University of South-Eastern Norway

www.usn.no

FMH606 Master's Thesis 2024

Industrial IT and Automation

Dynamic positioning, system identification and control of marine vessels - based on Balchen model

Abu Tammam Bin Nuruddin

Faculty of Technology, Natural sciences and Maritime Sciences

Campus Porsgrunn

University of South-Eastern Norway

www.usn.no

Course: FMH606 Master's Thesis, 2024

Title: Dynamic positioning, system identification and control of marine vessels - based on Balchen model

Number of pages: 98

Keywords: *Dynamic Positioning, Reduced Balchen Model, Model Predictive Control, Optimal Control with integral action, IAE and TV index.*

Student:	Abu Tammam Bin Nuruddin
Supervisor:	David Luigi Di Ruscio
External partner:	None

Summary:

Dynamic Positioning (DP) is essential for ensuring the secure operation of marine vessels across different industries. Hence, it is important to use reliable models and accurate control techniques for these vessels.

The main objective of this study is to simplify the Balchen low frequency model and develop accurate control methods to perform simulation experiments on the simplified model.

The model is simplified by eliminating the drag and momentum coefficients, and its behavior is observed. A comparison between the simplified model and the original model is performed. Standard MPC, Reduced Size MPC, Simple MPC, Simple MPC with integral action, and LQ optimal control with integral action were selected as control methods. The performance of the controllers is evaluated using Integrated Absolute Error (IAE) and Total Variation (TV) indexes.

The simulation results shows that the simplified model is a three-double integrator model. The comparison between the simplified model and the original model gives almost identical behavior. Based on the experimental results and the evaluation indexes, the Simple MPC with integral action is chosen as the best control approach for the simplified model, although the Reduced Size MPC also performs well overall.

Preface

Dynamic Positioning (DP) is a system which utilizes active thrust to automatically keep the vessel in position. The system utilizes thrusters, rudders, and propellers to withstand wind, waves, and current. Dynamic positioning systems were first introduced in 1960 and improved and modified to support the emerging offshore and oil industries.

One of the earliest methods that are implemented on the vessel is Balchen model. The model was used to build a DP system based on Kalman filtering and optimal control.

This thesis develops a dynamic positioning system based on a simplified Balchen model. Additionally, the thesis investigates the efficiency of the system using various types of controllers and provides a performance analysis of these controllers. MATLAB form Math-works was used to perform all the simulations.

This research is completed in Spring 2024 as a requirement for the Industrial IT and Automation course at the University of South-Eastern Norway.

I want to express my sincere gratitude and thanks to my supervisor Professor David Luigi Di Ruscio for his invaluable advice, guidance and direction throughout the research period. Finally, I want to give my thanks to my family and friends who always gave their support and encouragement.

Porsgrunn, 15th of May, 2024

Abu Tammam Bin Nuruddin

Contents

Nomenclature

Symbol		Description
$R(\psi)$	:	Transformation Matrix (NED to Body)
$F_{w_{su}}$	:	Wind force in surge [N]
$F_{w_{sw}}$	:	Wind force in sway [N]
N_w	:	Wind moment in yaw [N.m]
ρ	:	Density of the wind [kg/m^3]
V_{w_r}	:	Relative wind speed [m/s]
C_x, C_y, C_N	:	Wind coefficients
A_F	:	Windage area of the head wind [m^2]
A_L	:	Windage area of the beam [m^2]
L	:	Vessel length [m]
$v_{c_{su}}$	:	Water current velocity in surge direction [m/s]
$v_{c_{sw}}$	:	Water current velocity in sway direction [m/s]
N_c	:	Water current moment in yaw [N.m]
τ	:	Thruster Force [N]
K	:	Thruster coefficients
l	:	Thruster lever arm
m	:	Mass of a body [kg]
a	:	acceleration of a body [m/s^2]
F	:	Forces action on the body [N]
x_{su}	:	Position in Surge [m]
x_{sw}	:	Position in Sway [m]
ψ	:	Heading in yaw [rad]
v_{su}	:	Velocity in Surge [m/s]
v_{sw}	:	Velocity in Sway [m/s]

v_ψ	:	Heading rate [rad/s]
$F_{t_{su}}$	:	Thruster force in Surge [N]
$F_{t_{sw}}$	:	Thruster force in Sway [N]
N_t	:	Thruster moment in Yaw [N.m]
η_1, η_2, η_3	:	Gaussian white noise
d_i	:	Drag and momentum coefficients
x	:	State vector
u	:	Input vector
y	:	Output vector
w	:	Process noise
v	:	Measurement noise

Abbreviations		**Description**
DP	:	Dynamic Positioning
MIMO	:	Multiple Input Multiple Output
LQ	:	Linear Quadratic
MPC	:	Model Predictive Control
NED	:	North, East, Down coordinate System
GPS	:	Global Positioning System
DOF	:	Degrees of Freedom
IAE	:	Integrated Absolute Error
TV	:	Total Variation

List of Figures

List of Tables

1 Introduction

It is easy to imagine that the ocean is flat with a ship navigating its surface in a straightforward manner, capable of moving forward, backward, turning right, and turning left. However, a closer look shows just how dynamic the surface of the ocean can be. It is a bumpy, turbulent and ever-changing plane where a ship truly moves not just up and down also rolling to port and starboard and swiveling on the axis. The ship's motion is impacted by various factors such as the main engine, wind, waves and the current. It is important to understand these motions, even if they are small, so that it is clear exactly where the ship is when it is mapping, diving, recovering ROVS, operating offshore oil rigs, and other things. Nowadays, vessels use dynamic positioning systems to navigate in the oceans.

1.1 Dynamic positioning systems

A dynamic positioning system is a computer-based system that manipulates a vessel's thruster and propulsion system to maintain a desired location during environmental disturbances [1]. The system collects data from a variety of sensors, including wind sensors and location sensors. Wind sensors are used to measure both the velocity and direction of wind. Position sensors, such as GPS and gyro-compass, provide information about the precise location and direction of a vessel. The dynamic positioning system utilizes the gathered data to calculate a control signal that controls the propulsion system [2].

The DP system is a MIMO system. The system inputs consist of the forces produced by the thrusters in the surge and sway directions, as well as the moment given by the thrusters around the vertical axis. The system provides the vessel's position in surge and sway, as well as the vessel's heading in yaw. Figure 1.1 illustrates the inputs, outputs, states, and disturbances typically encountered by a vessel integrated with a Dynamic Positioning (DP) system.

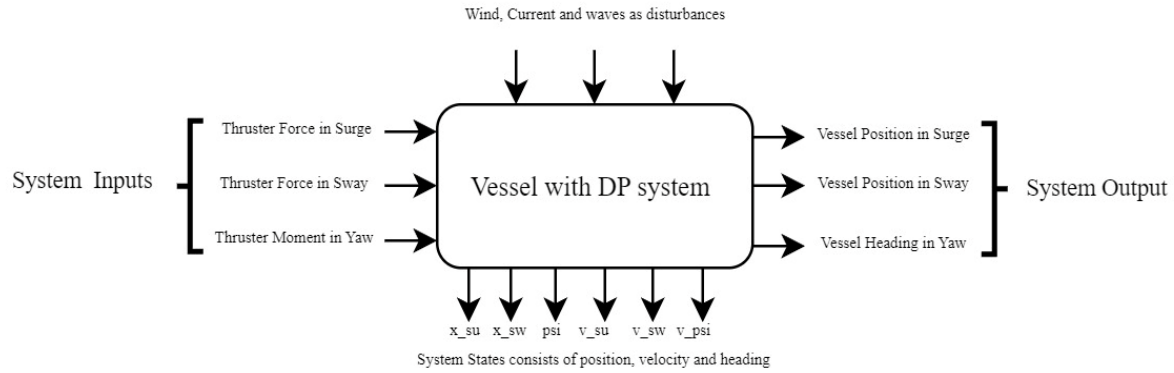

Figure 1.1: Block diagram of a vessel with DP systems.

The dynamic positioning system includes three components. The power subsystem is the first component of the DP system and supplies the necessary electrical power. The thruster subsystem provides the dynamic positioning (DP) system with the ability of adjusting both the power and direction of the propulsion force. The DP control subsystem works in combination with other subsystems to carefully regulate the position of the vessel [2].

1.2 Previous research on DP system of marine vessel

The research on dynamic positioning systems began in the early 1960s in the United States, which resulted in the development of a manual DP system [3]. The most notable improvement happened in the mid-1970s when Kalman filters and linear quadratic optimal controllers were implemented. The approach used in this study was based on a mathematical model, which was employed to predict and evaluate the vessel's movements. These systems required significant processing power in comparison to the computer resources that were accessible at that time. However, the standard was established, and it is accurate to state that the current leading manufacturers continue to benefit from the advantages of the outcomes and experiences gained all through that time. In 1977 a study was conducted for testing a newly developed control system for dynamic placement on the vessel 'Seaway Eagle' by Balchen and his team of researchers [4] [5]. Balchen's proposal involves removing the conventional PID control technique and rather utilizing present-day control theory principles, such as the Kalman filter and optimum control. Another British team, led by M. Grimble in 1980, has created a novel adaptive filtering algorithm designed for systems with unknown disturbances [6]. The estimator integrates both an adaptive filter and a Kalman filter. The state estimations are employed in a closed-loop feedback control system that is developed using the standard linear quadratic strategy.

Sørensen and Fossen (1996) propose a dynamic positioning (DP) system for marine vessels, demonstrating its ability to automatically position and guide the vessel using thruster and propeller operations. The study describes control functions that enable both station-keeping and tracking. The primary component of the DP system is a control architecture based on models, which includes an updated Linear Quadratic Gaussian (LQG) feedback controller and a model reference feedforward controller [7]. Another paper from Fossen and Sørensen present a novel approach to detect the dynamic characteristics of ships that are controlled by Dynamic Positioning systems [8]. The authors have developed an off-line parallel extended Kalman filter algorithm to address the challenging problem, which involves nonlinear and multivariable processes. The algorithm is confirmed by conducting comprehensive full-scale testing on a supply vessel, which utilizes dual sets of measurement data. This testing confirms the durability of the estimator and its capacity to converge. The study indicates that their approach can accurately forecast ship dynamics, hence improving control of DP systems using parameters that align with real-world data.

