as well as taking external disturbances and obstacle avoidance into account. The proposition is that the vessel should be controlled with an MPC. The usage and implementation of such a controller is to be explored. The intention is that this thesis will contribute to future development of docking algorithms for USVs.

To explore docking performance with MPC, the following topics are to be investigated:

- To reach a target position with high accuracy.
- Taking requirements for navigation in harbor environment in consideration.
  - Navigation should be at low speed.
  - Considering avoidance of possible obstacles such as buoys and shallows.
  - Avoidance of the dock, still being able to reach a position close to it.
  - Investigation of what the main external disturbances are and how to take them into consideration.
- Possible procedure(s) to navigate in harbor environment.
- Is MPC a possible solution to control the vessel in question?

1.3 Limitations

The physical setup of the vessel is not included in the problem formulation, and neither is the interface for sending control signals and receiving state data.

Known obstacles should be taken into account, and they are in this case assumed to be static. Unknown or moving objects are not included.

2

Background

Unmanned vessels navigating on sea face different difficulties compared to unmanned land vehicles. The environments and external disturbances are unpredictable and nonlinear, causing various types of challenges. Furthermore, to navigate at open sea, in an archipelago or in a harbor, can cause the seemingly similar environment to vary substantially. The work described in this thesis will focus on the complexity of maneuvering in a harbor and, more specifically, to the situation of docking. This includes challenges of non-linear behavior of the vessel, because of wind, ocean currents, waves, and drifting. It needs to be investigated how the vessel should be controlled and maneuver both due to its specific vessel model, and due to the situational aspects. It is of importance to not collide with the dock, even if it is to be close to it, and reaching a specified position. The USV in question is part of the WARA-PS (Wallenberg AI, Autonomous Systems and Software Program (WASP) Research Arena – Public Safety) [Wallenberg AI, Autonomous Systems and Software Program (WASP), 2021b] [Andersson et al., 2021]. It is mainly used to collect data and to cooperate with other, autonomous robots [Wallenberg AI, Autonomous Systems and Software Program (WASP), 2021a]. It is also an asset at the WASP Research Arena for Public Safety, WARA-PS, a research arena with focus on public safety.

Projects including the Piraya have already been developed, meaning that there are some results possible to base this work on. The system model of the Piraya is based on the established surface vehicle model described in Section 3.2, with its specific parameters being investigated and set by [Ljungberg, 2021]. Since the investigated situation involves docking and maneuvering in harbor area, velocities will be relatively low, which is also the case when Ljungberg found the specific parameters, using velocities from 1–2 m/s.

2.1 Model Predictive Control

Model predictive control, MPC, is a type of process control based on an optimization problem. The general idea of it is to optimize the upcoming states over a finite time

horizon, based on the current state at time instant t. When optimal control signals have been computed, the first sample is applied. After this, at time instant $t + T_s$, where T_s is the sample period, the optimization problem is solved anew [Åström and Bernhardsson, 2016].

The optimization problem is based on the model of the process in question. Based on the model, the behavior of the process can be predicted. The optimization problem is solved with regard to a set of variables, defined through the process model. A set of constraints also needs to be defined together with the process model. The cost function is to be optimized subject to these constraints.

The optimization problem is solved at each time instant, for a number of samples ahead. The number of samples ahead for which the problem is optimized, is called the prediction horizon. The prediction horizon is a finite number of time slots, or seconds. Its length can vary depending on the dynamics of the process, e.g., how far ahead it would be feasible to predict. Since it is computationally expensive, it is not desirable to have a prediction horizon longer than necessary.

The control horizon is defined as a subset of the prediction horizon. During the control horizon, the control signals are allowed to vary, and are in a corresponding manner a variable in the cost function. Control signals are calculated for the control horizon but usually only the first values are used as input to the process. In this thesis, the control horizon and the prediction horizon will be of the same length, as the control signal is allowed to be changed at every time instant over the prediction horizon.

A block diagram for a typical usage of MPC can be seen in Figure 2.1.

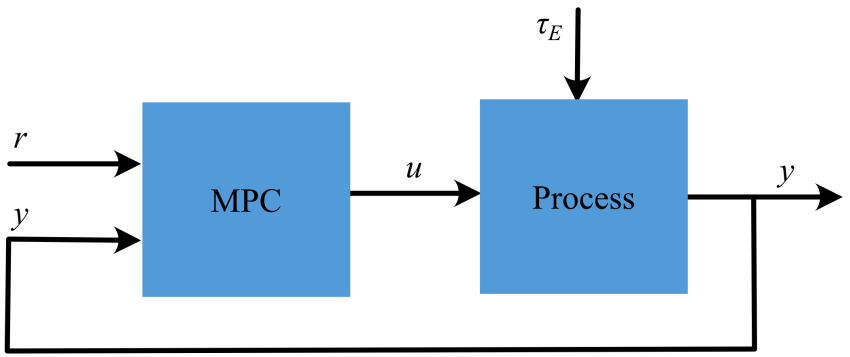

Figure 2.1 Schematic block diagram of a typical MPC usage. Control signals are denoted by u, τ_E is an external disturbance, r the reference value (often a reference trajectory), and y the state.

Model predictive control is favorable when systems of several variables are to be controlled. The constraints are respected, but the process can still be run close to

them, which is favorable for efficiency [Åström and Bernhardsson, 2016]. Its main advantage is the prediction of future dynamics and adjusted control signals at each time slot, based on the system model, cost function and constraints.

A typical, general optimization problem for an MPC could look like (2.1). Here, the cost function $J(x, u, x_{\text{ref}})$ is to be minimized with regard to variables x and u, subject to the model of the process, initial conditions where the first value of the prediction horizon, x_0, is equal to x_{init}, and the constraints for the variables, $\mathcal{X}$ and $\mathcal{U}$. The difference between state x_i and the reference value $x_{\text{ref},i}$ for each sample is penalized with a weight matrix Q. Changes in control signals, $\Delta u_j = u_j - u_{j-1}$, are penalized with weight matrix R, to avoid oscillation. The cost function is summed over the prediction horizon N.

$$\begin{aligned} \underset{x,u}{\text{minimize}} \quad & J(x, u, x_{\text{ref}}) = \sum_{i=0}^{N+1} \left\| x_i - x_{\text{ref},i} \right\|_Q^2 + \sum_{i=0}^{N} \| \Delta u_i \|_R^2 \\ \text{subject to} \quad & x_{i+1} = f(x_i, u) \quad \forall i \in 0 \ldots, N \\ & x_0 = x_{\text{init}} \\ & x \in \mathcal{X} \\ & u \in \mathcal{U} \end{aligned} \tag{2.1}$$

2.2 The Piraya

The vessel used is the Piraya of Saab Kockums, and can be seen in Figure 2.2. It has a length of 4 m and a width of 1.4 m. The motor is an outboard motor and is placed in the back of the vessel. The Piraya is equipped with a number of sensors. For positioning, Onboard INS (Inertial Navigation System) and GPS (Global Positioning System) without RTK (Real Time Kinematic) were used. This provided position, heading, forward velocity, and heading rate.

The vessel is controlled through throttle and angle of the throttle, in this thesis called rudder position. These can be controlled in a ROS environment.

Figure 2.2 The Piraya vessel of Saab Kockums, with a length of 4 m.

3

Modelling

To describe the dynamics of the vessel, a system model is necessary. This will be described in the following, as well as the two frames necessary to describe the vessel's state.

3.1 Coordinate Systems

Two coordinate frames are of interest for the model setup: the global frame and the local frame of the vessel. The global one, indexed n, is the NED-frame (north, east, down). This frame is considered a tangent to the Earth and is rotating with the Earth's axis. Since the vessel in this work will move within an area small enough for the Earth's curvature to have negligible impact, and the vessel will follow the Earth's rotation, neither will have a considerable impact. This global frame is principally used for tracking of the vessel's position. The position in global coordinates will in the following sections and chapters of this thesis be notated without the n-index.

The vessel's local frame, with index b, is body-fixed and useful for the control of the vessel. It enables tracking of forward and sideways velocities, as well as rotational velocities. The six degrees of freedom for the vessel's local frame are described in Table 3.1, where the same notation as in [Fossen, 1994] is used. It may be noted that the nomenclature for velocity in the z-direction is heave when considering surface vehicles. The heave usually concerns offset in z-direction due to waves. The forces and moments will be seen in the system equations of the vessel, in Section 3.2. The two frames in relation to each other can be viewed in Figure 3.1.

Table 3.1 The six DOF of the vessel in the body-fixed frame.

DOF	Axis and Euler angle	Velocity	Motion	Forces and Moments
1	x_b	u	surge	X
2	y_b	v	sway	Y
3	z_b	w	heave	Z
4	ϕ	p	roll	K
5	θ	q	pitch	M
6	ψ	r	yaw	N

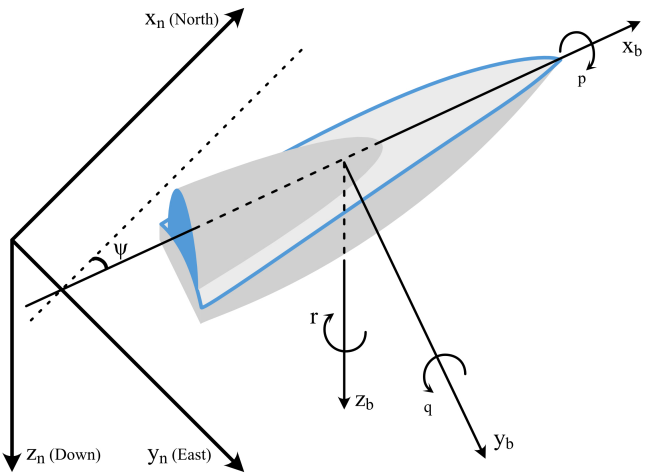

Figure 3.1 Coordinate systems NED, with index n, and local frame of the boat, with index b.

3.2 Model of the Ship

The ship model is based on the nonlinear model described in [Fossen, 1994]. It is expressed as described in Equation (3.1). The components of this equation will be further explained in the following.

$$\left.\begin{aligned} \dot{\tilde{\boldsymbol{\eta}}} &= \tilde{\mathbf{R}}(\tilde{\boldsymbol{\eta}})\tilde{\boldsymbol{\nu}} \\ \tilde{\mathbf{M}}\dot{\tilde{\boldsymbol{\nu}}} + \tilde{\mathbf{C}}(\tilde{\boldsymbol{\nu}})\tilde{\boldsymbol{\nu}} + \tilde{\mathbf{D}}(\tilde{\boldsymbol{\nu}})\tilde{\boldsymbol{\nu}} + \tilde{\mathbf{g}}(\tilde{\boldsymbol{\eta}}) &= \tilde{\boldsymbol{\tau}} + \tilde{\boldsymbol{\tau}}_E \end{aligned}\right\} \tag{3.1}$$

The pose vector, $\tilde{\boldsymbol{\eta}}$ contains the coordinates in x-, y- and z-directions as well as the angle rotations around the mentioned axes. The velocity vector $\tilde{\boldsymbol{\nu}}$ contains the corresponding linear and angular velocities. This model is of 6 DOF (indicated by tilde), which would be accurate for, e.g., an underwater vessel. As the vessel for this project is a surface vessel, the number of DOFs is reduced to 3 (variables now

indicated without tilde). It is assumed that the offset along z-direction due to waves and that the roll and pitch are negligible in the case of position tracking. Hence, in the following equations of the model, the pose vector is reduced to position in x, y, and yaw ψ, and velocity vector to the corresponding velocities:

$$\boldsymbol{\eta} = \begin{bmatrix} x \\ y \\ \psi \end{bmatrix}, \quad \boldsymbol{\nu} = \begin{bmatrix} u \\ v \\ r \end{bmatrix}. \tag{3.2}$$

The transformation matrix $\mathbf{R}(\boldsymbol{\eta})$ in 3 DOFs will have the simplified appearance as shown in (3.3). This matrix transforms from the vessel's local, body-fixed coordinate system to the global, NED coordinate system.

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

The matrix $\mathbf{M}$ is divided into two parts, in accordance with [Fossen, 1994], which are described below. The $\mathbf{M}_{RB}$ is the rigid body inertia matrix, dealing with forces due to moving the rigid body, and $\mathbf{M}_A$ is the added inertia matrix, dealing with forces and moments due to hydrodynamics [Ljungberg, 2020]. For notation of the hydrodynamic derivatives, $X_{\dot{u}}$, $Y_{\dot{v}}$, $Y_{\dot{r}}$, $N_{\dot{r}}$, in this section, the same notation as in [Fossen, 1994] and [Ljungberg, 2020] is used. As an example for explanation retrieved from [Ljungberg, 2020], X is the hydrodynamic added mass force in the x_b-direction, due to an acceleration $\dot{u}$ along the x_b axis: $X = X_{\dot{u}}\dot{u}$, where $X_{\dot{u}} = \partial X / \partial \dot{u}$.

$$\mathbf{M} = \mathbf{M}_{RB} + \mathbf{M}_A = \begin{bmatrix} m & 0 & 0 \\ 0 & m & mx_G \\ 0 & mx_G & I_z \end{bmatrix} + \begin{bmatrix} X_{\dot{u}} & 0 & 0 \\ 0 & Y_{\dot{v}} & Y_{\dot{r}} \\ 0 & Y_{\dot{r}} & N_{\dot{r}} \end{bmatrix} \tag{3.4}$$

Here, m is the vessel's mass, and I_z is the inertia about the boat's local z axis. The variable x_G is the distance from center of mass to the origin of the coordinate system of the boat. These will coincide, hence simplifying the rigid body inertia matrix to

$$\mathbf{M}_{RB} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I_z \end{bmatrix}. \tag{3.5}$$

The C-matrix describes Coriolis and centripetal forces which arise as the local coordinate system of the vessel moves and rotates with respect to the fixed, global coordinate system [Ljungberg, 2020]. In the same manner as the $\mathbf{M}$-matrix, this is divided into a matrix of rigid body kinematics and hydrodynamics:

$$\mathbf{C}(\boldsymbol{\nu}) = \mathbf{C}_{RB}(\boldsymbol{\nu}) + \mathbf{C}_A(\boldsymbol{\nu}) \tag{3.6}$$

The Coriolis matrices are described in accordance to [Ljungberg, 2020], and simplified in accordance to Equation (3.7a) due to the same assumption that x_G is negligible.

$$\mathbf{C}_{RB}(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -m(x_G r + v) \\ 0 & 0 & mu \\ m(x_G r + v) & -mu & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -mv \\ 0 & 0 & mu \\ mv & -mu & 0 \end{bmatrix} \tag{3.7a}$$

$$\mathbf{C}_A(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & -(Y_{\dot{v}} v + Y_{\dot{r}} r) \\ 0 & 0 & X_{\dot{u}} u \\ Y_{\dot{v}} v + Y_{\dot{r}} r & -X_{\dot{u}} & 0 \end{bmatrix} \tag{3.7b}$$

The matrix $\mathbf{D}(\boldsymbol{\nu})$ is the hydrodynamic damping and is, as suggested by [Fossen, 1994], expressed as in (3.8), with the linearized damping moments and forces. The surge and steering mode can be decoupled when considering a slender vessel at low speed, which is the case here.

$$\mathbf{D}(\boldsymbol{\nu}) = -\begin{bmatrix} X_u & 0 & 0 \\ 0 & Y_v & Y_r \\ 0 & N_v & N_r \end{bmatrix} \tag{3.8}$$

In Equation (3.1), the term $\mathbf{g}(\boldsymbol{\eta})$ denotes static forces, inter alia due to gravitational forces. In [Ljungberg, 2021], these are assumed to be zero.

The external disturbances τ_E are, in this case, the three main ones of wind, waves and currents:

$$\boldsymbol{\tau}_E = \tau_{\text{wind}} + \tau_{\text{waves}} + \tau_{\text{current}} \tag{3.9}$$

The vector $\mathbf{u} = [u_t \quad u_r]^T$ is the control signals, from thruster (u_t) and rudder (u_r). Note that bold $\mathbf{u}$ is the vector containing these control signals, and should not be confused with surge, u. The vector $\boldsymbol{\tau}$ (see Equation 3.10) is the impact on the vessel from the control signals, i.e., the propulsion forces and moments [Fossen, 1994] [Ljungberg, 2021]. Since the Piraya only has one, x-axis centered outboard motor, the actuator forces can be modelled relatively simply. As in [Ljungberg, 2021], it is assumed that the dimension and direction of the force from the motor will be proportional to the number of revolutions and the angle of it. If the motor is not mounted in the middle of the vessel's rotation center there will be a moment. For the considered vessel, the motor is mounted in the middle in the back of the boat, hence the displacement along the y-axis is zero but nonzero along the x-axis. This gives the actuator forces along x-, y-axis, and about ψ as in (3.10). Here, the parameters τ_i are modelling the actuator forces and Δ_x is the offset along the x-axis of where the

motor is mounted and the vessel's center of rotation.

$$\left.\begin{aligned} \boldsymbol{\tau} &= \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_\psi \end{bmatrix} \\ \tau_x(t) &= u_t(t)\cos(au_r(t)) \\ \tau_y(t) &= u_t(t)\sin(au_r(t)) \\ \tau_\psi(t) &= \Delta_x u_t(t)\sin(au_r(t)) \propto \tau_y(t) \end{aligned}\right\} \tag{3.10}$$

The control signals u_t and u_r are integers in the intervals $[0, 100]$ and $[-100, 100]$, respectively. For throttle, a signal of $u_t = 0$ corresponded to the motor being at rest, motionless, while $u_t = 100$ corresponded to maximal surge. For rudder, since the motor can be rotated $\pm\pi/6$ rad, the scaling coefficient $a = \pi/600$, in (3.10). Note that this gives us the definition set $\mathcal{U}$ for the control signals $\mathbf{u}$; $\mathbf{u} \in \mathcal{U}$.

This gives us the simplified model of the Piraya vessel, in 3 DOFs:

$$\left.\begin{aligned} \dot{\boldsymbol{\eta}} &= \mathbf{R}(\psi)\boldsymbol{\nu} \\ \mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} &= \boldsymbol{\tau} + \boldsymbol{\tau}_E \end{aligned}\right\} \tag{3.11}$$

In [Ljungberg, 2021] the model described in (3.11) is discretized using an Euler explicit method. The result is

$$\begin{bmatrix} u(t+T_s) \\ v(t+T_s) \\ r(t+T_s) \end{bmatrix} = \begin{bmatrix} u(t) \\ v(t) \\ r(t) \end{bmatrix} + \begin{bmatrix} \theta_1 u(t) & \theta_2 v(t) r(t) & \theta_3 \tau_x(t) \\ \theta_4 v(t) & \theta_5 u(t) r(t) & \theta_6 \tau_y(t) \\ \theta_7 r(t) & \theta_8 u(t) v(t) & \theta_9 \tau_y(t) \end{bmatrix} \tag{3.12}$$

which, in a shorter form will be written as

$$\boldsymbol{\nu}(t+T_s) = \boldsymbol{\nu}(t) + \boldsymbol{\Psi}(t) \tag{3.13}$$

in this thesis. The parameter $T_s = 0.2$ s is the time steps at which the data were logged when the parameters $\boldsymbol{\theta}$ were determined. The size of Δ_x is determined together with the other parameters, hence it is part of $\boldsymbol{\theta}$. It could be worth to notice, that only linear damping was taken into consideration, as it was assumed to be dominant at low speeds. In the same manner, the wake was neglected, as it has a negligible effect at low speeds.

Ljungberg determined and validated the parameters of $\boldsymbol{\theta}$, and they are stated in [Ljungberg, 2021] to be

$$\begin{aligned} \boldsymbol{\theta} &= [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4 \quad \theta_5 \quad \theta_6 \quad \theta_7 \quad \theta_8 \quad \theta_9] \\ &= [-0.0478 \quad 0.1259 \quad 0.0166 \quad -0.2375 \quad 0.1526 \\ &\qquad \ldots \quad 0.0737 \quad -0.3191 \quad 0.0031 \quad -0.0345]. \end{aligned} \tag{3.14}$$

Since the parameters were verified for velocities up to 2 m/s, the definition set for surge is set to $0 \leq u \leq 2$, called $\mathcal{V}$. If velocity is negative, i.e., the vessel is reversing, it will be due to external disturbances since the allowed control-signal interval will only cause forward motion.

3.3 Sway and Yaw Rate

In the model (3.12), it can be seen that the sway and yaw rate are related through the τ_y term. This means that when setting the rudder to turn one way, the vessel will also get a sideways velocity in the counter direction. This means that certain maneuvers may be difficult or not possible to accomplish. For instance, if the vessel has a dock or other type of obstacle close to port, then a control signal to turn starboard make the vessel initially sway to port and possibly cause a collision. This correlation can be distinctively seen in the simulations (Chapter 6), as well as noticed in some of the experiments (Chapter 7).

3.4 Disturbances

There are several difficulties to modelling waves and currents in a harbor environment, since these are specific due to the set of the specific area. Out of the three main disturbances, wind would be the easiest one to measure, if access to an anemometer is possible.

To model wind disturbances in simulations, one can choose to do it in several ways. In [Naukowe and Miller, 2014], a simple model is established in such a way that a relative wind differs from 0 to 1 m/s in velocity, and with an angle differing between 0° to 40°.

In [Fossen, 1994], another approach is discussed, which is a more complex model. Wind is modelled in terms of a mean speed of the wind combined with smaller variations simulating gusting.

In [Yang et al., 2013], the disturbances of wind, waves and current are all lumped together into a vector, giving time-varying environmental disturbances.

Yet another approach could be to model the drag force of wind in respect to the drag coefficient and the vessel's reference area at each time instant. The drag force F_d would be defined as in Equation (3.15) [Wikimedia Foundation, Inc., 2022], where ρ is the density of the wind, u its relative velocity to the vessel, A the reference area, and c_d the drag coefficient,

$$F_d = \frac{1}{2}\rho u^2 c_d A. \tag{3.15}$$

To only model wind could be sufficient in simulations, but when running experiments with the boat it might not be sufficient to assume an offset is only due to wind. To be able to compensate for all kinds of disturbances, they could be lumped together in

such a manner as to measure the offset per second, when all actuator forces are zero. This would give a vector of offset velocities in global coordinates, along the x- and y-directions, as well as rotational. The offset vector is multiplied with the inverse of the rotation matrix, $\mathbf{R}(\psi)^{-1}$, to obtain the corresponding offset in the body-fixed frame. If these are added to the velocities in the model as shown in (3.12), this would give an approximation of all three main disturbances. This last type of disturbance modelling was implemented into the simulations.

Given a measured offset in the global frame of $u_{n,d}$, $v_{n,d}$, and $r_{n,d}$ in x- and y-direction for a time period of T_s, the system equation of (3.12) could be expressed as

$$\left.\begin{aligned} \begin{bmatrix} u_d \\ v_d \\ r_d \end{bmatrix} &= \mathbf{R}(\psi)^{-1} \begin{bmatrix} u_{n,d} \\ v_{n,d} \\ r_{n,d} \end{bmatrix} \\ \begin{bmatrix} u(t+T_s) \\ v(t+T_s) \\ r(t+T_s) \end{bmatrix} &= \begin{bmatrix} u(t) \\ v(t) \\ r(t) \end{bmatrix} + \begin{bmatrix} \theta_1 u(t) & \theta_2 v(t) r(t) & \theta_3 \tau_x(t) \\ \theta_4 v(t) & \theta_5 u(t) r(t) & \theta_6 \tau_y(t) \\ \theta_7 r(t) & \theta_8 u(t) v(t) & \theta_9 \tau_y(t) \end{bmatrix} + \begin{bmatrix} u_d \\ v_d \\ r_d \end{bmatrix} \end{aligned}\right\}. \tag{3.16}$$

4

Motion Planning and Control

For the task of finding the correct position by a dock and navigating in harbor environment, three main approaches have been developed and implemented. The first approach focuses on one target position. The second approach has the implementation of reaching waypoints on its way to the target position. The third approach is to follow an already logged trajectory.

For these three approaches, an additional application of obstacle avoidance was developed. The function of this can be added on to any of the three approaches.

It is assumed that coordinates and eventual heading of target position, waypoints, dock, or obstacles are known and given in the controller. These could for example be known from a map or correspondingly.

Differences in the cost functions and constraints of the approaches will be described in the following section. However, they are all based on the general optimization problem

$$
\begin{aligned}
\underset{\boldsymbol{\eta},\boldsymbol{\nu},\mathbf{u}}{\text{minimize}} \quad & J_{\text{spec}}(\boldsymbol{\eta}, \boldsymbol{\nu}, \boldsymbol{\eta}_{\text{ref}}, \boldsymbol{\nu}_{\text{ref}}, \mathbf{u}) \\
\text{subject to} \quad & \left.\begin{aligned} \boldsymbol{\nu}_{i+1} &= \boldsymbol{\nu}_i + \boldsymbol{\Psi}_i \\ \boldsymbol{\eta}_{i+1} &= \boldsymbol{\eta}_i + T_s \mathbf{R}(\psi_i)\boldsymbol{\nu}_{i+1} \end{aligned}\right\} \quad \forall i \in 0 \ldots, N \\
& \boldsymbol{\eta}_0 = \boldsymbol{\eta}_{\text{init}}, \quad \boldsymbol{\nu}_0 = \boldsymbol{\nu}_{\text{init}} \\
& \mathbf{u} \in \mathcal{U} \\
& \boldsymbol{\eta} \in \mathcal{X} \\
& \boldsymbol{\nu} \in \mathcal{V},
\end{aligned}
\tag{4.1}
$$

where J_{spec} is the cost function to be specified for the approach, $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ are as described in (3.2), $\boldsymbol{\eta}_{\text{ref}}$ and $\boldsymbol{\nu}_{\text{ref}}$ their corresponding reference values which also are specified for the approach, $\mathbf{u}$ is the control signals, and T_s the sampling period. The constraints are given by the model dynamics, initial conditions with initial values $\boldsymbol{\eta}_{\text{init}}$ and $\boldsymbol{\nu}_{\text{init}}$, and the variable constraints $\mathcal{X}$, $\mathcal{U}$, and $\mathcal{V}$. In (3.13), $\boldsymbol{\Psi}_i$ is defined. If

disturbances are added to this, $\mathbf{\Psi}_i$ will have an additional term of $[u_d \quad v_d \quad r_d]^T$, as defined in (3.16).