There was also a lot of effort given to develop nonlinear control. Vessel DP systems are typically constructed with the assumption that the equations of motion can be linearized around a fixed yaw angle. This allows for the application of linear and gain scheduling approaches.

Fossen and Grøvlen in 1998 presents a globally exponentially stable (GES) nonlinear control approach that removes the need for this assumption. A non-linear observer has been integrated into the design, hence requiring just position measurements [9]. Aarset also performed almost similar studies in 1998 [10]. One notable benefit of the nonlinear ideas is the significant reduction in the time needed for tuning and calibration during the installation of a new system. This is due to the decreased complexity of the model and controller. The nonlinear approaches developed by Fossen have been successfully applied in a full-scale turret-anchored Floating Production, Storage, and Offloading (FPSO) system [11]. Aamo and Fossen (1999) proposed a technique for effectively managing both the thruster and line tension in oil exploration and production activities conducted in deeper waters [12]. Fossen also created a Marine Systems Simulation Toolbox, known as MSS, using MATLAB [13]. This toolbox is designed to model DP systems [14]. At the lower end of the spectrum, a number of vendors now provide manual joystick control systems that have restricted DP capabilities based on the studies conducted by Källström and Theorén in 1994 [15].

1.3 Objective of this research

One of the earliest mathematical models for vessel Dynamic Positioning (DP) was developed by Balchen [4]. This model was implemented to create a DP system using Kalman filtering and optimal control. This report presents the reconstruction of the DP system, along with the development of a modified and simplified model. The model is based on three double integrators, with the drag and momentum coefficients set to zero in the surge, sway, and yaw directions. As the control subsystem is the key component of the dynamic positioning system, this study will mainly focus on the control system.

A linear version of the reduced Balchen model is developed, which is then implemented in MATLAB to analyze and compare its behavior with the reduced model. Different controllers are used for the experiment. First, the Standard MPC controller is implemented to assess the system's performance. The Standard MPC uses excessive calculations in solving the optimization problem. Therefore, a Reduced-size MPC controller is also implemented for evaluating the system's performance. The study also includes a Simple MPC, developed by David Di Ruscio [16], to see the model's performance. Finally, using system dynamics, a LQ optimal controller with integral action is created and tested. The performance analysis of the controller is also done using the Integrated Absolute Error (IAE) and Total Variation (TV) index. Figure 1.2 shows the simple block diagram of the DP system developed for this report.

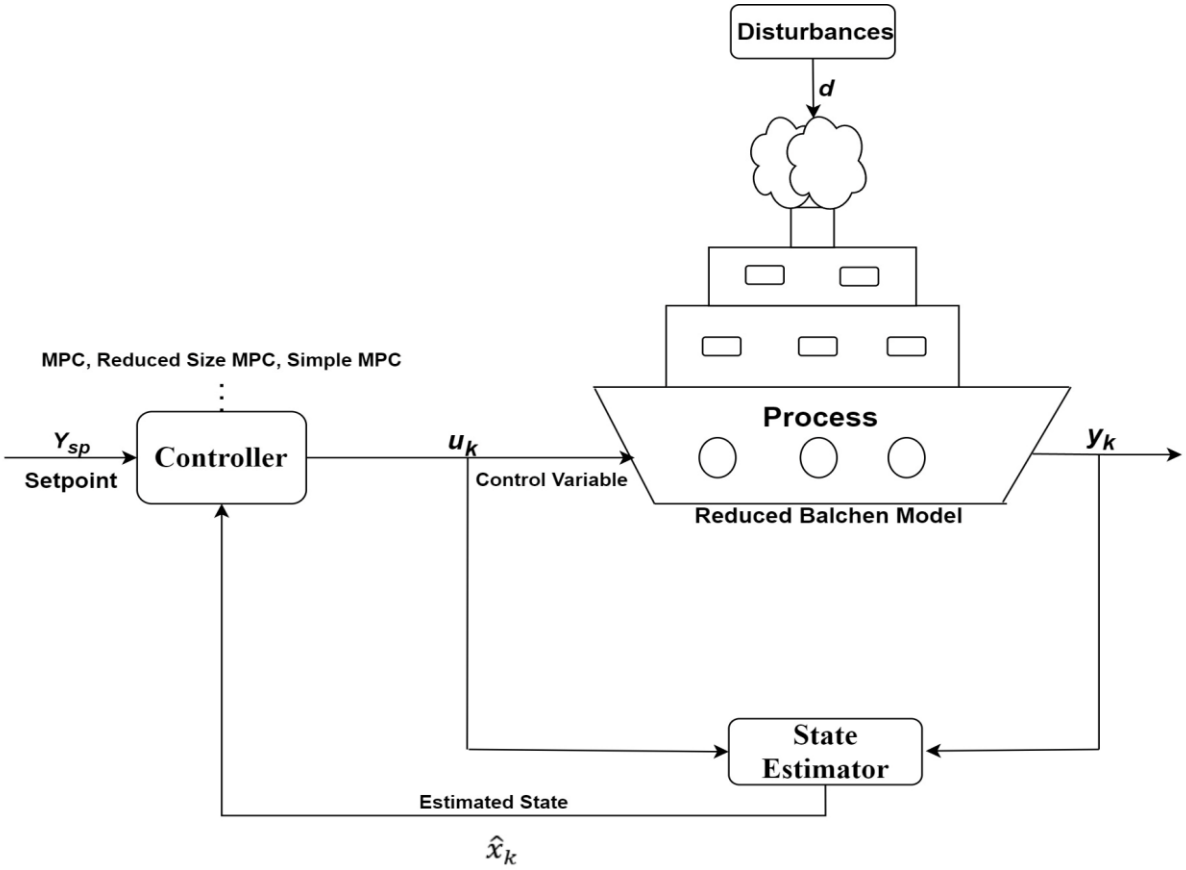

Figure 1.2: Basic block diagram of the DP control system with reduced Balchen model as process, different controllers and state estimator.

1.4 Report structure

This report has six chapters. The first chapter serves as an introductory section that provides a comprehensive outline of a dynamic positioning system, prior research conducted on it, and the primary objectives of this paper.

The second chapter includes a description of the Balchen mathematical model. Additionally, this chapter provides the simplified Balchen model, which is later used for analysis. In addition, the text also discusses the forces acting on the marine vessel and the coordinate systems used in the ocean.

The third chapter outlines controlling methods such as: MPC, LQ controller and Simple MPC which are employed in creating a effective DP system, whereas the fourth chapter presents the simulations conducted on the model and analyzes the outcomes drom the experiments.The performance analsysis of the different controllers also done in this chapter.

The fifth chapter discusses the key findings obtained from the experiment, as well as proposing some ideas for further research. The final section provides a conclusion to the findings of this paper.

2 Theory

This chapter presents an introduction to the fundamental concepts associated with Balchen's mathematical model of DP system of marine vessel. Also, the simplified Balchen model, which is developed without considering drag and momentum coefficients is discussed. Furthermore, the coordinate system employed by the marine vessel, as well as the forces that affect the vessel movement in the ocean are documented in this chapter.

2.1 Vessel coordinate system

In the design of feedback control systems for marine vessels, it is common to utilize reduced-order models due to the absence of all degrees of freedom (DOFs) in most vehicles. One DOF models are applicable for designing controllers for forward speed (surge), autopilots for heading (yaw), and devices for damping roll motion [17].

Three DOF models are typically employed for ships, semi-submersibles, and undersea vehicles in horizontal-plane configurations. These models uses surge, sway, and yaw motions. They have uses in dynamic positioning (DP) systems, trajectory-tracking control systems, and path-following systems.

Six DOF models, which include surge, sway, heave, roll, pitch, and yaw, are a set of interconnected equations of motion that are utilized for simulating and predicting the combined movements of a vessel. These models can also be utilized in sophisticated control systems for underwater vehicles, which are operated in all degrees of freedom [17]. Figure 2.1: Six degrees of freedom of a marine vessel shows the six DOF of a vessel.

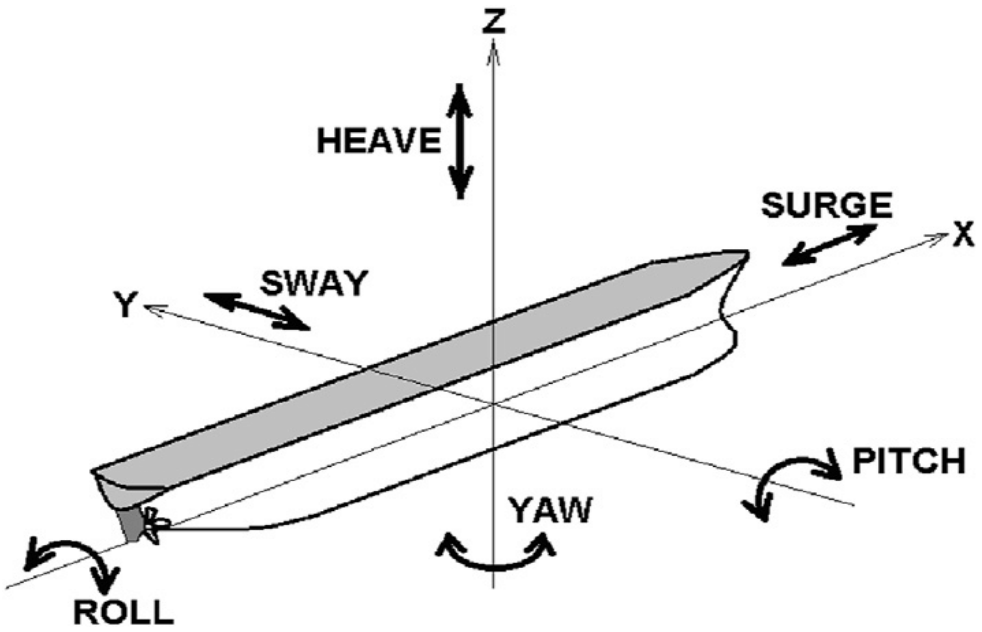

Figure 2.1: Six degrees of freedom of a marine vessel. [18]

Degrees of freedom refer to the vessel's capacity to navigate in different directions while at ocean. Three of them related to axial movements along the x, y, and z axes, while the other three correspond to rotational motions around those axes. The motions that occur along the x, y, and z axes are known as surge, sway, and heave movements. Pitch, roll, and yaw are the terms used to describe the rotational movements around the x, y, and z axes.