In all the cost functions specified for the approaches, the weight matrix R will penalize usage of control signals and hence, in extension, energy. Based on empirical testing, R is defined as

$$R = \text{diag}(0.001, 0.01). \tag{4.2}$$

A matrix I will penalize change of control signals, to avoid oscillation. Matrix I is the identity matrix.

4.1 One Target Position

For the first approach, a single target position is given, which includes x- and y-coordinates as well as the heading. The path to the target position is not specified. This means the path is planned purely based on the MPC, with cost function defined as

$$\begin{aligned} J_{\text{endpoint}}(\boldsymbol{\eta}, \boldsymbol{\nu}, \boldsymbol{\eta}_{\text{ref}}, \boldsymbol{\nu}_{\text{ref}}, \mathbf{u}) = \left\|\boldsymbol{\eta}_0 - \boldsymbol{\eta}_{\text{ref},0}\right\|_{Q_1}^2 + q_\psi c_\psi(\boldsymbol{\eta}_0) + \|\boldsymbol{\nu}_0\|_{Q_2}^2 \\ + \sum_{j=0}^{N} \Big(\left\|\boldsymbol{\eta}_{j+1} - \boldsymbol{\eta}_{\text{ref},j+1}\right\|_{Q_1}^2 + q_\psi c_\psi(\boldsymbol{\eta}_j) + \left\|\boldsymbol{\nu}_{j+1} - \boldsymbol{\nu}_{\text{ref},j+1}\right\|_{Q_2}^2 \\ + \left\|\mathbf{u}_j - \mathbf{u}_{j-1}\right\|_I^2 + \left\|\mathbf{u}_j\right\|_R^2 \Big). \end{aligned} \tag{4.3}$$

Definition of the weights are given in the following.

When only one target position is given, all reference values will be the target position. This means that all optimized path positions over the prediction horizon will strive to that one position. Hence, the cost function (4.3) is penalizing when being far away from the target position, with quadratic cost and weight matrix Q_1. Velocities are also penalized with quadratic cost and weight matrix Q_2.

While the optimal path is being calculated, that is, while the desired target position is not yet within the prediction horizon, distance the target coordinates $(x_{\text{end}}, y_{\text{end}})$ will be highly penalized. However, error in the heading, w.r.t. the heading at the target position, should not be as heavily penalized while being far away from the goal but should increase as the target position is being approached and coming closer. Neither should velocities strive to be zero, as the reference velocities $\boldsymbol{\nu}_{\text{ref}}$ for end state were set to zero, while approaching the goal state. Hence, the weights Q_1 and Q_2 while aiming for the desired target position, are chosen as

$$Q_1 = \text{diag}(100, 100, 0), \qquad Q_2 = \text{diag}(0, 0, 0). \tag{4.4}$$

As the target position is being approached, it becomes more important to reach the exact goal state. As soon as the desired target position is within the prediction horizon,

the weight matrices will be changed such that there will be more weight on the x- and y-position coordinates and on the velocities. When the last optimized position over the prediction horizon is within a certain tolerance of the target position, the state optimization will be more exact with the weight matrices as

$$Q_1 = \text{diag}(1000, 1000, 0), \qquad Q_2 = \text{diag}(100, 100, 100). \tag{4.5}$$

For the heading, on the other hand, the cost is defined as suggested in [Martinsen et al., 2020], with heading cost function $c_\psi(\boldsymbol{\eta}_j)$. The defined cost of the heading consists of one part of the heading error, and one part which is a Gaussian function and reduces the penalization when the goal position is far away. The heading error cost can be seen in (4.6), where $q_\psi = 20$ and $\delta = 10$.

$$q_\psi c_\psi(\boldsymbol{\eta}_j) = q_\psi \frac{1}{2}(1 - \cos(\psi_j - \psi_{\text{ref}})) \exp\left(\frac{(x_j - x_{\text{ref}})^2 + (y_j - y_{\text{ref}})^2}{2\delta^2}\right) \tag{4.6}$$

This method will generate a rather straight path. Depending on the target heading and from which direction the vessel is approaching, only having one specified desired target point could be sufficient. However, in, for example, a situation when the vessel is travelling from the west towards a target position at east with target heading north, it could be desirable to travel a smoother path and reach the target position from the south. Especially if there is a dock, or similar, to the east of the target position. This is illustrated in Figure 4.1.

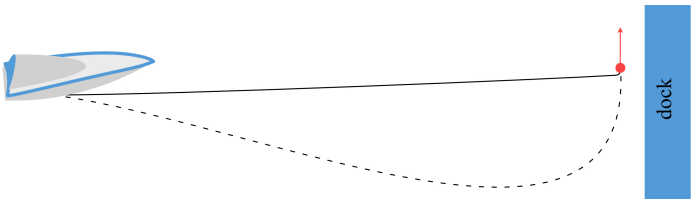

Figure 4.1 Illustration of when it might not be sufficient to only specify a target position, where the vessel will take the shortest route (full line) and precision of end heading might be suffering. For precision, to take the dashed route could be more efficient, giving a smoother path.

4.2 Waypoint Following

The second approach is the waypoint approach. This enables another path than the most straight forward, with the possibility to add points to reach in order on the way towards the target position. The number of waypoints can differ from one up to any number. In the scenario described previously, when reaching a target position

from the west with end heading to the north, it would with this waypoint solution be possible to add a waypoint south of the target position. The vessel would then have a smoother approach, first going south of the target position and reaching it with the correct heading. The waypoints are reached in order, having only one waypoint at a time as reference point. When the vessel is within a specific radius of the waypoint, the reference value will switch to the next waypoint. This makes it possible to keep a smooth path. Simulations of the vessel in question were made, where the vessel's surge u was kept around 2 m/s, and full rudder signal was given. This showed that the model had a turning radius of 12 m. The waypoint radius in the simulations is hence 5 m, to make it reach the point close enough but still have a relatively smooth path. This is with the assumption that the desired route is somewhat forward and not to make a U-turn. If this is desired, the waypoint radius should be considered to be changed to approximately 12 m. A schematic of the waypoint following can be seen in Figures 4.2 and 4.3. In these figures, it is possible to clearly see how the vessel first gets a side velocity to port when it turns rudder to go to starboard, as mentioned in Section 3.3. The desired waypoints are given as a vector with coordinates of each

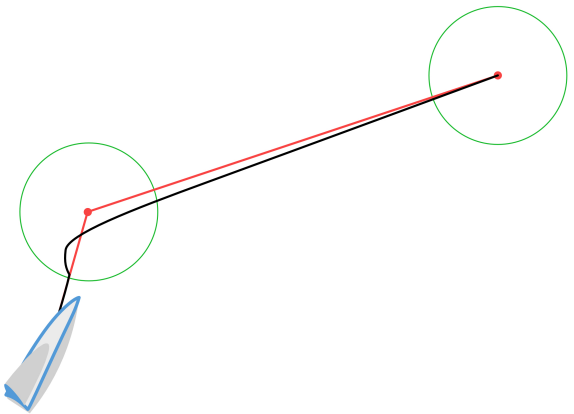

Figure 4.2 Waypoint following. Green circles are of radius corresponding to when the reference point is set to the next waypoint. Red lines indicate the reference path if the vessel were to reach each point precisely and then turn in place. Black line indicates the actual, travelled path.

point, in the order in which they are to be reached. The waypoints can be defined through, for example, sea maps as in [Eniro Sverige AB, 2022]. Hence, it is assumed that no reference direction will be given, but only x- and y-coordinates in the global frame. The reference value for heading will be set to zero at all the given coordinates, but will have zero weight in the weight matrix Q_1, as in (4.4), such that only reaching

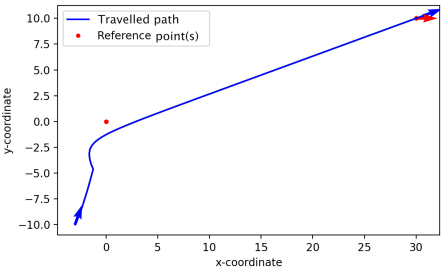

Figure 4.3 Reference image from simulations, as reference for the drawing in Figure 4.2.

the desired waypoints is prioritized. The cost function is then chosen as

$$J_{\text{waypoints}}(\boldsymbol{\eta}, \boldsymbol{\nu}, \boldsymbol{\eta}_{\text{ref}}, \boldsymbol{\nu}_{\text{ref}}, \mathbf{u}) = \left\|\boldsymbol{\eta}_0 - \boldsymbol{\eta}_{\text{ref},0}\right\|_{Q_1}^2 + \|\boldsymbol{\nu}_0\|_{Q_2}^2 + \sum_{j=0}^{N}\Big(\left\|\boldsymbol{\eta}_{j+1} - \boldsymbol{\eta}_{\text{ref},j+1}\right\|_{Q_1}^2 + \left\|\boldsymbol{\nu}_{j+1} - \boldsymbol{\nu}_{\text{ref},j+1}\right\|_{Q_2}^2 + \left\|\mathbf{u}_j - \mathbf{u}_{j-1}\right\|_I^2 + \left\|\mathbf{u}_j\right\|_R^2\Big). \quad (4.7)$$

Notice that the function for heading error, (4.6), is not used in this case, since no weight is to be put on heading positioning while following the waypoints. The only given, desired heading will be defined for the desired target position, after all waypoints have been reached. In the same manner as for the approach of only having one target position, when the target position is within the prediction horizon, the weight matrices will be changed to

$$Q_1 = \text{diag}(1000, 1000, 1000), \qquad Q_2 = \text{diag}(100, 100, 100). \quad (4.8)$$

It is assumed that the waypoints are given in the desired reaching order in the given vector. No matter the vessel's initial state, the MPC will begin with optimizing the trajectory to reach the first waypoint, and thereafter continue with the second waypoint of the vector, and so on. Like in the case of optimizing for a single target position, all optimized states over the prediction horizon will strive to only one point as reference value, e.g., the current waypoint. This means that η_{ref} is constant over the prediciton horizon.

4.3 Logged Trajectory

The third approach is to follow a trajectory logged by an earlier simulation or logged when it was travelled by the real vessel. The advantage of this is that it is known

to be a feasible route, under the current circumstances. It could also be a trajectory computed by a higher level path planner. Unlike the waypoint following, these points will be connected to a certain speed. It is of importance that the data are logged with the same sampling period as the model is simulated with, to reach the same speed. If for example the reference trajectory is logged with time steps $T_s = 0.2$ s at a velocity of 1 m/s and the model follows it with time step $T_s = 0.5$ s, it will be travelling with a velocity of 0.4 m/s. Unless this is a desired behavior, the trajectory could be resampled.

Unlike the two approaches previously mentioned, this does not have a target point and hence only has the weight matrices described in (4.4). The cost function is defined as

$$\begin{aligned} J_{\text{loggedTraj}}(\boldsymbol{\eta}, \boldsymbol{\nu}, \boldsymbol{\eta}_{\text{ref}}, \boldsymbol{\nu}_{\text{ref}}, \mathbf{u}) = \left\|\boldsymbol{\eta}_0 - \boldsymbol{\eta}_{\text{ref},0}\right\|_{Q_1}^2 + \|\boldsymbol{\nu}_0\|_{Q_2}^2 \\ + \sum_{j=0}^{N} \Big(\left\|\boldsymbol{\eta}_{j+1} - \boldsymbol{\eta}_{\text{ref},j+1}\right\|_{Q_1}^2 + \left\|\boldsymbol{\nu}_{j+1} - \boldsymbol{\nu}_{\text{ref},j+1}\right\|_{Q_2}^2 \\ + \left\|\mathbf{u}_j - \mathbf{u}_{j-1}\right\|_I^2 + \left\|\mathbf{u}_j\right\|_R^2 \Big). \end{aligned} \tag{4.9}$$

4.4 Obstacle Avoidance

For obstacle avoidance, two areas of use have been investigated. The first is avoidance of a point obstacle, for example a rock. It is possible to add a radius to this obstacle, for which it is not allowed to travel within the corresponding circle. In addition, it is possible to add a "safety radius" with a soft constraint in the MPC, using a slack variable. The slack variable is a part of the cost function in the MPC, where it is optimized for each position. The slack variable is multiplied with a high figure in the cost function, such that if it needs to have a value greater than zero, there will be a high cost, and so it will only be greater than zero if it is unavoidable. The second scenario implemented is the dock at Gränsö. The goal is to get close to the dock but not to touch it. Therefore, the coordinates of the dock are introduced as a hard constraint. This is implemented with regard to the Euclidean distance between the vessel's position and the position of the dock, plus the vessel's dimensions and a safety margin of choice. The coordinates of the dock and the potential obstacle leave us with the allowed set of values for the state vector, $\boldsymbol{\eta} \in \mathcal{X}$.

As an example, if the obstacle avoidance slack variable was to be added to the cost function of reaching a target position, as discussed in Section 4.1, the optimization

problem would be to minimize

$$J_{\text{obstacle}}(\boldsymbol{\eta}, \boldsymbol{\nu}, \boldsymbol{\eta}_{\text{ref}}, \boldsymbol{\nu}_{\text{ref}}, \mathbf{u}, \mathbf{s}) = \left\|\boldsymbol{\eta}_0 - \boldsymbol{\eta}_{\text{ref},0}\right\|_{Q_1}^2 + q_\psi c_\psi(\boldsymbol{\eta}_0) + \|\boldsymbol{\nu}_0\|_{Q_2}^2 + k_s s_0$$
$$\sum_{j=0}^{N}\Big(\left\|\boldsymbol{\eta}_{j+1} - \boldsymbol{\eta}_{\text{ref},j+1}\right\|_{Q_1}^2 + q_\psi c_\psi(\boldsymbol{\eta}_j) + \left\|\boldsymbol{\nu}_{j+1} - \boldsymbol{\nu}_{\text{ref},j+1}\right\|_{Q_2}^2 +$$
$$\left\|\mathbf{u}_j - \mathbf{u}_{j-1}\right\|_I^2 + \left\|\mathbf{u}_j\right\|_R^2 + k_s s_{j+1}\Big) \quad (4.10)$$

where the slack coefficient k_s was chosen to a high figure to give high penalization when the slack variables was given a value, in this case 10^9. For all the slack variables holds that $\mathbf{s} \geq 0$.

For this problem, the constraints

$$\sqrt{(x_j - x_{\text{obst}})^2 + (y_j - y_{\text{obst}})^2} \geq (r_{\text{obst}} + r_{\text{safety}}) - s_j, \quad j = 0, \ldots, N, N+1 \quad (4.11)$$

and

$$\sqrt{(x_j - x_{\text{dock},i})^2 + (y_j - y_{\text{dock},i})^2}$$
$$\geq \sqrt{\Big(\frac{L_p}{2}\cos(\psi_j) - \frac{W_p}{2}\sin(\psi_j)\Big)^2 + \Big(\frac{L_p}{2}\sin(\psi_j) + \frac{W_p}{2}\cos(\psi_j)^2\Big)} + m_{\text{dock}},$$
$$j = 0, \ldots, N, N+1, \quad i = 0, \ldots, D-1, D \quad (4.12)$$

hold. Here, $(x_{\text{obst}}, y_{\text{obst}})$ are coordinates of the obstacle in question, r_{obst} its radius and r_{safety} the safety radius. The parameters $(x_{\text{dock},i}, y_{\text{dock},i})$ are the coordinates for the dock and D the number of coordinates — so, to set the conditions regarding the dock, each of its coordinates is iterated for each position (x_j, y_j) of the vessel in relation to each of the dock coordinates, $(x_{\text{dock},i}, y_{\text{dock},i})$. The parameter m_{dock} can be seen as a margin to the dock, and L_p and W_p are the length and width of the vessel. The direction of the vessel will affect how close the vessel is to the obstacle, since marine vehicles in general have a greater length than width. The reason why the vessel's dimensions currently only are taken into account when being close to the dock, is that not including it in the constraints for avoiding a circular obstacle lowered the computation time of the MPC. Since it is desired to let the vessel come close to the dock, it was prioritized to make use of the dimensions in this situation. Instead, the safety radius was added to the obstacle.

In a corresponding manner, the slack variable should be added to the cost functions of waypoint following or logged trajectory, if there are obstacles to take into account.

Illustration for the obstacle avoidance can be seen in Figure 4.4, and for the dock in Figure 4.5.

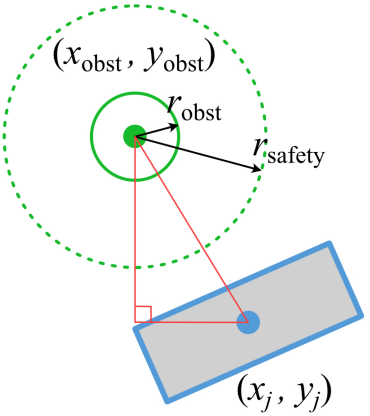

Figure 4.4 Illustration of the situation and different notations of obstacle avoidance.

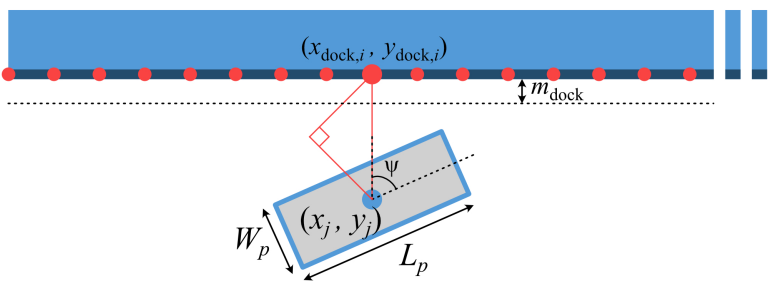

Figure 4.5 Illustration of the situation and different notations of dock avoidance.

5

Implementation and Simulation

The autopilot using MPC was implemented and tested in simulations. This allowed for fine-tuning of parameters, such as those in the cost function, and performance, for example to lower computation time. After successful testing of the implementation, it was run on a small-scale model car with Ilon wheels. This kind of wheels enables the vehicle to move freely in the xy-plane, without the restrictions of principally only being able to move forwards or backwards, as is associated to vehicles' normal wheels. The fact that the car was equipped with Ilon wheels made it possible for it to physically emulate movement of the boat. Experimenting on the car did not only simulate movement and usage of the MPC, but also a ROS interface. This facilitated the preparation for the ROS communication and necessary changes to be made in order for the implementation to work in this kind of environment in experiments, as opposed to when only being run in a digital Python environment.

5.1 Software

There are several software solutions available to implementing an MPC controller. Some of them were tested, to make a decision of which one to use.

MATLAB has a model predictive control toolbox, [The MathWorks, Inc., 2022b]. It has several examples and can be used together with Simulink. Another toolbox which was tested in combination with the MPC toolbox was the Control System Toolbox [The MathWorks, Inc., 2022a]. This toolbox facilitated the specification of a system model.

Another project is CVX/CVXGEN, which allows you to specify a convex optimization problem [CVXGEN, 2013]. Code is generated, either to use with MATLAB as interface or to use as C code. CVX also has support for Python.

Yet another tool for solving optimization problems is CasADi [CasADi, 2018] [Andersson et al., 2019]. It has an intuitive interface for setting up various system models and cost functions. This tool works with Python, MATLAB/Octave and C++.

Since control of the real Piraya vessel will be done through ROS, it was desirable to do simulations in the same programming language as will be used for the Piraya. Hence, either Python or C++ should be used. It was chosen to use CasADi in Python, to implement the MPC.

5.2 Simulations

For the simulations, the code was divided into three scripts, in addition to the main function. A schematic overview of the simulation scripts and their interactions can be seen in Figure 5.2. In the main function, the MPC function is called. Here the start position, start velocities, initial control signals, prediction horizon, maximum number of simulated seconds, and desired references are defined and used as input parameters to the MPC. Desired references vary depending on which approach is to be run, as described in Chapter 4. This is information such as target state, waypoints, and logged reference trajectory. The three other scripts will be explained in the following, starting with the script performing the MPC, which calls the scripts for calculation of the reference trajectory or point(s), and the script for simulating boat movement.

MPC and Simulation

The discrete-time system described in (3.12) was set up and defined as a symbolic CasADi function, with input parameters $\boldsymbol{\nu}(t)$ and $\mathbf{u}(t)$ and output $\boldsymbol{\nu}(t+T_s)$. After this, two CasADi symbolic arrays of size $(N+1, 3)$, where N is the prediction horizon in number of steps (that is, the prediction horizon in seconds divided by the time step T_s), were defined to contain position and velocities, and one CasADi symbolic array of size $(N, 2)$ was created to contain the control signals. These three arrays were defined as the variables, to which the problem should be optimized with respect to. The problem is set up by iterating over the prediction horizon, defining how velocity and position at time step $t + T_s$ relates to what it was at time step t. For the velocities, this is given by Equation (3.12), and for positions it is given by (5.1). That is, they are defined by the vessel's system dynamics.

In the case of a measured disturbance vector, as described in Section 3.4, this was added to the model dynamics. It was both added to the equations by which the problem setup is described, and to the boat model simulating one time step forward, as described in Section 5.2.

The velocity at time t is multiplied with the time step and added to the last position coordinate. The velocities are given in the vessel-fixed coordinate system and are hence multiplied with the rotation matrix to generate coordinates in the global coordinate system:

$$\boldsymbol{\eta}(t + T_s) = \boldsymbol{\eta}(t) + \mathbf{R}(\psi(t))\boldsymbol{\nu}(t)T_s \tag{5.1}$$

These constraints were set up for each time instant of the prediction horizon, and so was the cost function. The cost functions are specified for the different approaches, and are described in Chapter 4.

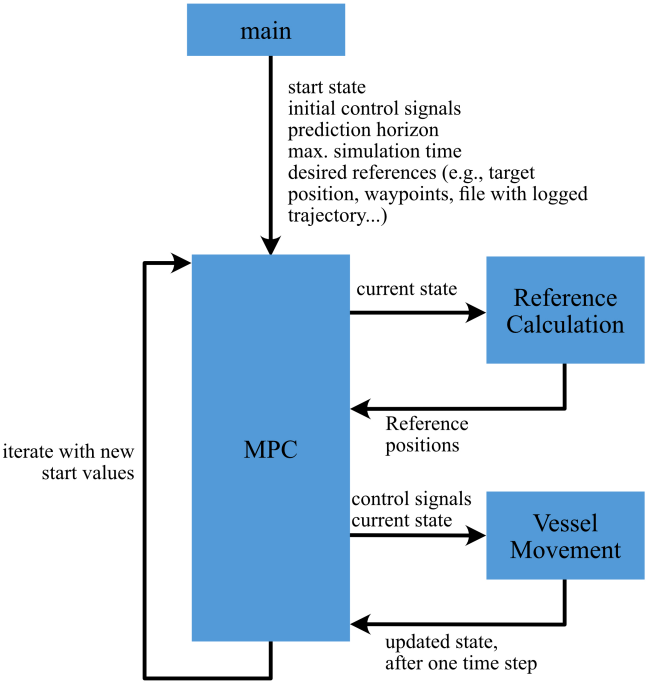

Figure 5.1 A schematic overview of the simulation scripts.

The optimization problem was solved at each sampling instant. It was also solved one time just before the iterations began, without sending those first control signals, but the first values of the optimized state, velocity, and control signals were set as first initial guesses for the numerical solution of the MPC optimization problem to avoid extra initial delay which else occured. For each iteration, the first values of the optimal vectors were set as initial guesses for the next iteration of the MPC.

Reference Calculation

For reference calculation, the functions to calculate the different kinds of reference points varied depending on how the desired route of the vessel is defined, as described in Sections 4.1–4.3; if it is only a target state, a target state with sub points along the way (waypoint following), or a trajectory defined by sampled points. For the first and second approaches, when only a target position and when having a target position with waypoints, the weight matrices are to be updated once the target position is within the prediction horizon. In simulations with no unpredicted external disturbance, it would be possible to set the condition such that the weight matrices change when the last optimized state over the prediction horizon exactly equals the target position,

but since it is not certain that it will be this exact when running experiments, it was instead set to having a tolerance of closeness. This is done with NumPy's function `allclose()` [NumPy Developers, 2022], which compares two vectors element-wise and returns true if they are equal within the tolerance, e.g., if $|a - b| \leq (a_{\text{tol}} + r_{\text{tol}} * |b|)$ is element-wise true. Here, a and b are vectors, in this case the last state of the optimized states and the desired target state. Parameters a_{tol} and r_{tol} are absolute and relative tolerances. For north and east position they were set in the order of 10^{-8} to 10^{-5}.

For the third approach the reference points are given as a trajectory, which is logged or calculated by a higher level path planner. For this case the trajectory is given in a .csv file, being collected as an array. No matter where the simulated vessel starts, it will be calculated where the closest point of the given trajectory is, and this point will be set as the first reference point. The following reference points will be collected from the given vector, for as far as the prediction horizon reaches, and be matched to the following optimized points of the prediction horizon. The distance is calculated as the Euclidean norm, and for each MPC iteration it will be calculated anew, always setting the closest point of the trajectory as the first reference value in $\boldsymbol{\eta}_{\text{ref}}$.

Vessel Movement

The vessel movement script in Figure 5.2 contains a model of the boat, to which control signals were sent. Using the given control signals, movement of one time step T_s was simulated. Since the equations of the boat model, in addition to using the control signals, also use current state and current velocities, they are also input parameters. After the next position $\boldsymbol{\eta}(t + T_s)$ and next velocity $\boldsymbol{\nu}(t + T_s)$ have been calculated through the system equations defined for the boat, they are returned.