Two separate kinds of coordinate systems are used for vessel navigation. The NED coordinate system in a DP vessel provides directions with respect to the Earth's surface, mainly North, East, and Down. In a DP vessel, the body frame coordinate system is fixed to the vessel and its origin located at the center of mass of the vessel. The utilization of coordinate systems is essential for achieving accurate control and navigation. These systems perform a critical role in allowing DP systems to effectively maintain the precise position and heading of a vessel, particularly under challenging conditions. Each coordinate system is shown in Figure 2.2. Equation 2.1 shows the transformation matrix R(ψ) which converts coordinates from the NED frame to the body frame [17].

$$R(\psi) = \begin{bmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad 2.1$$

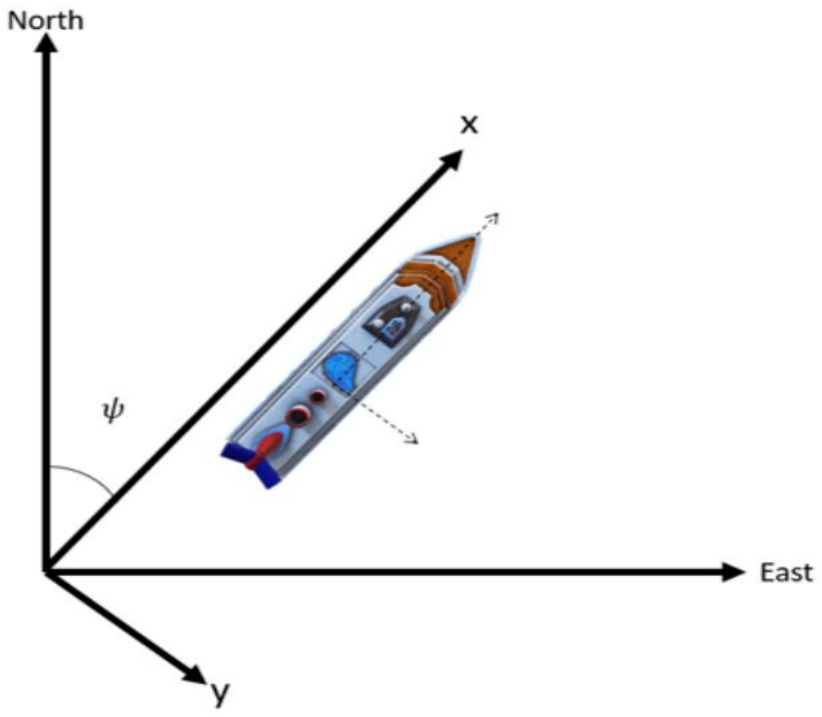

Figure 2.2: NED and Body coordinate system. [19]

2.2 Forces acting on the vessel

The forces exerted on the vessel can be divided into two categories: external influences, such as wind, water current, and sea wave forces, and internal forces, which include the vessel's thrusters and propulsion forces [17].

2.2.1 Wind force

The wind forces and moment experienced by a moving vessel are directly related to the square of the relative wind speed (V_{w_r}) ,wind angle of attack (γ) and the windage area [17]. The windage area indicates the specific area of a vessel that is directly exposed to the impact of the wind. The overall size of this region is defined by the vessel's shape. It is necessary to calculate and integrate wind forces in surge and sway, as well as wind moment in yaw and send this information into the control system of dynamic positioning systems in order to minimize their effect on the vessel's movement. The calculation of wind forces and moments for symmetric vessels can be conducted using equation 2.2 [17] [18].

$$F_w = \begin{bmatrix} F_{w_{su}} \\ F_{w_{sw}} \\ N_w \end{bmatrix} = \frac{1}{2}\rho {V^2}_{w_r} \begin{bmatrix} C_x cos(\gamma) A_F \\ C_y sin(\gamma) A_L \\ C_N sin(2\gamma) A_L L \end{bmatrix} \quad 2.2$$

Where,

$F_{w_{su}}$: wind force in surge [N]

$F_{w_{sw}}$: wind force in sway [N]

N_w: wind moment in yaw [N.m]

ρ: density of the wind [kg/m^3]

V_{w_r}: relative wind speed [m/s]

C_x , C_y, C_N : wind coefficients

A_F, A_L : windage area of the head wind, windage area of the beam

L: vessel length [m]

2.2.2 Water Current force

Forces which are exerted on a vessel by water currents are directly equal to the square of the difference between the velocity of the vessel and the velocity of the water current. In order to incorporate the current speed impact into the vessel motion equations proposed by Balchen [4], it is necessary to convert them to the Body-frame coordinate system. Balchen employed equations 2.3 to 2.5 for showing the water current speed with parameters which vary slowly.

$$\dot{v}_{c_N} = \eta_{c_N} \tag{2.3}$$

$$\dot{v}_{c_E} = \eta_{c_E} \tag{2.4}$$

$$\dot{N}_{c_\psi} = \eta_{c_\psi} \tag{2.5}$$

v_{C_N}: water current velocity in North direction in the NED coordinate system.

v_{C_E}: water current velocity in East direction in the NED coordinate system.

N_{C_ψ}: water current moment in yaw in the NED coordinate system.

$\eta_{c_N}, \eta_{c_E}, \eta_{c_\psi}$: zero mean white noise process.

$$\begin{bmatrix} v_{c_{su}} \\ v_{c_{sw}} \\ N_c \end{bmatrix} = R(\psi)^T \begin{bmatrix} v_{c_N} \\ v_{c_E} \\ N_{c_\psi} \end{bmatrix} \tag{2.6}$$

Where:

$v_{c_{su}}$: water current velocity in surge direction in Body coordinate system.

$v_{c_{sw}}$: water current velocity in sway direction in Body coordinate system.

N_c: water current moment in yaw in Body coordinate system.

2.2.3 Wave force

To simulate a dynamic positioning system, one can divide and analyze the wave-induced forces into their first and second-order effects [17]. First-order waves are oscillatory motions that have a zero mean. Second-order waves are forces that are generated by the slowly varying parts of wave drift forces and are not equal to zero. When constructing DP systems, it is essential to evaluate the effectiveness in the presence of wave [17]. Wave forces can be divided into two components: a gradually varying component and an oscillating component. The feedback control system has to compensate for oscillatory components [17].

To reduce the mean component, the integral action can be used. The oscillatory component can be frequently removed by simply applying a low-pass filter [17]. Multiple wave force models are available for the purpose of prediction. One of the most used model is the State-Space Model for Wave Responses. This model offers a straightforward and effective depiction of wave forces, making it appropriate for modeling and assessing feedback control systems.Wave forces are excluded and not incorporated into the control system in this report. However, readers who are interested in more comprehensive information can see to reference [17].

2.2.4 Thruster and propulsion force

Multiple kinds of thrusters exist. The most popular types are primary propellers, tunnel thrusters, and azimuth thrusters. The primary propellers influence the vessel's movement in the surge direction. Tunnel thrusters influence the sideways position of a vessel, while azimuth thrusters can rotate the vessel around the z-axis and impact its heading [17]. Figure 2.3 shows the different thrusters in a DP marine vessel.

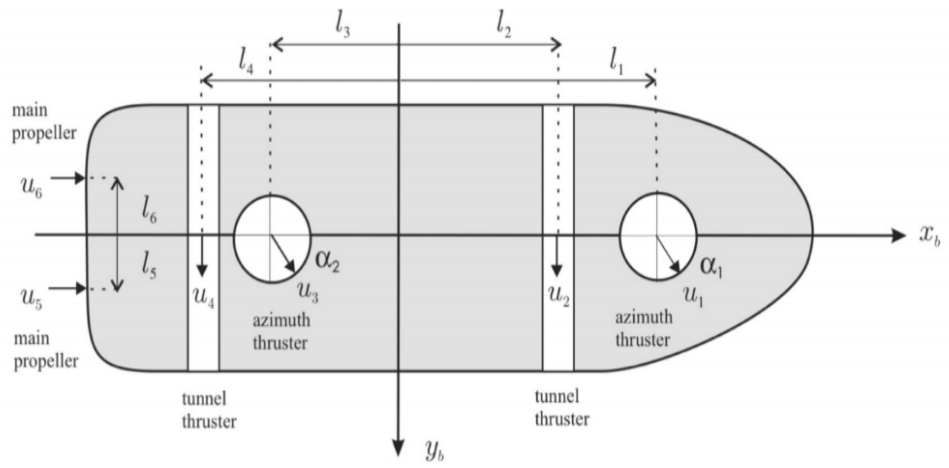

Figure 2.3: Different thrusters in a DP marine vessel [17].

Equation 2.7 provides the generalized thruster forces in surge, sway, and yaw showed in Body-frame coordinate system [17].

$$\tau = \begin{bmatrix} F_x \\ F_y \\ l_x F_y - l_y F_x \end{bmatrix} \quad 2.7$$

Where l_x, l_y are thrusters lever arms with respect to the center of the vessel.

The thruster configuration matrix Tcan be used to find out the magnitude of the force produced by each thruster in all directions. The *T* matrix has a column number equal to the number of thrusters and a row count equal to the degrees of freedom, which is typically three in dynamic positioning systems. The initial row of the *T* matrix indicates the forces generated by the thrusters in the forward direction. The second row presents the forces applied in the sideways direction, while the third row indicates the torques produced by the thrusters in the rotational direction.The generalized force can be described as the force that acts on a system as shown in equation 2.8 [17].

$$\tau = TF \quad 2.8$$

$$F = Ku \quad 2.9$$

Where,

K: Thruster coefficient diagonal matrix.

u: control input vector.

For example, The thruster force τ is calculated by equation 2.10 and 2.11, which takes into account two main propellers, two azimuth thruster, and two tunnel thrusters as shown in Figure 2.3 [17].

$$\tau = TKu \quad 2.10$$

$$\overbrace{\begin{bmatrix}\tau_1\\ \tau_2\\ \tau_6\end{bmatrix}}^{\tau} = \overbrace{\begin{bmatrix}1 & 0 & 0 & 1 & 0 & 0 & 1 & 1\\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0\\ 0 & l_1 & l_2 & 0 & l_3 & l_4 & -l_5 & -6\end{bmatrix}}^{T} \overbrace{\begin{bmatrix}K_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & K_1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & K_2 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & K_3 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & K_3 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & K_4 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & K_5 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_6\end{bmatrix}}^{K} \overbrace{\begin{bmatrix}u_{1x}\\ u_{1y}\\ u_2\\ u_{3x}\\ u_{3y}\\ u_4\\ u_5\\ u_6\end{bmatrix}}^{u} \quad 2.11$$

Where,

$l_1, l_2, l_3, l_4, l_5, l_6$ are thruster lever arms.