Execution Time and Discretization

To be able to use a discretization other than the original $T_s = 0.2$ s is useful for control when the calculations of the MPC is taking longer than 0.2 s. Simulations showed that the discretization described in this section was feasible, hence no other discretization (e.g., the Runge–Kutta 4 method) was done.

As mentioned, the time steps of the model in [Ljungberg, 2021] are based on $T_s = 0.2$ s. Since the model is Euler discretized in accordance with (3.12), it is transformed to continuous time as below. Consider Equation (3.12). Dividing with

T_s, we get a continuous-time model, in accordance with (5.2).

$$\begin{bmatrix} \dot{u}(t) \\ \dot{v}(t) \\ \dot{r}(t) \end{bmatrix} = \begin{bmatrix} \frac{\theta_1}{T_s} u(t) & \frac{\theta_2}{T_s} v(t) r(t) & \frac{\theta_3}{T_s} \tau_x(t) \\ \frac{\theta_4}{T_s} v(t) & \frac{\theta_5}{T_s} u(t) r(t) & \frac{\theta_6}{T_s} \tau_y(t) \\ \frac{\theta_7}{T_s} r(t) & \frac{\theta_8}{T_s} u(t) v(t) & \frac{\theta_9}{T_s} \tau_y(t) \end{bmatrix}$$
$$= \begin{bmatrix} \theta_{\text{cont},1} u(t) & \theta_{\text{cont},2} v(t) r(t) & \theta_{\text{cont},3} \tau_x(t) \\ \theta_{\text{cont},4} v(t) & \theta_{\text{cont},5} u(t) r(t) & \theta_{\text{cont},6} \tau_y(t) \\ \theta_{\text{cont},7} r(t) & \theta_{\text{cont},8} u(t) v(t) & \theta_{\text{cont},9} \tau_y(t) \end{bmatrix} \quad (5.2)$$

This enables for discretization with time step of other length, $\hat{T}_s$. This is shown in (5.3).

$$\begin{bmatrix} u(t+\hat{T}_s) \\ v(t+\hat{T}_s) \\ r(t+\hat{T}_s) \end{bmatrix} = \begin{bmatrix} u(t) \\ v(t) \\ r(t) \end{bmatrix} + \begin{bmatrix} \hat{T}_s \theta_{\text{cont},1} u(t) & \hat{T}_s \theta_{\text{cont},2} v(t) r(t) & \hat{T}_s \theta_{\text{cont},3} \tau_x(t) \\ \hat{T}_s \theta_{\text{cont},4} v(t) & \hat{T}_s \theta_{\text{cont},5} u(t) r(t) & \hat{T}_s \theta_{\text{cont},6} \tau_y(t) \\ \hat{T}_s \theta_{\text{cont},7} r(t) & \hat{T}_s \theta_{\text{cont},8} u(t) v(t) & \hat{T}_s \theta_{\text{cont},9} \tau_y(t) \end{bmatrix} \quad (5.3)$$

5.3 Experiments with Ilon Car

Experiments were done with the Ilon car, a small model car with Ilon wheels. The wheels enable the car to move freely in the xy-plane and to simulate the boat dynamics. Since the Robot Operating System (ROS) is used on both the car and the Piraya, working with the car would allow testing the MPC and communication between the ROS packages. The flowchart of the different instants for driving of the car can be seen in Figure 5.2 and they will be described more detailed in the following. The car itself can be seen in Figure 5.3.

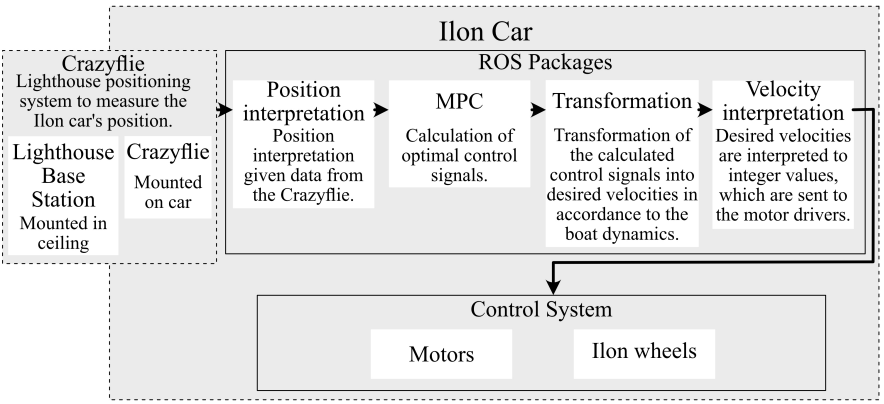

Figure 5.2 Schematic of the Ilon car system for MPC and boat dynamic simulations.

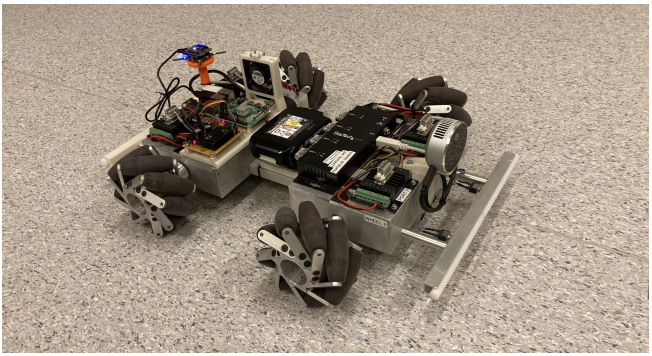

Figure 5.3 The Ilon car.

Positioning

For positioning, a lighthouse positioning system [Bitcraze AB, 2022b] was used. A Crazyflie [Bitcraze AB, 2022a] with a lighthouse deck needed to be mounted to the car, so a holder for the Crazyflie was modelled in SolidWorks [Dassault Systèmes SolidWorks Corporation, 2022] and 3D-printed. To this, the Crazyflie was mounted such that its heading was in the same direction as the car. Lighthouse base stations were mounted to the ceiling. They emit infra-red laser which is detected by the lighthouse deck on the Crazyflie and enable positioning of the car.

The positions $\boldsymbol{\eta}$, as defined in (3.2), as well as velocities along the global x- and y-axis are obtained through the Crazyflie. These data are in ENU-frame, (east, north, up) and were converted to the NED-frame as

$$\begin{bmatrix} x_{\text{NED}} \\ y_{\text{NED}} \\ \psi_{\text{NED}} \end{bmatrix} = \begin{bmatrix} y_{\text{ENU}} \\ x_{\text{ENU}} \\ -(\psi_{\text{ENU}} - \frac{\pi}{2}) \end{bmatrix}. \tag{5.4}$$

Velocities were converted in the same manner from the ENU to the NED frame, and after this multiplied with the inverted rotation matrix $\mathbf{R}(\psi)^{-1}$ to obtain velocities $\boldsymbol{\nu}$ of the car's local, body-fixed frame.

Since the Crazyflie is offset from the center of the Ilon car, this is to be compensated for. The new coordinates are calculated as follows. Indices "CF" indicate it is the data given from the Crazyflie. The offset is notated with δ_i, in x or y, respectively. The angle ψ is calculated such that it will range between $-\pi$ and π. As reference image,

see Figure 5.4. All values are now assumed to be in the NED-frame.

$$\left.\begin{aligned} x &= x_{\text{CF}} + \delta_x \cos(\psi) + \delta_y \sin(\psi) \\ y &= y_{\text{CF}} + \delta_x \sin(\psi) - \delta_y \cos(\psi) \\ \psi &\in [0, 2\pi) \\ \delta_x &= 0.185 \text{ m} \\ \delta_y &= 0.035 \text{ m} \end{aligned}\right\} \tag{5.5}$$

The calculated and transformed values are sent to the MPC with a ROS message.

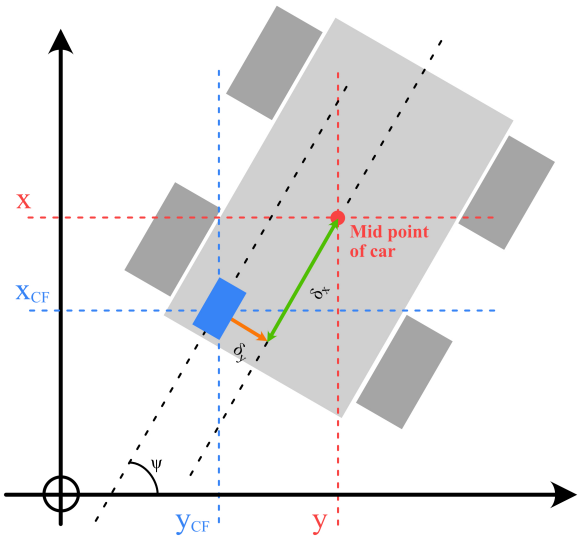

Figure 5.4 Position of the Crazyflie on the car in relation to the car's mid-point. The drawing is not true to scale.

Data from the Crazyflie are sampled, and sent directly after calculations, with a rate of 0.2 s since this is the smallest time step the MPC will have.

MPC

The script with the MPC uses the received position and velocity data as start point and start velocity, for each MPC iteration. The Crazyflie did not have data for rotation velocity. This was instead calculated in the MPC as difference in angle divided by the sample time.

After this, an optimal solution is planned and calculated much in the same manner as described in Section (5.2). When optimal control signals for thrust and rudder

have been computed, they are sent to the script for transformation. This describes one iteration of the script. After this, new start position and velocities are collected. A callback changes values of current state as soon as a new message with data is published to the topic, so for the MPC, it will get the current value whenever it is ready.

Piraya Simulation

The Ilon car is controlled through a script which acquires ROS message of the type Twist [Open Robotics, 2022], containing velocities in the car's local coordinate system. The Twist message expresses linear and angular velocities. To make the car simulate the boat, velocities $\boldsymbol{\nu}$ are calculated through the system model of the boat, according to (5.3), using the control signals received from the MPC-calculations. When these are sent to the script of the car controlling the motors, the car will move in a boat-like manner. After calculation of how the control signals would cause the vessel to move, the velocities were divided by 80 to make the car move much slower, as it had such a small area available. This way, it had better mobility, corresponding to its size in proportion to the Piraya's. When sending positive sway with the Twist object, the car would get a velocity to the left, which is the opposite direction of its body-fixed NED frame. Because of this, the sway was multiplied with -1 to get the correct sign.

6

Results of Simulations and Experiments with Ilon Car

The results of the performed simulations and experiments with the Ilon car will be discussed. Firstly, the simulations will be shown, and secondly, the experiments with the small-scale car. The results of the experiments with the Piraya are discussed in Chapter 7.

6.1 Simulations

Simulations were performed for the three different approaches discussed in Chapter 4.

For the case of only one target position, simulations were performed with three different settings, to show the influence of external disturbances. A simulation without external disturbances, a simulation with a disturbance not modelled in the MPC, and a simulation with the same disturbance but also modelled in the MPC, as discussed in Section 3.4, were performed.

For the approach of following waypoints, an S-shaped curve with waypoints was to be followed. The coordinates of the waypoints were specified through sea map of [Eniro Sverige AB, 2022], and the waypoints were to be followed during the experiments at Gränsö.

To simulate the approach of following a logged trajectory, the reference trajectory will be of a trajectory previously logged using a model simulation. Control signals were sent to a model of the vessel, making it run a path. The path was logged with sampling period $T_s = 0.2$ s.

Lastly, simulation of obstacle avoidance was performed. It was implemented to the case of having one target point. Both a circular obstacle to avoid and a dock to not get too close to, were added.

One Target Position

The results of using the MPC to optimize to reach a target position can be seen in Figures 6.1–6.6. The target position was satisfactorily reached for all three cases, when

only considering the path. Figures 6.1 and 6.2 are from simulation of an undisturbed model, i.e., the simulated boat to which the control signals were sent, returned the expected next states. Figures 6.3 and 6.4 show results from the same setup, but the simulated boat had an offset corresponding to −0.2 m/s in x-direction, +0.5 m/s in y-direction, and zero rotational offset. Lastly, Figures 6.5 and 6.6 show results from the simulation of where the same offset is assumed to have been measured and hence is implemented in the model of the MPC, onto which the computations of control signals are based. For all three cases, T_s was set to 0.4 s and prediction horizon to 20 s, and thus $N = 40$. The reason to this is to give the vessel enough time to slow down or react, as it cannot brake. However, the length of the prediction horizon could be investigated further, for optimal length.

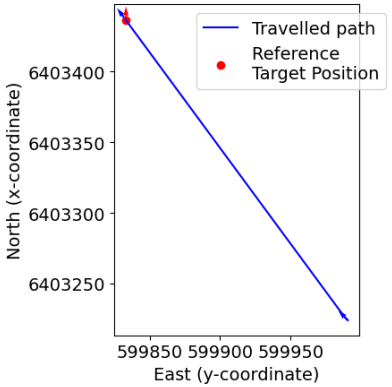

Figure 6.1 Path of undisturbed model.

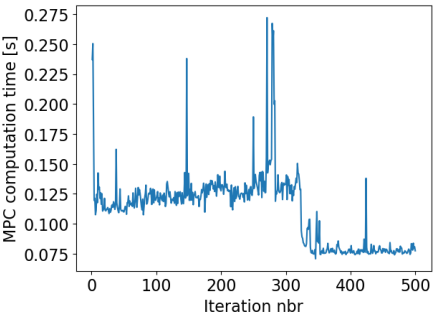

Figure 6.2 Computation time of undisturbed model.

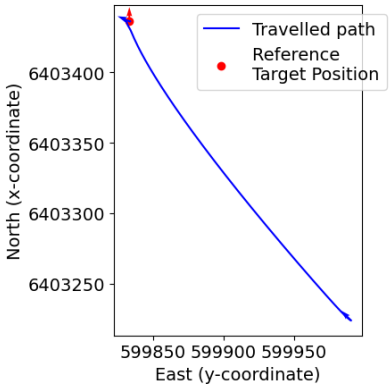

Figure 6.3 Path of disturbed model, without implemented offset in MPC.

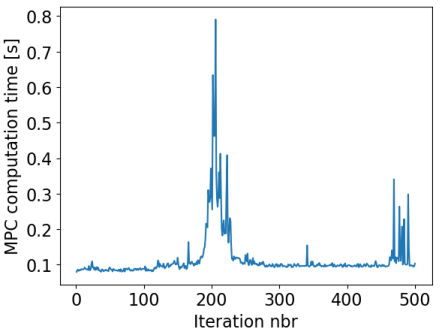

Figure 6.4 Computation time of disturbed model, without implemented offset in MPC.

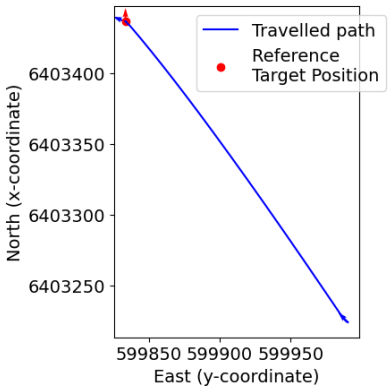

Figure 6.5 Path of model with implemented offset in MPC.

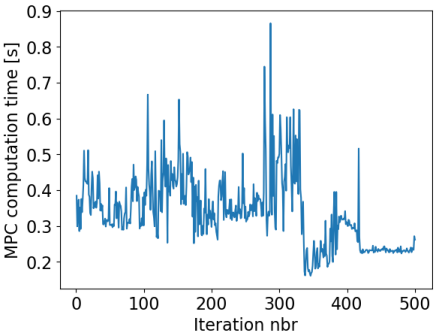

Figure 6.6 Computation time of model with implemented offset in MPC.

As can be seen, the target point is reached in all three cases, but the path of the disturbed model in Figure 6.3 has some curvature and makes a small, extra twirl at the end. The resulting path of having a calculated offset is straighter, see Figure 6.5. However, calculation time differs for each MPC iteration. For the undisturbed model, it is below 0.4 s with a good margin (see Figure 6.2). For the disturbed one, it reaches above in some cases (see Figure 6.4), but for the one with measured offset, it is often taking longer than 0.4 s (see Figure 6.6). This means that the control signals would be sent slower than what the model in the MPC has calculated for, and can degrade performance of the control. The cause of the long computation times might be that the model now includes transformation of the offset velocities from the global frame

to the body-fixed frame, and the offset in surge and sway at each time instant will differ, since it will depend on the heading at the instant. This makes it much more computationally expensive. Still, it gives a more precise result, so it might be a useful method if it could be acceptable with longer sampling period T_s.

Waypoint Following

For evaluating waypoint following with the real Piraya vessel at Gränsö, some reference points had been decided upon from the maps in [Eniro Sverige AB, 2022]. The same points are used in the simulations of the waypoint following. The start position was set to a position a bit off from the first waypoint, T_s was set to 0.4 s and the prediction horizon $N = 50$. Results of the simulation can be seen in Figures 6.7–6.9.

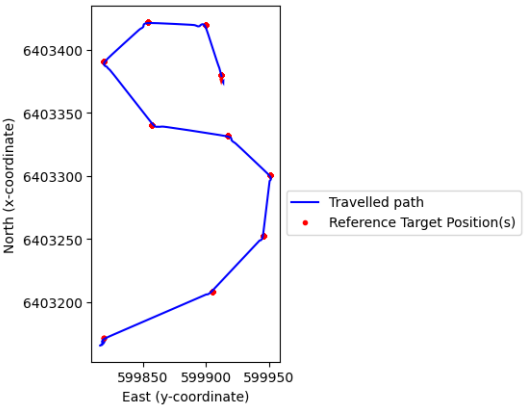

Figure 6.7 Simulation of waypoint following, travelled path.

In the figures it can be seen that the S-shape is being followed, after first finding the first waypoint. It is also possible to see the effect of the sway that arises when the rudder is turned as it starts aiming for the next waypoint, both by studying the travelled path and by comparing yaw rate and sway plots, which are mirrored in sign and proportion.

Logged Trajectory

The results of following a logged trajectory can be seen in Figures 6.10–6.11. This trajectory and the MPC model was logged with sampling period $T_s = 0.2$ s. Figure 6.10 shows results when start position and velocities were the same as when the trajectory was logged. Figure 6.11 shows when the vessel started some distance away from the trajectory and with zero start velocities.

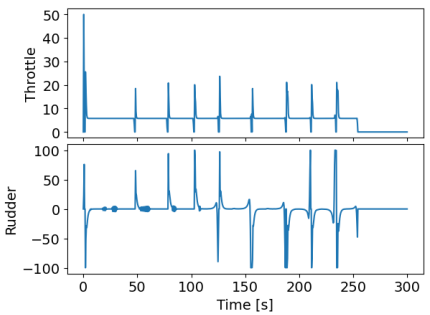

Figure 6.8 Simulation of waypoint following, control signals.

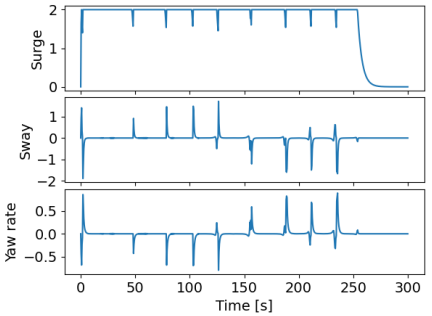

Figure 6.9 Simulation of waypoint following, velocities. Surge and sway in $\mathrm{m\,s^{-1}}$, yaw rate in $\mathrm{rad\,s^{-1}}$.

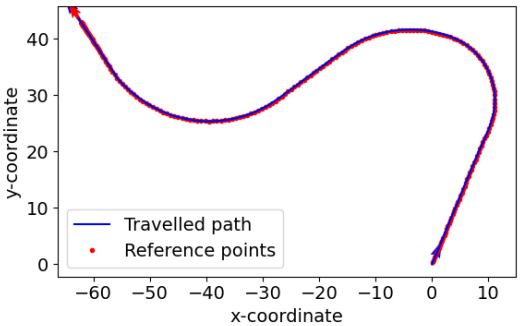

Figure 6.10 Following a logged trajectory reference, with same initial values as when the trajectory was logged. The x- and y-axes are in meters.

Obstacle Avoidance

Obstacle avoidance while reaching a target position, with a circular obstacle and a dock with absolute positions, was also tested. The circular object had a radius of 1 m and a safety radius of 3 m. This means, if the vessel were to enter within the 3 m radius, there would be high penalization of the slack variable. The dock was to be kept at a distance of 0.4 m, in relation to its size and angle, as described in (4.11). The obstacle was set up in such a manner so that the vessel's start position, target point, and obstacle was placed along a straight line, forcing the vessel to face the obstacle. Simulation results can be seen in Figure 6.12. As the vector of slack variables was added to the cost function and, for each time instant, it was needed to recalculate the distance due to the vessel's heading, the MPC computation in the case of obstacle avoidance takes too long to be feasible. The time step T_s was set to 0.4 s and the prediction horizon N to 10. As can be seen, the vessel takes a right turn by

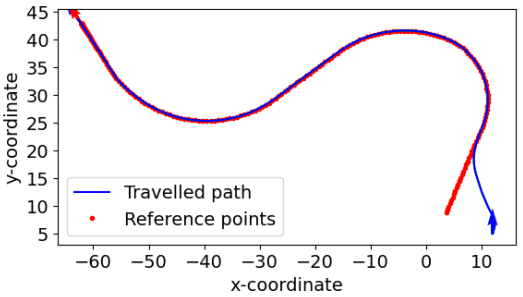

Figure 6.11 Following a logged trajectory reference, starting at a position away from the logged trajectory. The x- and y-axes are in meters.

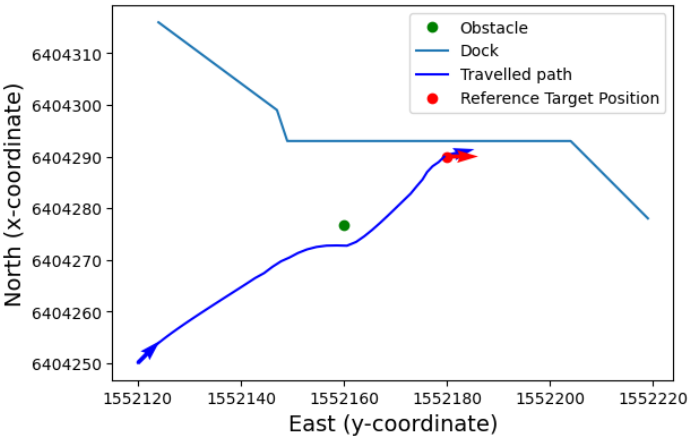

Figure 6.12 Obstacle avoidance with a target position as reference. The circular obstacle can be seen as the green dot, the dock is illustrated with the light blue line.

the obstacle, going south of it. If the prediction horizon was possible to make longer, it might have optimized the control inputs differently, to facilitate reaching the target position from another angle as this would mean less use of the control signals $\mathbf{u}$. It would be possible to, instead of shortening the prediction horizon, increase the sampling period. However, this would lower the accuracy of the target positioning too much. One possible solution to this would be to lower the velocities and approach the target position slowly when being close, but it is difficult to foresee how this would affect the control, as the vessel would be more sensitive to drifting and might need to adjust heading, but having the correlation of sway and yaw rate making it difficult to adjust too much if close to a dock.

6.2 Experiments with Ilon Car

Experiments with the Ilon car tested to reach a single target position and to reaching it with a waypoint on the way. Results can be seen in Figures 6.13–6.15 for the single target position, and in Figures 6.16–6.18 for the approach of reaching a waypoint before the target position. It can be noticed, that for the waypoint reference, the radius of when to switch aiming point was set to 0.1 m, as opposed to the Piraya's 5 m.

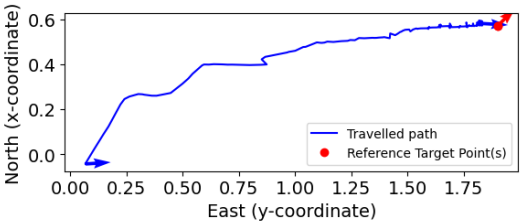

Figure 6.13 Path of Ilon car, experiment of the approach of having one target position as reference. The x- and y-axes are in meters.

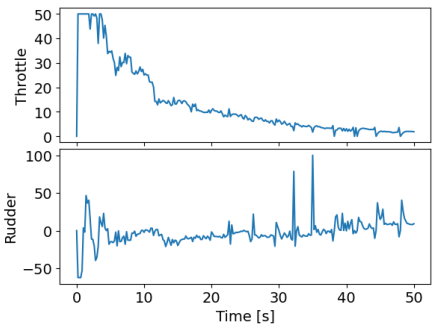

Figure 6.14 Control signals of Ilon car, experiment of the approach of having one target position as reference.

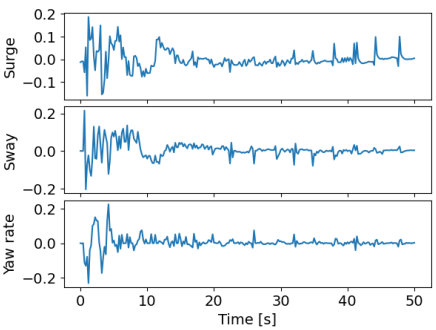

Figure 6.15 Velocities of Ilon car, experiment of the approach of having one target position as reference. Surge and sway in $\mathrm{m\,s^{-1}}$, yaw rate in $\mathrm{rad\,s^{-1}}$.

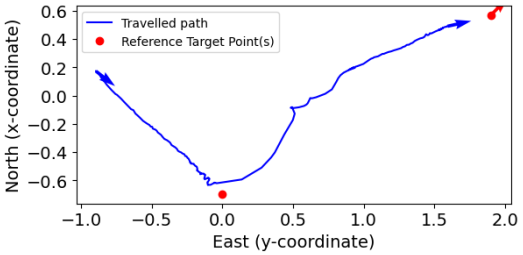

Figure 6.16 Path of Ilon car, experiment of the approach of having a waypoint to reach before the target position.