$K_1, K_2, K_3, K_4, K_5, K_6$ are thruster coefficients.

u_2, u_4 are the control inputs for tunnel thrusters.

u_5, u_6 are the control input for the first and second main propeller.

$u_{1x}, u_{1y}, u_{3x}, u_{3y}$ are the components in x and y of the azimuth thruster control inputs.

2.3 Mathematical Model

The Balchen model is used for studying maritime vessel dynamics. Balchen divided the model for vessel dynamics into two parts. Two models were developed: a low frequency model for wind, water current, and thruster forces, and a high-frequency model for sea wave forces [4].

2.3.1 Balchen's low frequency model

This model is developed by using the Newton's second law which states that, at any given point in time, the net force on a body is equal to its acceleration multiplied by its mass, as well as similarly, the rate at which the body's momentum changes over time.

$$ma = \sum F \qquad 2.12$$

Where,
m: mass of a body [*kg*]
a: acceleration of the body [m/s^2]
F: forces action on the body [*N*]

The low frequency model from Balchen [4] is given bellow from equation *2.13* to equation 2.18 [18].

$$\frac{dx_{su}}{dt} = v_{su} \qquad 2.13$$

$$\frac{dv_{su}}{dt} = -\frac{d_1}{m_1}|v_{su} - v_{c_{su}}|(v_{su} - v_{c_{su}}) + \frac{1}{m_1}(F_{w_{su}} + F_{t_{su}}) + \eta_1 \qquad 2.14$$

$$\frac{dx_{sw}}{dt} = v_{su} \qquad 2.15$$

$$\frac{dv_{sw}}{dt} = -\frac{d_2}{m_2}|v_{sw} - v_{c_{sw}}|(v_{sw} - v_{c_{sw}}) + \frac{1}{m_2}(F_{w_{sw}} + F_{t_{sw}}) + \eta_2 \qquad 2.16$$

$$\frac{d\psi}{dt} = v_\psi \qquad 2.17$$

$$\frac{dv_\psi}{dt} = -\frac{d_3}{m_3}|v_\psi|v_\psi - \frac{d_4}{m_3}|v_{sw} - v_{c_{sw}}|(v_{sw} - v_{c_{sw}}) + \frac{1}{m_3}(N_c + N_w + N_t) + \eta_3 \qquad 2.18$$

Where;

x_{su}: position in surge [m].

x_{sw}: position in sway [m].

ψ: heading in yaw [rad].

v_{su}: $velocity\ ins\ urge[m/s]$.

v_{sw}: $velocity\ in\ sway[m/s]$.

v_{ψ}: heading rate [rad/s]

m_1, m_2, m_3: internal coefficient which are constants.

$F_{t_{su}}, F_{t_{sw}}, N_t$: thrust force in surge, sway and thrust moment in yaw.

$F_{w_{su}}, F_{w_{sw}}, N_w$: wind force in surge, sway and wind moment in yaw.

η_1, η_2, η_3: are assumed zero mean gaussian white noise process.

$d_i, i = 1,2,3,4$ are drag and momentum coefficients.

Parameter values for $m_1, m_2, and\ m_3$ can be found in Table 2.1 and taken from Balchen paper [4]. Also, windage area of head wind (A_F)and beam wind (A_L), overall length (L) of the vessel is necessary to calculate the wind forces in surge, sway direction and wind moment in yaw. All of this information also taken from the same source and listed in the table below.

Table 2.1: Vessel parameters

Parameter	**value**
m_1	$2.4 \times 10^7[kg]$
m_2	$4 \times 10^7[kg]$
m_3	$4.5 \times 10^{10}[kg.m^2]$
A_F	$500[m^2]$
A_F	$1100[m^2]$
L	$73.2[m]$
d_1	$5 \times 10^{-5}\left[N/(m/s)^2\right]$
d_2	$22 \times 10^{-5}\left[N/(m/s)^2\right]$

d_3	$12 \times 10^{11} \left[N.m / (m/rad)^2 \right]$
d_4	$225 \times 10^{-15} \left[N.m / (m/s)^2 \right]$

2.3.2 Reduced model

For this study, eliminating the drag and momentum coefficient from the non-linear low frequency model the equations 2.13 to 2.18 is used to find the reduced model.

$$\frac{dx_{su}}{dt} = v_{su} \tag{2.19}$$

$$\frac{dv_{su}}{dt} = \frac{1}{m_1}\left(F_{w_{su}} + F_{t_{su}}\right) + \eta_1 \tag{2.20}$$

$$\frac{dx_{sw}}{dt} = v_{su} \tag{2.21}$$

$$\frac{dv_{sw}}{dt} = \frac{1}{m_2}\left(F_{w_{sw}} + F_{t_{sw}}\right) + \eta_2 \tag{2.22}$$

$$\frac{d\psi}{dt} = v_{\psi} \tag{2.23}$$

$$\frac{dv_{\psi}}{dt} = (N_w + N_t) + \eta_3 \tag{2.24}$$

After modifying the equations, the model no longer includes drag and momentum coefficients. The drag and momentum coefficients include additional nonlinearities and coupling factors that impact the overall behavior of the system. The model's behavior now resembles that of a three double integrator. It is a fundamental property of an integrator that if the input to a system is constant or changes slowly, the state variables will increase over time if they are not constrained or controlled. Furthermore, while this simplified model incorporates the fundamental kinematics of the system, it fails to compensate for the dynamic effects of water that are present in the original model. Afterwards, the reduced non-linear model is linearized and different controls are implemented to observe the model's behavior. Appendix B shows the MATLAB code for the reduced model. Appendix C shows the MATLAB code for the non-linear Balchen model.

3 Methods

This chapter provides a study of the methods employed in the experiment for controlling the vessel. To begin with, the chapter presents the linearized model of the reduced Balchen model. It is beneficial for implementing the linearized model to various control systems. Finally, this chapter covers several control algorithms, including standard Model Predictive Control (MPC), Reduced-size MPC, Simple MPC with integral action, and LQ optimum control with integral action.

3.1 Linearization of the mathematical model

Linearizing a nonlinear model helps to facilitate the understanding of the complex system more easily because it simulates a process within limited operating ranges. To make the nonlinear model linear Taylor series expansion was used [18] [19]. Equation 2.19 to 2.24 used to linearized the model. Linear model equation of the system is,

$$\dot{\delta x} = A_c \delta x + B_c \delta u \tag{3.1}$$

$$\delta y = D_c \delta x \tag{3.2}$$

Where,

$$\delta x = x - x_{op} \tag{3.3}$$

$$\delta u = u - u_{op} \tag{3.4}$$

$$\delta y = y - y_{op} \tag{3.5}$$

$$A_C = \left.\frac{\partial f}{\partial x^T}\right|_{x_{op},u_{op}} \tag{3.6}$$

$$B_C = \left.\frac{\partial f}{\partial u^T}\right|_{x_{op},u_{op}} \tag{3.7}$$

$$D_C = \left.\frac{\partial g}{\partial x^T}\right|_{x_{op},u_{op}} \tag{3.8}$$

For this system MIMO system, there is three inputs, three outputs and six states. Input, output and state vectors are given below.

$$x = \begin{bmatrix} x_{su} \\ x_{sw} \\ \psi \\ v_{su} \\ v_{sw} \\ v_{\psi} \end{bmatrix} \quad 3.9$$

$$u = \begin{bmatrix} F_{t_{su}} \\ F_{t_{sw}} \\ N_t \end{bmatrix} \quad 3.10$$

$$y = \begin{bmatrix} x_{su} \\ x_{sw} \\ \psi \end{bmatrix} \quad 3.11$$

After applying the Taylor series, the six different states are specified by equations 3.12 to 3.17.

$$f_1 = \frac{dx_{su}}{dt} = v_{su} \quad 3.12$$

$$f_2 = \frac{dv_{su}}{dt} = \frac{1}{m_1}\left(F_{w_{su}} + F_{t_{su}}\right) + \eta_1 \quad 3.13$$

$$f_3 = \frac{dx_{sw}}{dt} = v_{su} \quad 3.14$$

$$f_4 = \frac{dv_{sw}}{dt} = \frac{1}{m_2}\left(F_{w_{sw}} + F_{t_{sw}}\right) + \eta_2 \quad 3.15$$

$$f_5 = \frac{d\psi}{dt} = v_{\psi} \quad 3.16$$

$$f_6 = \frac{dv_{\psi}}{dt} = (N_w + N_t) + \eta_3 \quad 3.17$$

The matrices A_C, B_C, D_C calculated as shown in equation 3.18 to 3.20. The results after solving the matrices A_C, B_C given in section 4.1.2.