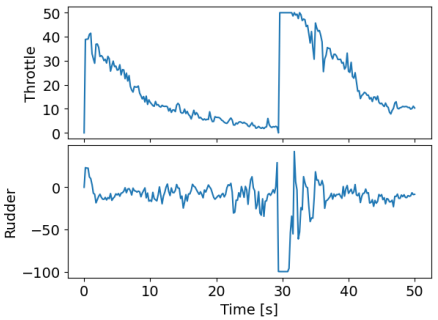

Figure 6.17 Control signals of Ilon car, experiment of the approach of having a waypoint to reach before the target position.

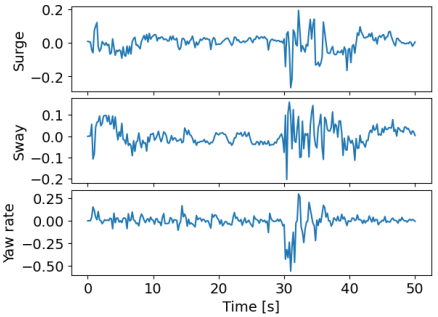

Figure 6.18 Velocities of Ilon car, experiment of the approach of having a waypoint to reach before target the position. Surge and sway in $\mathrm{m\,s^{-1}}$, yaw rate in $\mathrm{rad\,s^{-1}}$.

In both results, the path looks irregular with sharp turns. The control signals vary a lot too, along with the velocities. Some unevenness could be observed in the car's movement, but not the extent as observed in the diagrams. The observed unevenness can partially be explained by attempts to compensate for shaky positioning when an outlier position value is received, causing uneven control signals and in extension an uneven path. Another part could be that the velocities were kept so low, and that the wheels had problems gripping the floor, making them grip and slip at times. The additional unevenness seen in the diagrams, in both positions and velocities, is most likely because of error in the received data from the Crazyflie positioning system. This data is probably also what is causing the controller to the, at times, strange control actions. However, it can be seen that the car in general manages to get the position correct.

7

Piraya Experiments at Gränsö

The MPC script was tested on the actual Piraya vessel at Gränsö, Västervik, during May 2022. A map of the area can be seen in Figure 7.1. The image is a cropped screenshot of the map provided by [Team of OpenSeaMap, 2022].

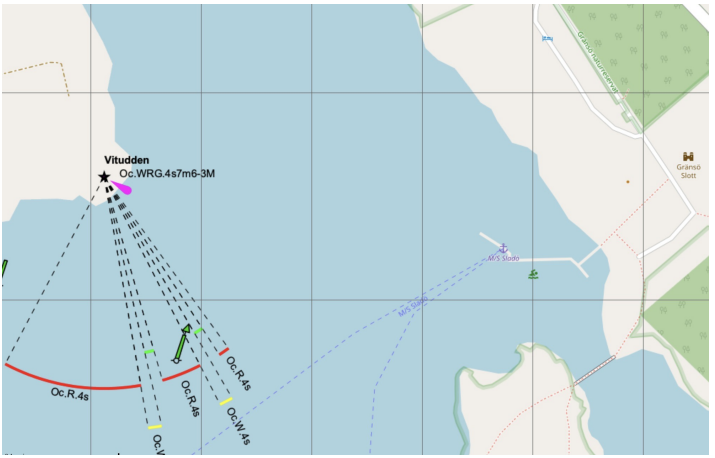

Figure 7.1 The test area, with the dock from which the experiments were conducted to the right. The picture is a cropped screenshot of the map from [Team of OpenSeaMap, 2022].

The only modifications made, were regarding the interface. Three experiments were to be performed; to reach a target point, to travel along a set of waypoints, and to reach a target point with a certain desired angle by the dock. The two latter will be discussed here, since the first one mainly ensured that the vessel responded correctly to the MPC commands and moved reasonably.

In simulations, it was feasible to set a maximum velocity of 2 m/s, except for when unexpected disturbances were added. In reality, unexpected disturbances were always present, and so it was not possible to set an allowed maximum velocity since it would not be able to keep it and the problem would be considered infeasible. Instead, a maximum of the throttle signal was set, to attain a maximum velocity of approximately 2 m/s. This gave a satisfactory surge of just below 2 m/s.

It was decided to not evaluate the function of following a logged trajectory. A logged trajectory could approximately be compared to having lots of waypoints. As the waypoint following worked well, it was decided to prioritize other experiments with the Piraya.

For the waypoint experiment, the points which were also used in the simulations, forming the S-like curve, were to be followed. The vessel was further away and facing the opposite direction than in simulations, but still manages to reach the first point and all the following points satisfyingly. It could be worth to mention, it was quite windy during the test, with winds up to 8 m/s from the south-west, hence approximately straight towards land where the dock was placed. The time step for the MPC was set to $T_s = 0.4$ s. The result can be seen in Figures 7.2–7.5. A comparison of the approximate route in the test area can be seen in Figure 7.3. The curve has been edited onto a cropped screenshot of a map from [Team of OpenSeaMap, 2022]. It can be possible to discern the correlation of sway and yaw rate, as discussed in Section 3.3, for each control with respect to the waypoints. The drifting at the end can be explained by force from the wind. The MPC was stopped when the vessel had reached the target point, but logging continued. This experiment was done with the simple waypoint-controller, without an offset implemented.

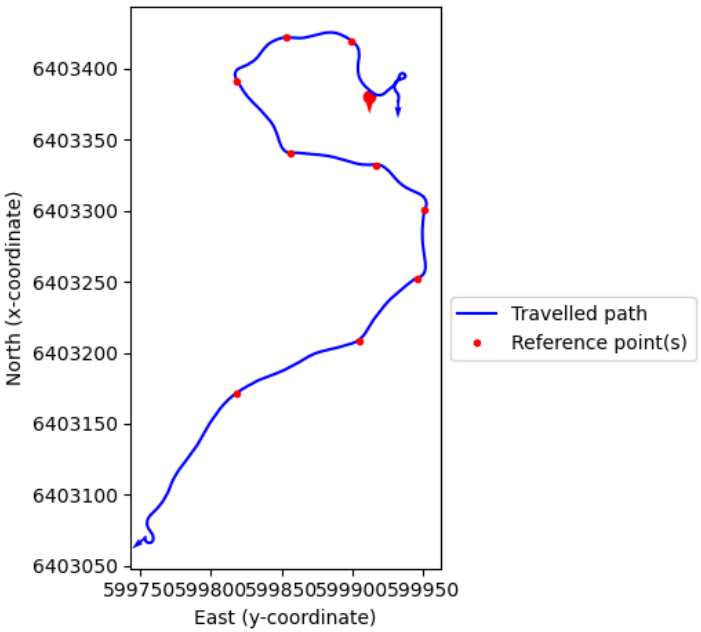

Figure 7.2 Travelled path when the Piraya was controlled through waypoint following. Each time a waypoint is reached and the next one is set as reference point, it is possible to notice the correlation of sway and yaw rate, as discussed in Section 3.3.

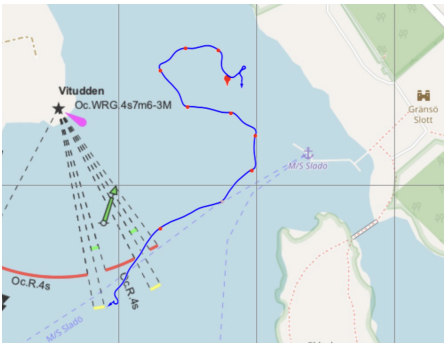

Figure 7.3 A comparison of the approximate route in the test area. The curve has been edited onto a cropped screenshot, with lowered opacity for better visualization, of a map from [Team of OpenSeaMap, 2022].

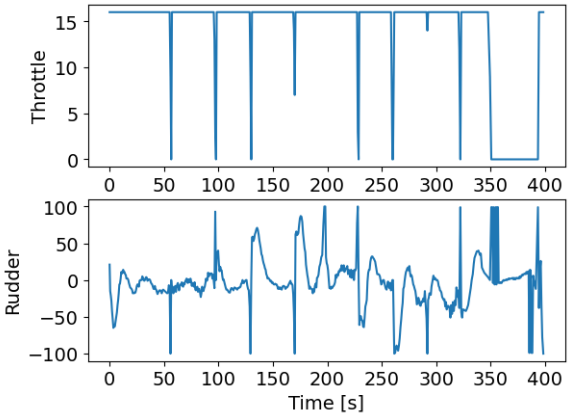

Figure 7.4 Control signals when the Piraya was controlled through waypoint following.

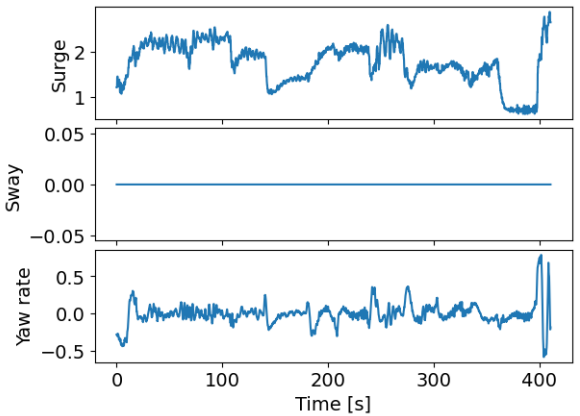

Figure 7.5 Velocities when the Piraya was controlled through waypoint following. The reason to the sway being zero is that no sensor data about sway were given. Surge and sway in $\mathrm{m\,s^{-1}}$, yaw rate in $\mathrm{rad\,s^{-1}}$.

For finding the desired target position closer to the dock, the result can be seen in Figures 7.6–7.9. The desired target position was reached satisfactorily, even though the vessel's start position was on a distance and facing the wrong direction. A photography of the Piraya when target position had been reached through in this experiment, can be seen in Figure 7.10. The target position was some meters away from the dock, facing south. After the vessel had reached the target position, it keeps drifting since it cannot brake or have negative surge. However, notice that it is doing this at very

low velocities. End velocities were $\boldsymbol{\nu} = \begin{bmatrix} 0.2453688 & 0.0 & -0.016875638 \end{bmatrix}^T$ m s^{-1}. Regarding the correlation of sway and yaw rate, it can be noticed that yaw rate increases at time = 20 s and at time ≈ 95 s the vessel turns starboard. At time = 20 s, it is possible to notice that the travelled path deviate a bit to port. However, the same behavior is not as noticeable at time ≈ 95 s. As sway was not logged, it is difficult to see the exact correlation.

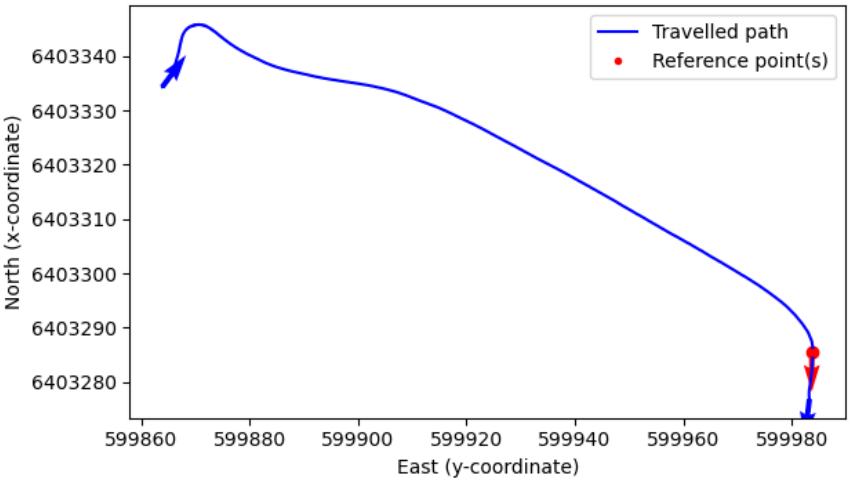

Figure 7.6 Travelled path when a specified target position was to be reached.

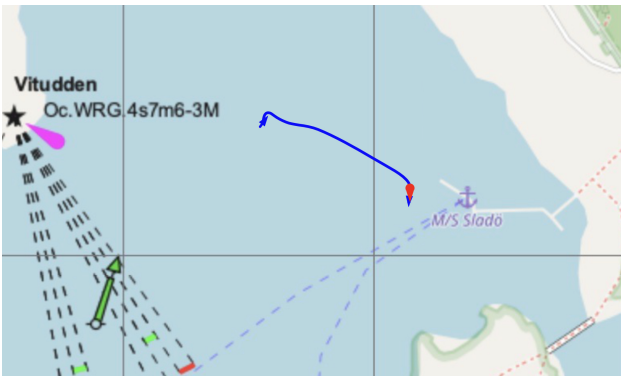

Figure 7.7 A comparison of the approximate route in the test area. The curve has been edited onto a cropped screenshot, with lowered opacity for better visualization, of a map from [Team of OpenSeaMap, 2022]. A part of the dock is not visible on the map, which makes the vessel look further away from it than it was in reality.