$$A_C = \begin{bmatrix} \frac{\partial F_1}{\partial x_{su}} & \frac{\partial F_1}{\partial x_{sw}} & \frac{\partial F_1}{\partial \psi} & \frac{\partial F_1}{\partial v_{su}} & \frac{\partial F_1}{\partial v_{sw}} & \frac{\partial F_1}{\partial v_{\psi}} \\ \frac{\partial F_2}{\partial x_{su}} & \frac{\partial F_2}{\partial x_{sw}} & \frac{\partial F_2}{\partial \psi} & \frac{\partial F_2}{\partial v_{su}} & \frac{\partial F_2}{\partial v_{sw}} & \frac{\partial F_2}{\partial v_{\psi}} \\ \frac{\partial F_3}{\partial x_{su}} & \frac{\partial F_3}{\partial x_{sw}} & \frac{\partial F_3}{\partial \psi} & \frac{\partial F_3}{\partial v_{su}} & \frac{\partial F_3}{\partial v_{sw}} & \frac{\partial F_3}{\partial v_{\psi}} \\ \frac{\partial F_4}{\partial x_{su}} & \frac{\partial F_4}{\partial x_{sw}} & \frac{\partial F_4}{\partial \psi} & \frac{\partial F_4}{\partial v_{su}} & \frac{\partial F_4}{\partial v_{sw}} & \frac{\partial F_4}{\partial v_{\psi}} \\ \frac{\partial F_5}{\partial x_{su}} & \frac{\partial F_5}{\partial x_{sw}} & \frac{\partial F_5}{\partial \psi} & \frac{\partial F_5}{\partial v_{su}} & \frac{\partial F_5}{\partial v_{sw}} & \frac{\partial F_5}{\partial v_{\psi}} \\ \frac{\partial F_6}{\partial x_{su}} & \frac{\partial F_6}{\partial x_{sw}} & \frac{\partial F_6}{\partial \psi} & \frac{\partial F_6}{\partial v_{su}} & \frac{\partial F_6}{\partial v_{sw}} & \frac{\partial F_6}{\partial v_{\psi}} \end{bmatrix}_{x_{op},u_{op}} \quad 3.18$$

$$B_c = \begin{bmatrix} \frac{\partial F_1}{\partial F_{t_{su}}} & \frac{\partial F_1}{\partial F_{t_{sw}}} & \frac{\partial F_1}{\partial N_t} \\ \frac{\partial F_2}{\partial F_{t_{su}}} & \frac{\partial F_2}{\partial F_{t_{sw}}} & \frac{\partial F_2}{\partial N_t} \\ \frac{\partial F_3}{\partial F_{t_{su}}} & \frac{\partial F_3}{\partial F_{t_{sw}}} & \frac{\partial F_3}{\partial N_t} \\ \frac{\partial F_4}{\partial F_{t_{su}}} & \frac{\partial F_4}{\partial F_{t_{sw}}} & \frac{\partial F_4}{\partial N_t} \\ \frac{\partial F_5}{\partial F_{t_{su}}} & \frac{\partial F_5}{\partial F_{t_{sw}}} & \frac{\partial F_5}{\partial N_t} \\ \frac{\partial F_6}{\partial F_{t_{su}}} & \frac{\partial F_6}{\partial F_{t_{sw}}} & \frac{\partial F_6}{\partial N_t} \end{bmatrix}_{x_{op},u_{op}} \quad 3.19$$

$$D_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad 3.20$$

The operation point x_{op}, u_{op} is chosen to be same as the vessel's center of gravity in the body coordinate system. The state space model in equation 3.1 and 3.2 is discretized as shown below.

$$x_{k+1} = Ax_k + Bu_k + BFw_k \quad 3.21$$

$$y_k = Dx_k \quad 3.22$$

3.2 Model Predictive Control (MPC)

Two types of MPC are used for this experiment. First one is a Standard MPC and later one is reduced size MPC. Reduced-size MPC elements most of the unnecessary calculations (mostly matrices with zero elements) that may present in the Standard MPC. This helps the optimization problem less complex.

3.2.1 Standard MPC

The main objective of Model Predictive Control (MPC) is to reduce the variation between the system's output and the intended reference point. The objective function that used to reduce the system error is given below in equation 3.23 [19].

$$min\, J \;=\; \frac{1}{2}\sum_{k=1}^{N} e_k^T\, Q_k e_k \;+\; u_{k-1}^T P_{k-1}\, u_{k-1} \qquad 3.23$$

Subjected to,

$$x_{k+1} = Ax_k + Bu_k + BFw_k \qquad 3.24$$

$$y_k = Dx_k \qquad 3.25$$

$$e_k = r_k - y_k \qquad 3.26$$

Q_k: Weighting matrix for e_k . It is a diagonal matrix with n_y number of weighting elements on the diagonal corresponding to each output.

p_k: Weighting matrix for u_k . It is a diagonal matrix with n_u number of weighting elements on the diagonal corresponding to each input.

In order to solve the MPC problem, the equation 3.23 is re-formulated to a standard equation using the Kronecker products.

$$J = \frac{1}{2}\, z^T H z + c^T z \qquad 3.27$$

Subjected to,

$A_e z = b_e$ is an equality constraint

$A_i z \;\leq\; b_i$ is an inequality constraint

The bounds are $z_L \;\leq\; z \;\leq\; z_u$

$z = [n_u, n_x, n_e, n_y]^T$, where z is the vector of unknowns

the total number of the unknows (n_z) in vector *z* for the whole prediction length (N) is,

$n_z = N(n_u + n_x + n_y + n_y)$

Now expending the objective function from *K* =1 to *K* =*N* for the equation 3.23,

$$J = \frac{1}{2}\,[e_1^T Q_1 e_1 + e_2^T Q_2 e_2 + \cdots + e_N^T Q_N e_N + u_0^T P_0\, u_{0+} u_1^T P_1\, u_1 + \cdots + u_{N-1}^T P_{N-1}\, u_{N-1}] \quad 3.28$$

Now comparing this equation to 3.27,

$$\mathrm{J} = \frac{1}{2} \overbrace{\begin{bmatrix} u \\ x \\ e \\ y \end{bmatrix}^T}^{z^T} \overbrace{\begin{bmatrix} H_{11} & 0 & 0 & 0 \\ 0 & H_{22} & 0 & 0 \\ 0 & 0 & H_{33} & 0 \\ 0 & 0 & 0 & H_{44} \end{bmatrix}}^{H} \overbrace{\begin{bmatrix} u \\ x \\ e \\ y \end{bmatrix}}^{Z} + \overbrace{\begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix}^T}^{c^T} \overbrace{\begin{bmatrix} u \\ x \\ e \\ y \end{bmatrix}}^{Z} \quad 3.29$$

By multiplying matrices from 3.29,

$$J = \frac{1}{2}\,[u^T H_{11} u + x^T H_{22} x + e^T H_{33} e + y^T H_{44} y + C_1^T u + C_2^T x + C_3^T y + C_4^T u] \quad 3.30$$

Comparing equation 3.28 and 3.30,

$$u^T H_{11} u = u_0^T P_0\, u_{0+} u_1^T P_1\, u_1 + \cdots + u_{N-1}^T P_{N-1}\, u_{N-1} \quad 3.31$$

$$e^T H_{33} e = e_1^T Q_1 e_1 + e_2^T Q_2 e_2 + \cdots + e_N^T Q_N \delta e_N \quad 3.32$$

From the equation 3.31 and 3.32 the *H* matrices (Hessian matrix) now can be calculated.

$$H_{11} = \begin{bmatrix} P_1 & 0 & \cdots & 0 \\ 0 & P_2 & \cdots & 0 \\ 0 & \cdots & P_3 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & P_N \end{bmatrix} = I_N \otimes P \quad 3.33$$

$$H_{33} = \begin{bmatrix} Q_0 & 0 & \cdots & 0 \\ 0 & Q_1 & \cdots & 0 \\ 0 & \cdots & Q_2 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & Q_N \end{bmatrix} = I_N \otimes \mathrm{Q} \quad 3.34$$

Where, $\otimes$ is the Kronecker product and I_N is the identity matrix with the size of '*N*'.

$$H_{22} = I_N \otimes 0_{n_x .\, n_x} \quad 3.35$$

$$H_{44} = I_N \otimes 0_{n_y .\, n_y} \quad 3.36$$

From the equation 3.30, it's clearly seen that there is no linear item, so all the matrices for *C* is zero.

$$C = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix} = \begin{bmatrix} 0_{N.n_u} \\ 0_{N.n_x} \\ 0_{N.n_y} \\ 0_{N.n_y} \end{bmatrix} = 0_{(n_z \times 1)} \qquad 3.37$$

Now, the equality constrains needs to be expressed from the equation 3.27. The matrix A_e and vector b_e can be written as follows,

$$\overbrace{\begin{bmatrix} A_{e,1u} & A_{e,1x} & A_{e,1e} & A_{e,1y} \\ A_{e,2u} & A_{e,2x} & A_{e,2e} & A_{e,2y} \\ A_{e,3u} & A_{e,3x} & A_{e,3e} & A_{e,3y} \end{bmatrix}}^{A_e} \overbrace{\begin{bmatrix} u \\ x \\ e \\ y \end{bmatrix}}^{z} = \overbrace{\begin{bmatrix} b_{e,1} \\ b_{e,2} \\ b_{e,3} \end{bmatrix}}^{b_e} \qquad 3.38$$

Each row of A_e in equation *3.38* corresponding to each equality constraint from equation *3.24* to *3.26*.

After solving each row for the equality constrains, the following equations are formed,

$$A_{e,1u} = \begin{bmatrix} -B & 0 & \cdots & 0 \\ 0 & -B & \cdots & 0 \\ 0 & \cdots & -B & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & -B \end{bmatrix} = -I_N \otimes B \qquad 3.39$$

$$A_{e,1x} = \begin{bmatrix} I & 0 & \cdots & 0 & 0 \\ -A & I & \cdots & 0 & 0 \\ 0 & -A & I & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & -A & I \end{bmatrix} = I_{N.n_x} - (I_{N-1} \otimes A) \qquad 3.40$$

$$A_{e,1e} = 0_{N.n_x \times N.n_y} \qquad 3.41$$

$$A_{e,1y=} 0_{N.n_x \times N.n_y} \qquad 3.42$$

$$b_{e,1} = \begin{bmatrix} Ax_0 + BFw_1 \\ BFw_2 \\ \vdots \\ \vdots \\ BFw_n \end{bmatrix}_{N.n_x \times 1} \qquad 3.43$$

For the next row,

$$A_{e,2u} = 0_{N.n_y \times N.n_u} \tag{3.44}$$

$$A_{e,2x} = \begin{bmatrix} C & 0 & \cdots & 0 \\ 0 & C & \cdots & 0 \\ 0 & \cdots & C & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & C \end{bmatrix} = I_N \otimes C \tag{3.45}$$

$$A_{e,2e} = 0_{N.n_y \times N.n_y} \tag{3.46}$$

$$A_{e,2y} = \begin{bmatrix} I & 0 & \cdots & 0 \\ 0 & I & \cdots & 0 \\ 0 & \cdots & I & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & I \end{bmatrix} = I_{N.n_y} \tag{3.47}$$

$$b_{e,2} = \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}_{N.n_y \times 1} = 0_{N.n_y \times 1} \tag{3.48}$$

The final row,

$$A_{e,3u} = 0_{N.n_y \times N.n_u} \tag{3.49}$$

$$A_{e,3x} = 0_{N.n_y \times N.n_x} \tag{3.50}$$

$$A_{e,3e} = \begin{bmatrix} I & 0 & \cdots & 0 \\ 0 & I & \cdots & 0 \\ 0 & \cdots & I & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & I \end{bmatrix} = I_{N.n_y} \tag{3.51}$$

$$A_{e,3\delta y} = \begin{bmatrix} I & 0 & \cdots & 0 \\ 0 & I & \cdots & 0 \\ 0 & \cdots & I & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & I \end{bmatrix} = I_{N.n_y} \tag{3.52}$$

$$b_{e,3} = \begin{bmatrix} r_1 \\ \vdots \\ \vdots \\ r_N \end{bmatrix}_{N.n_y \times 1} \quad 3.53$$

So now putting all the solved matrices to the question 3.38,

$$\overbrace{\begin{bmatrix} -I_N \otimes B & I_{N.n_x} - (I_{N-1} \otimes A) & 0_{N.n_x \times N.n_y} & 0_{N.n_x \times N.n_y} \\ 0_{N.n_y \times N.n_u} & I_N \otimes C & 0_{N.n_y \times N.n_y} & I_{N.n_y} \\ 0_{N.n_y \times N.n_u} & 0_{N.n_y \times N.n_x} & I_{N.n_y} & I_{N.n_y} \end{bmatrix}}^{A_e} \overbrace{\begin{bmatrix} u \\ x \\ e \\ y \end{bmatrix}}^{z}$$

$$= \begin{bmatrix} \overbrace{Ax_0 + BFw_1}^{b_e} \\ \vdots \\ \vdots \\ BFw_N \\ \cdots \\ 0 \\ \vdots \\ \vdots \\ 0 \\ \cdots \\ r_1 \\ \vdots \\ \vdots \\ r_N \end{bmatrix} \quad 3.54$$

After getting the values for H, C, A_e, b_e matrices the standard formulation is completed for the purpose of the analysis. These values are fundamental components for designing of the linear MPC control system. The *quadprog.m* solver in MATLAB is used to solve the MPC problem. The algoritm for the Standard MPC included in Appendix F. The results of the simulation are documented in section 4.2.1.

3.2.2 Reduced size MPC

The objective function of the Reduced-size MPC is formulated by substituting $e_k = r_k - y_k$ to the equation 3.23 gives [18] [19],

$$min\, J = \frac{1}{2} \sum_{k=1}^{N} (r_k - y_k)^T Q_k (r_k - y_k) + u_{k-1}^T P_{k-1} u_{k-1} \quad 3.55$$

$$min\, J = \frac{1}{2} \sum_{k=1}^{N} ({y_k}^T Q_k y_k + u_{k-1}^T P_k u_{k-1}) - 2(Q_k r_k)^T) y_k + {r_k}^T Q_k r_k \quad 3.56$$

As Q is a symmetric matrix, so $Q_k = {Q_k}^T$. Also, ${r_k}^T Q_k r_k$ is a constant, so the modified objective function is formulated in equation 3.57.

$$min\,J\; = \frac{1}{2}\sum_{k=1}^{N}\left(y_k{}^T\,Q_k\,y_k + u_{k-1}^T P_k\,u_{k-1}\right) - 2(Q_k r_k)^T)\,y_k + r_k{}^T Q_k r_k \qquad 3.57$$

Now, $y_k = Dx_k$ and substituting it to the above equation,

$$min\,J\; = \frac{1}{2}\sum_{k=1}^{N}\left(x_k{}^T\,\tilde{Q}_k\,x_k + u_{k-1}^T P_k\,u_{k-1}\right) - 2(D^T Q_k\,r_k)^T\,x_k \qquad 3.58$$

Where, $\tilde{Q}\; = D^T\,QD$

By writing the equation to the standard form $z^T Hz + c^T z$,

$$J = \overbrace{\begin{bmatrix}u_{k-1}\\ x_k\end{bmatrix}^T}^{z^T}\overbrace{\begin{bmatrix}H_{11} & 0\\ 0 & H_{22}\end{bmatrix}}^{H}\overbrace{\begin{bmatrix}u_{k-1}\\ x_k\end{bmatrix}}^{Z}+\overbrace{\begin{bmatrix}c_1\\ c_2\end{bmatrix}^T}^{c^T}\overbrace{\begin{bmatrix}u_{k-1}\\ x_k\end{bmatrix}}^{z} \qquad 3.59$$

Now to find the *H* and *C* matrices,

$$H_{11}\; = \begin{bmatrix}P_1 & 0 & \cdots & 0\\ 0 & P_2 & \cdots & 0\\ 0 & \cdots & P_3 & 0\\ \vdots & \ddots & \vdots & \vdots\\ 0 & \cdots & 0 & P_N\end{bmatrix} = I_N \otimes P \qquad 3.60$$

$$H_{22} = \begin{bmatrix}Q_0 & 0 & \cdots & 0\\ 0 & Q_1 & \cdots & 0\\ 0 & \cdots & Q_2 & 0\\ \vdots & \ddots & \vdots & \vdots\\ 0 & \cdots & 0 & Q_N\end{bmatrix} = I_N \otimes Q \qquad 3.61$$

$$C_1 = \begin{bmatrix}0\\ \vdots\\ \vdots\\ 0\end{bmatrix}_{N.n_u \times 1} \qquad 3.62$$

$$C_1 = \begin{bmatrix}D^T\,Q_1 r_1\\ D^T\,Q_2 r_2\\ \vdots\\ D^T\,Q_N r_N\end{bmatrix}_{N\times 1} \qquad 3.63$$

Now for the equality constrains,

$$[A_{eq_u}\quad A_{eq_x}]\begin{bmatrix}u_{k-1}\\ x_k\end{bmatrix} = B_{eq} \qquad 3.64$$

Comparing with equation 3.24,

$$A_{equ} = \begin{bmatrix} -B & 0 & \cdots & 0 \\ 0 & -B & \cdots & 0 \\ 0 & \cdots & -B & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & -B \end{bmatrix} = -I_N \otimes B \qquad 3.65$$

$$A_{eqx} = \begin{bmatrix} I & 0 & \cdots & 0 & 0 \\ -A & I & \cdots & 0 & 0 \\ 0 & -A & I & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & -A & I \end{bmatrix} = I_{N.n_x} - (I_{N-1} \otimes A) \qquad 3.66$$

$$b_{eq} = \begin{bmatrix} Ax_0 + BFw_1 \\ BFw_2 \\ \vdots \\ \vdots \\ BFw_n \end{bmatrix}_{N.n_x \times 1} \qquad 3.67$$

Now putting all the matrices together,

$$\overbrace{\begin{bmatrix} -I_N \otimes B & I_{N.n_x} - (I_{N-1} \otimes A) \end{bmatrix}}^{A_e} \overbrace{\begin{bmatrix} u_{k-1} \\ x_k \end{bmatrix}}^{Z} = \overbrace{\begin{bmatrix} Ax_0 + BFw_1 \\ BFw_2 \\ \vdots \\ \vdots \\ BFw_n \end{bmatrix}_{N.n_x \times 1}}^{b_e} \qquad 3.68$$

Now the size of H, C, A_e, b_e are being reduced. With the help of *quadprog.m* solver in MATLAB Reduced-size MPC is simulated and the results are documented in section 4.2.2.

3.3 LQ optimal control with integral action

LQ optimal control stands for Linear Quadratic optimal control. It's a method employed in control theory for creating feedback controllers for linear systems that are affected by quadratic cost functions. In LQ optimal control, the aim is to minimize a quadratic cost function that accurately represents the balance between control effort and system performance. LQ controller with integral action has similar attributes as a standard PID controller [20]. The discrete model comes with the model disturbances as shown in equation 3.69 and 3.70.

$$x_{k+1} = Ax_k + Bu_k + w_k \qquad 3.69$$

$$y_k = Dx_k + v_k \qquad 3.70$$

The LQ problem of a discrete model for infinite time horizon and with integral action given by equation 3.71,

$$J_i = \frac{1}{2}\sum_{k=1}^{\infty}(y_k - r)^T Q_k (y_k - r) + \Delta u_k^T P_k u_k \quad 3.71$$

Where, *r*: is the reference point

Q and *P*: Weighting matrices

For solving the quadratic problem, it must be independent from the disturbances w_k and measurement noise v_k. So, the new state space model can be written as,

$$\Delta x_{k+1} = A\Delta x_k + B\Delta u_k \quad 3.72$$

$$y_k = y_{k-1} + D\Delta x_k \quad 3.73$$

From [20], it shows that the state space model can be written as a augmented model as like the equation 3.74 and 3.75,

$$\begin{bmatrix}\Delta x_{k+1}\\ y_k - r\end{bmatrix} = \begin{bmatrix}A & 0_{n\times m}\\ D & I_{m\times m}\end{bmatrix}\begin{bmatrix}\Delta x_k\\ y_{k-1} - r\end{bmatrix} + \begin{bmatrix}B\\ 0_{m\times r}\end{bmatrix}\Delta u_k \quad 3.74$$

$$y_{k-1} - r = \begin{bmatrix}D & I_{m\times m}\end{bmatrix}\begin{bmatrix}\Delta x_k\\ y_{k-1} - r\end{bmatrix} \quad 3.75$$

Using the augmented model, the new model equations are formed in 3.76 and 3.77.

$$\tilde{x}_{k+1} = \tilde{A}\tilde{x}_k + \tilde{B}\Delta u_k \quad 3.76$$

$$\tilde{y}_k = \tilde{D}\tilde{x}_k \quad 3.77$$

By minimizing the objective function with respect to the deviation Δu_k the quadratic problem can be solved. The optimal control deviation of Δu_k is given in equation 3.78.

$$\Delta u_k = \begin{bmatrix}G_1 & G_2\end{bmatrix}\begin{bmatrix}\Delta x_k\\ y_{k-1} - r\end{bmatrix} \quad 3.78$$

$$u_k = u_{k-1} + G_1\Delta x_k + G_2(y_{k-1} - r_k) \quad 3.79$$

The control input at time K can be calculated from the previous control signal. $G_1\Delta x_k$ is related to the state deviation and $G_2(y_{k-1} - r_k)$ deviation between output and the setpoint.

The optimal feedback matrix *G* is calculated using the *dlqdu_pi* MATLAB function [20], and documented in Appendix H. Next this function used to develop the code that shows the LQ optimal controller performance in surge, sway and yaw direction. Also, the input signal is clipped to see the performance of the controller with limited control signal. All the results are

documented in section 4.3. The MATLAB code that uses to control the reduce model given in Appendix I.