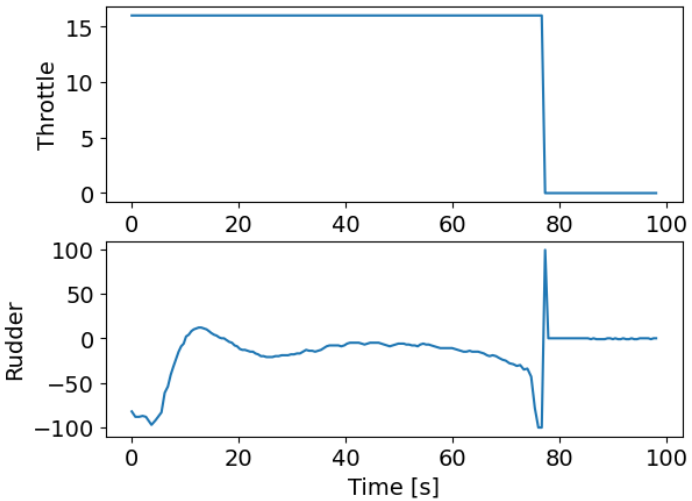

Figure 7.8 Control signals when a specified target position was to be reached.

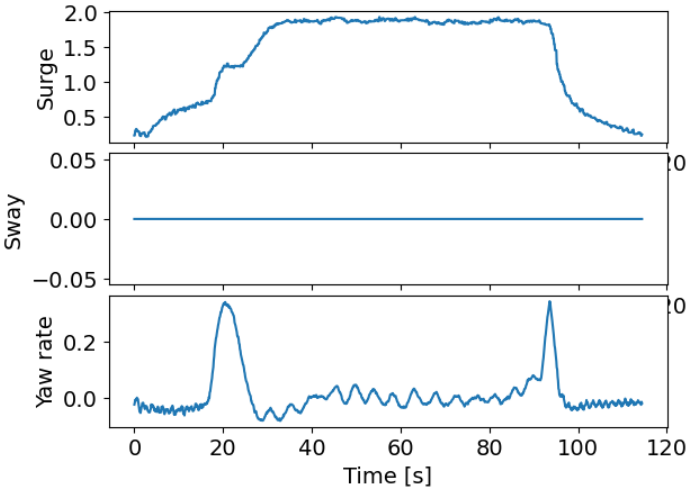

Figure 7.9 Velocities when a specified target position was to be reached. Notice the two peaks of yaw rate at time = 20 s and at time ≈ 95 s. Surge and sway in m s^{-1}, yaw rate in rad s^{-1}.

Figure 7.10 Real Piraya reaching its target position, as in Figure 7.6.

8

Conclusions

A model predictive controller for docking a sea vessel at low speeds has been developed, as well as a base for using the Ilon car to simulate the vessel's behavior. It has then been shown that the model defined by [Ljungberg, 2021] was feasible, and that it was possible to successfully control the vessel by using it in an MPC, which gave satisfactorily results, shown through both simulations and experimental evaluation on the real vessel (a Piraya vessel). In both cases, the control shows high accuracy. Regarding heading at the target position, it was even more accurate when controlling the real vessel. This was most likely due to that the vessel in reality has a smaller turning radius than the model. Another noticed difference between simulations and experiments was regarding the correlation of sway and yaw rate. The behavior of deviating to the opposite side when turning one way, could be clearly seen in simulations. At some instants of the experiments it could be observed, but not at all times.

Different kinds of methods for travelling towards a target point were explored and shown to be effective. The vessel could plan its path after input of conditions such as waypoints, target heading, obstacles, and measured offset. However, the computation time in some simulation cases, as with the implemented, measured offset, exceeded the assumed sampling time. To reduce computation time, a proposal for possible modification is to choose a shorter control horizon. This would reduce the number of variables to be optimized for the whole prediction horizon.

Experiments with the Ilon car did not show as high accuracy as the simulations and when running on the real Piraya. Since simulations and experiments of the MPC-implementation worked well, the undesired behavior of the Ilon car is probably because of fault in positioning and measured velocities. Implementing a Kalman filter or outlier detection could help to reduce the faulty signals, to deduce what position values are probable.

Experimenting with the real Piraya showed that it could successfully follow waypoints even in the presence of moderate wind, without it being measured and taken into account.

To summarize in relation to the problem formulation in Section 1.2, an MPC for performing docking has been developed and implemented. Three approaches to

handle the situation of reaching a docking position, as well as a suggestion of handling external disturbances, obstacles, and close to dock maneuvering, have been designed. This has been done with consideration to the conditions of harbor environment, for instance as all maneuvers are performed at low speeds, and through the path planning. For the three main approaches, a proposal is to use the one having a single target point as reference when target position is relatively close and nothing further in the environment needs to be considered. The approach of using waypoints could be suggested to be used when the target position is desired to be reached from a certain angle, for instance if the target position is a delimited mooring. Waypoints could also be used when target position is far away and some headland, mark, or similar should be rounded before, bur velocities still should be low. To use the approach of logged trajectory is suggested when it is of importance that the trajectory, before reaching the target position, is performed with precision. This could be if there are lots of seafarers in the area. The approaches could also be used in combinations, for example to follow a logged trajectory or path decided by a high level path planner, but once a certain area has been reached the approach of only having one target position is practiced.

8.1 Future Work

To the developed control design and implementation, it would be possible to add functions to make it even more autonomous. Target state, waypoints, and obstacles could be perceived and computed autonomously instead of added manually. For example, a LiDAR sensor could be implemented to spot obstacles, and could be used together with a sea map for detecting known shallows. With a LiDAR sensor, moving obstacles could be included, and could possibly be avoided using the same approach as discussed for static obstacles. Usage of the MPC in combination with a high level path planner could also be a further extension, for instance in the case of following a logged trajectory.

For further improvement of the MPC implementation, it could be investigated exactly how many values should be set as initial guesses for the optimization problem. In this thesis, only the first values of the optimal solutions were set, as this showed the quickest computation time of the MPC for experiment cases at Gränsö. However, the optimal number differs due to the situation, and to adjust this appropriately could possibly decrease computation time. Also, the tuning of the weight parameters in the cost function have potential of being further investigated, for the optimal relation between them, even if the current, described parameters have approximately the correct dimensions.

Regarding the system model of the vessel, it was noticed that the turning radius of the simulated vessel and the real vessel differed. This and an updated estimation could be a desired extension of the model. It could be interesting to further investigate this behavior for more precise controlling. To log sway and be able to observe the

correlation could be a first step.

Another extension to the model could be to add the function of using negative throttle, for the possibility of having negative surge. This would make accuracy even higher in the sense that the vessel could easier stay at a desired position, and not keep drifting forward even if it is at low speed.

Hopefully, the developed controller can be used for future support of development of autonomous docking, being applied on other USVs and improve safety when navigating in harbors.

Bibliography

Andersson, J. A. E., J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl (2019). "CasADi – A software framework for nonlinear optimization and optimal control". *Mathematical Programming Computation* **11**:1, pp. 1–36. DOI: 10.1007/s12532-018-0139-4.

Andersson, O., P. Doherty, M. Lager, J.-O. Lindh, L. Persson, E. A. Topp, J. Tordenlid, and B. Wahlberg (2021). "WARA-PS: a research arena for public safety demonstrations and autonomous collaborative rescue robotics experimentation". *Autonomous Intelligent Systems* **1**. DOI: 10.1007/s43684-021-00009-9.

Åström, K. J. and B. Bernhardsson (2016). *Model Predictive Control (MPC)*. URL: https://www.control.lth.se/fileadmin/control/Education/DoctorateProgram/ControlSystemsSynthesis/2016/MPC.pdf (visited on 2022-06-02).

Bitcraze AB (2022a). *Crazyflie 2.1*. URL: https://www.bitcraze.io/products/crazyflie-2-1/ (visited on 2022-04-20).

Bitcraze AB (2022b). *Lighthouse positioning system*. URL: https://www.bitcraze.io/documentation/system/positioning/ligthouse-positioning-system/ (visited on 2022-04-20).

CasADi (2018). *CasADi*. URL: https://web.casadi.org/ (visited on 2022-06-10).

CVXGEN (2013). *CVXGEN: code generation for convex optimization*. URL: https://cvxgen.com/docs/index.html (visited on 2022-05-10).

Danielsson, T. (1965). *Att angöra en brygga*. AB Svensk Filmindustri.

Dassault Systèmes SolidWorks Corporation (2022). *Solidworks*. URL: https://www.solidworks.com/ (visited on 2022-04-20).

Duun-Henriksen, A., S. Schmidt, R. Røge, J. Møller, K. Nørgaard, J. Jørgensen, and H. Madsen (2013). *Model Identification Using Stochastic Differential Equation Grey-Box Models in Diabetes – Scientific Figure on ResearchGate*. DOI: 10.1177/19322968130070022O. URL: https://www.researchgate.net/figure/Illustration-of-the-concept-of-grey-box-modeling-White-box-models-are-based-mainly-on%7B%5C_%7Dfig5%7B%5C_%7D236138545 (visited on 2022-05-12).

Eniro Sverige AB (2022). *Eniro kartor*. URL: https://kartor.eniro.se/ (visited on 2022-06-01).

Fossen, T. I. (1994). *Guidance and Control of Ocean Vehicles*. John Wiley & Sons.

Ljungberg, F. (2020). *Estimation of Nonlinear Greybox Models for Marine Applications*. Licentiate Thesis no. 1880. Department of Electrical Engineering, Linköping University.

Ljungberg, F. (2021). "Systemidentifiering för Piraya". Saab Kockums AB.

Martinsen, A. B., G. Bitar, A. M. Lekkas, and S. Gros (2020). "Optimization-Based Automatic Docking and Berthing of ASVs Using Exteroceptive Sensors: Theory and Experiments". *IEEE Access* **8**, pp. 204974–204986. DOI: 10.1109/ACCESS.2020.3037171.

Naukowe, Z. and A. Miller (2014). "Model predictive control of the ship's motion in presence of wind disturbances". *Scientific Journals Maritime University of Szczecin* **39**, pp. 107–115.

NumPy Developers (2022). *Numpy.allclose*. URL: https://numpy.org/doc/stable/reference/generated/numpy.allclose.html (visited on 2022-05-24).

Open Robotics (2022). *Geometry_msgs/twist message*. URL: http://docs.ros.org/en/noetic/api/geometry%7B%5C_%7Dmsgs/html/msg/Twist.html (visited on 2022-04-20).

Team of OpenSeaMap (2022). *OpenSeaMap - The free nautical chart*. URL: https://map.openseamap.org/ (visited on 2022-06-07).

The MathWorks, Inc. (2022a). *Control system toolbox*. URL: https://se.mathworks.com/products/control.html (visited on 2022-05-10).

The MathWorks, Inc. (2022b). *Model predictive control toolbox*. URL: https://se.mathworks.com/products/model-predictive-control.html (visited on 2022-05-10).

Voigt, J. and A. Alkaysi (2020). *Autopilot for a Personal Watercraft*. Master's thesis, TFRT-6103. Department of Automatic Control, Lund University.

Wallenberg AI, Autonomous Systems and Software Program (WASP) (20, 2021a). *Autonoma drönare på räddningsuppdrag*. URL: https://www.mynewsdesk.com/se/wallenberg-ai-autonomous-systems-and-software-program/

pressreleases / autonoma - droenare - paa - raeddningsuppdrag - 3138536 (visited on 2022-03-15).

Wallenberg AI, Autonomous Systems and Software Program (WASP) (2021b). *WARA-Public Safety*. URL: https://wasp-sweden.org/research/research-arenas/wara-ps-public-safety/ (visited on 2022-06-07).

Wikimedia Foundation, Inc. (2022). *Drag equation*. URL: https://en.wikipedia.org/wiki/Drag%7B%5C_%7Dequation (visited on 2022-05-08).

Yang, Y., J. Du, H. Liu, C. Guo, and A. Abraham (2013). "A trajectory tracking robust controller of surface vessels with disturbance uncertainties". *IEEE Transactions on Control Systems Technology* **22**, pp. 1511–1518. DOI: 10.1109/TCST.2013.2281936.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER'S THESIS

Date of issue
June 2022

Document Number
TFRT-6164

Author(s)
Sofia Kockum

Supervisor
Birgitta Wingqvist, Dept. of Automatic Control, Lund University, Sweden
Björn Olofsson, Dept. of Automatic Control, Lund University, Sweden
Anders Robertsson, Dept. of Automatic Control, Lund University, Sweden
Karl-Erik Årzén, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle
Autonomous Docking of an Unmanned Surface Vehicle using Model Predictive Control

Abstract

Autonomous docking of marine vessels presents challenges different from the ones faced when travelling at open sea or in an archipelago. External disturbances due to the varying environment and accurate positioning at low speed are examples of these kinds of challenges. The aim of this work has been to develop and implement an autopilot algorithm for docking. This was to be done for a specific marine vessel, using model predictive control (MPC). The vessel in question was Saab Kockums' Piraya, an unmanned surface vehicle (USV).

The problem formulation includes design and implementation of the MPC itself, as well as handling external disturbances and obstacle avoidance. Different software approaches to solving the optimization problem have been explored, as well as different solutions to the matter of advancing towards the desired position. Three approaches were implemented in an MPC framework.

The developed controllers have been successfully tested in simulations, in experiments using a small-scale model car dynamically representing the USV, as well as on the USV at Gränsö, Västervik.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-61

Recipient's notes

Security classification

http://www.control.lth.se/publications/