3.4 Simple MPC

Simple MPC is developed by David Di Ruscio [16] is a controller that is usefully to a slow varying system and measurement trends. The MPC algorithm also can use integral action to reduce the control error faster. The state space model for this model is given below,

$$x_{k+1} = Ax_k + Bu_k \quad 3.80$$

$$y_k = Dx_k \quad 3.81$$

$$y_{k+1/L} = F_L u_{k/L} + p_L \quad 3.82$$

Where:

L is the prediction horizon.

F_L is a constant matrix ($F_L = [O_L B \quad {H_L}^d]$).

O_L: The observability matrix of the pair A, D.

${H_L}^d$: is the Toeplitz matrix of the impulse response matrices $DA^{i-1}B$.

$O_L, O_L B \; and \; {H_L}^d$ is calculated using *ss2h.m* function in MATLAB. The function is documented in Appendix L and developed by David Di Ruscio.

$$p_L = O_L A x_k \quad 3.83$$

So, the modified objective function to be minimized is given by equation

$$J_k = \frac{1}{2} (y_{k+1/L} - r_{k+1/L})^T Q_k (y_{k+1/L} - r_{k+1/L}) + \Delta u_{k/L}^T P u_{k/L} \quad 3.84$$

By substituting the $y_{k+1/L}$ from equation 3.82 to the objective function equation 3.85.

$$J_k = \frac{1}{2} {u_{k/L}}^T ({F_L}^T Q F_L + P) u_{k/L} + 2 ({F_L}^T Q (p_L - r_{k+1/L})) u_{k/L} \quad 3.85$$

The objective function can be compared with the standard objective function given in equation 3.86.

$$J_k = \frac{1}{2} {u_{k/L}}^T H u_{k/L} + C^T u_{k/L} \quad 3.86$$

So, the value of H and C is,

$$H = F_L{}^T Q\ F_L\ +\ P \qquad 3.87$$

$$C = 2\ (\ F_L{}^T Q(p_L\ -\ r_{k+1/L})) \qquad 3.88$$

The *quadprog.m* function in MATLAB is used to solve the optimization problem. The results are documented in 4.4.1. The MATLAB code for Simple MPC to control the vessel is given in Appendix J.

For the Simple MPC with integral action $\tilde{x}_k\ =\ \begin{bmatrix}\Delta x_k \\ y_{k-1}\end{bmatrix}$ is used to calculate the p_L. The control inputs are thruster forces and moments. The input thruster values can be assumed as inequality constrains as shown from equation 3.89 to 3.91.

$$u_{min} \leq u \leq u_{max} \qquad 3.89$$

$$u \leq u_{max} \qquad 3.90$$

$$-u \leq -u_{min} \qquad 3.91$$

Now from [16], u_k can be written in a relationship with Δu_k in equation 3.92

$$u_k\ =\ S\Delta u_k + \mathrm{c}u_{k-1} \qquad 3.92$$

Where $S\epsilon\mathbb{R}^{n_u.L\times\ n_u.L}$ and $C\epsilon\mathbb{R}^{n_u.L\times\ n_u}$. These are the matrices that contains ones and zeros. This matrices are calculated by *scmat.m* function developed by David Di Ruscio given in Appendix M.

The main purpose is to optimize the control increment Δu. So, substituting it to the inequality constrains gives,

$$S\Delta u \leq u_{max}\ -\ \mathrm{c}u_{k-1} \qquad 3.93$$

$$-S\Delta u \leq -u_{min}\ +\ \mathrm{c}u_{k-1} \qquad 3.94$$

From the inequality constrains from above, the standard from is written in equation

$$A_i\Delta u \leq B_i \qquad 3.95$$

Where, $A_i\ =\ \begin{bmatrix}S \\ -S\end{bmatrix}$ and $B_i\ =\ \begin{bmatrix}u_{max}\ -\ \mathrm{c}u_{k-1} \\ -u_{min}\ +\ \mathrm{c}u_{k-1}\end{bmatrix}$. Both of this used directly to the *quadprog.m* function. The MATLAB code for Simple MPC with integral action to control the vessel is given in Appendix K. The results of Simple MPC with integral action is documented in 4.4.2.

3.5 Kalman Filter as a state estimator

A common method for estimating the values of unknown state variables in dynamic systems is the Kalman Filter. The Kalman Filter technique was initially designed for systems that are expected to be represented by a linear state space model. However, the system model is often nonlinear in numerous applications. Furthermore, the linear model can be regarded as a specific case of a nonlinear model [21]. In this report the discrete time Kalman filter is used. This Kalman filter corrects the predictions, So, it has a predictor part and a corrector part. The Kalman filter equation is given in equation 3.96. $\bar{x}_{k+1}$ predicts the states using the vessel linear model as given in equation 3.97[17] [18].

$$\hat{x}_{k+1} = \bar{x}_{k+1} + K\,(y_k - \hat{y}_k) \qquad 3.96$$

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k + BF_w \qquad 3.97$$

For the experiment Kalman Filter gain is calculated using the *dlqe.m* MATLAB function from the discrete time system given in equations 3.98 to 3.99 [18].
This function takes *A,G,D,Q,R* as inputs and gives *K,P,Z,E* as outputs.

$$x_{k+1} = Ax_k + Bu_k + Gw_k \qquad 3.98$$

$$y_k = Dx_k + v_k \qquad 3.99$$

Where, w_k *and* v_k: Process and Measurement noise.
$Q = E(ww')$: process noise covariance.
$R = E(vv')$*:* measurement noise covariance.
K: Kalman gain.
P: Riccati solution.
Z: Error covariance.
E: Estimator poles.

4 Results

In this chapter the behavior of the reduced model and comparison of the model to the original model is presented. This chapter also gives the detailed discussion of the findings after testing the simplified Balchen model with different controllers, such as the LQ optimum controller, the MPC controller, and the Simple MPC controller. Finally, this chapter documents and evaluates the performance of the controllers using the integral of absolute error (IAE) and the total variation (TV) index.

4.1 Model behavior

The behavior of the simplified model is discussed in this section.

4.1.1 Reduced model

The reduced model is developed in MATLB using the equation from 2.19 to 2.24. This experiment is conducted without taking into account any wind force acting on the vessel. Also, no thruster forces are given on the vessel. Figure 4.1 shows the behavior of the reduced Balchen model. The updated model removes the drag and momentum coefficient, resulting in a simplified version consisting of three double integrators. Also, from the figure it is clearly understandable that if the model is left uncontrolled, state variables will gradually rise without any limitations.

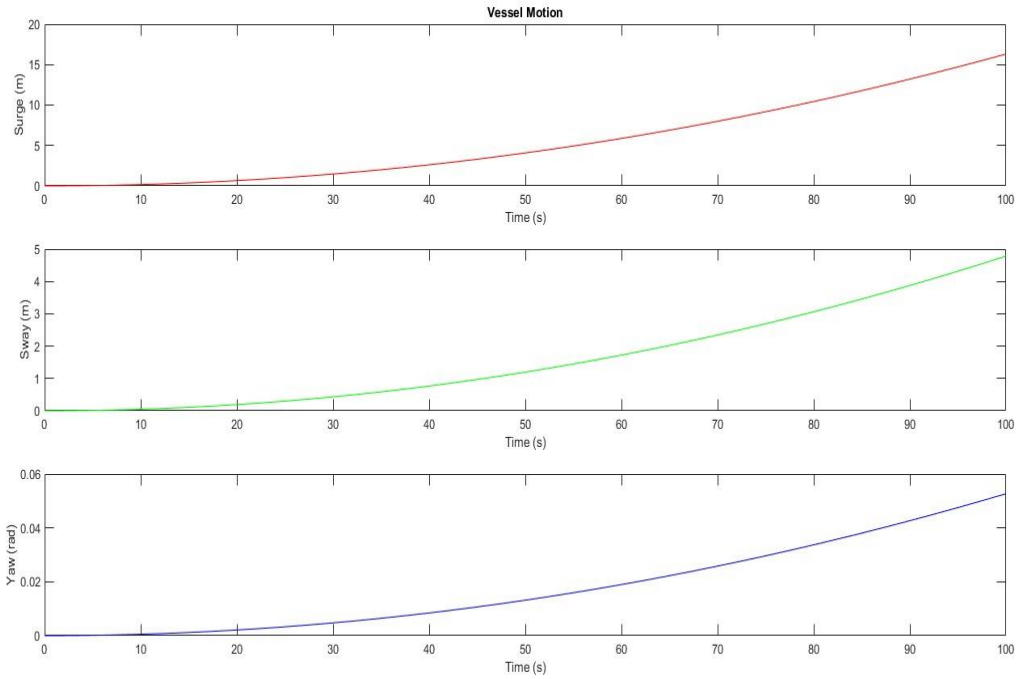

Figure 4.1: Behavior of the reduced Balchen model.

4.1.2 Linearization

The Balchen reduced model has been simplified and transformed into a linear model. Then the two models are simulated in MATLAB to observe their behavior in the presence of environmental disturbances.

The Taylor series expansion is used to find the model matrices A_c, B_c, D_c as shown in equations 4.1 to 4.3.

$$A_c = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad 4.1$$

$$B_c = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{m_1} & 0 & 0 \\ 0 & \frac{1}{m_2} & 0 \\ 0 & 0 & \frac{1}{m_3} \end{bmatrix} \quad 4.2$$

$$D_c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad 4.3$$

For the environmental effect the wind disturbances are generated randomly where wind speed has a value from 0 to 20 *m/s* and the direction of the wind has a value between 0 to 360 *deg.*

Figure 4.2 shows the vessel position in surge under the wind force. The initial subplot indicates that both the linearized and reduce model have identical behavior. The model is not receiving any control signal as shown in the next subplot. The third subplot shows the amount of the wind force acting on the vessel. The wind's velocity and direction are displayed in the fourth and fifth subplots.

The wind velocity remains constant at 18 m/s for the first 4 minutes of the simulation, while the wind direction shifts from approximately 380° to 250°. Both wind velocity and direction continue to decrease until 15 minutes. For this reason, the wind force decreases in the surge position from $6 \times 10^4\ N$ to $-1 \times 10^4\ N$ after 2 minutes of the simulation. As a result, the surge vessel position increases from 0 m to 2000 m after 20 minutes of the simulation.

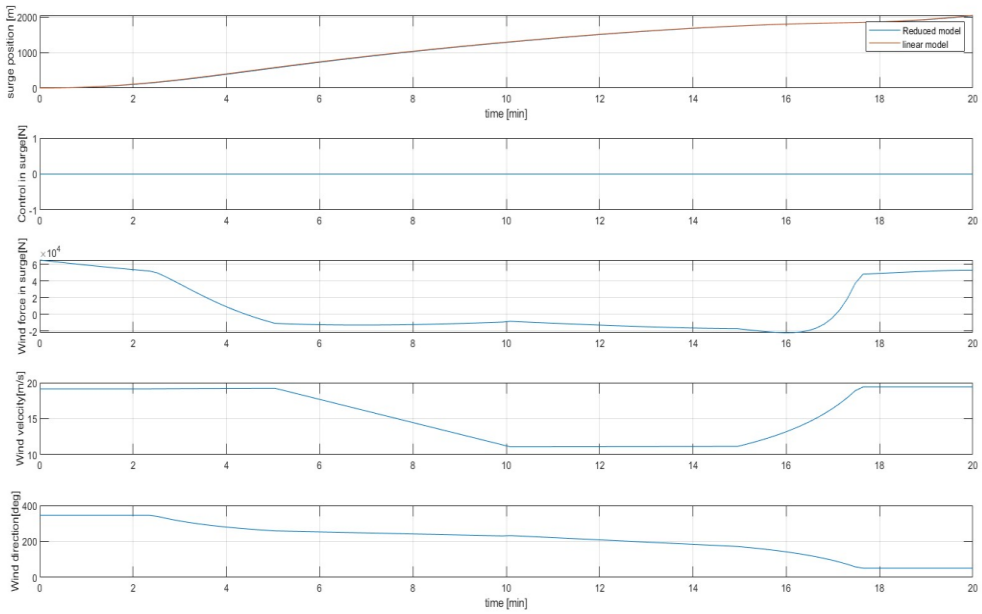

Figure 4.2: Reduced model behavior under wind disturbances in surge direction.

Figure 4.4 shows the vessel's movement in the sway direction under wind disturbances. The first plot shows that behavior of both models the same under wind forces. In the figure, the wind velocity is constant for the first quarter of the simulation. After that, it gradually declines until it reaches a value of 9 *m/s* at the 10-minute mark, then continues with the same velocity until 15 minutes. Additionally, the wind direction shifts from 380° to 200°. As a result, the wind force increases between $-0.2 \times 10^5\ N$ to $0\ N$ during the first 15 minutes of the simulation. During the final 4 minutes of the simulation, both the velocity and the force increase to 20 m/s and $2 \times 10^5\ N$. The gradually increasing wind force is causing a decrease in sway direction from 0 *m* to approximately $-1100\ m$.

Figure 4.3 shows model behavior under the wind moment in yaw. The first subplot shows that the both of the models do not have any behavioral differences. The third subplot shows that the wind moment started at $-1 \times 10^6\ Nm$, and after 3 minutes, it started to increase and stay approximately $0.5 \times 10^6\ Nm\ to\ 0 \times 10^6\ Nm$ until 15 minutes. That causes the deviation in the yaw direction to go from 0° to -100°. For the last 4 minutes of the simulation, the wind direction is changed from 180° to approximately 20°, and the wind moment is increased to $0 \times 10^6\ Nm$. As a result, there is a slight increase of 20° in the yaw direction in the last minute of the simulation.

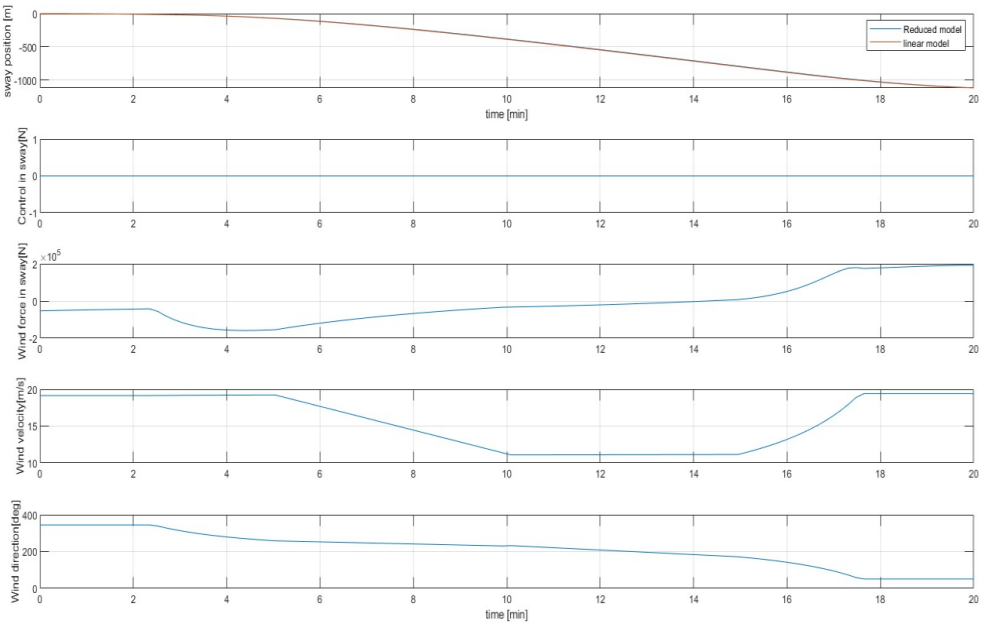

Figure 4.4: Reduced model behavior under wind disturbances in sway direction.

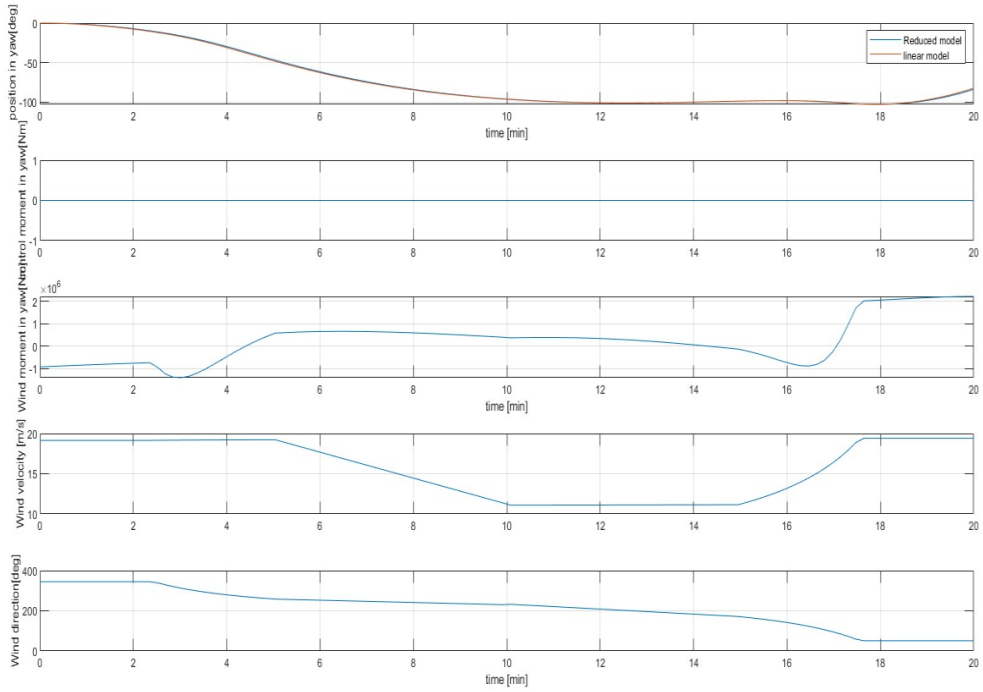

Figure 4.3: Reduced model behavior under wind disturbances in yaw direction.

4.1.3 Comparison between Balchen non-linear low frequency model and reduced model

By eliminating the drag and momentum coefficient, the original model basically functioned as a three double integrator model. The behavior of the original model and the reduce model is simulated in the presence of wind disturbances and without any external input force. Figure 4.5 shows that the Balchen model and the reduced model is behaving exacltly same under disturbances in surge direction. That is also same for in sway and yaw direction as shown in Figure 4.7 and Figure 4.6. This comparison is interesting since the Balchen low-frequency nonlinear model can be referred to as a linear model when the drag and momentum coefficient are removed.

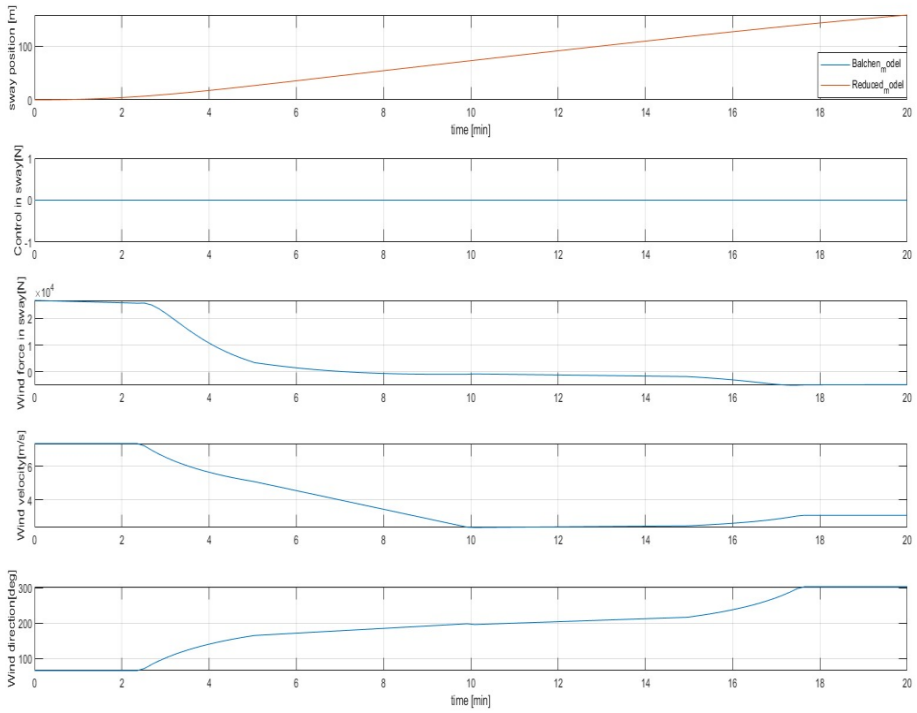

Figure 4.5: Comparison between Balchen and Reduced Balchen model in surge direction.

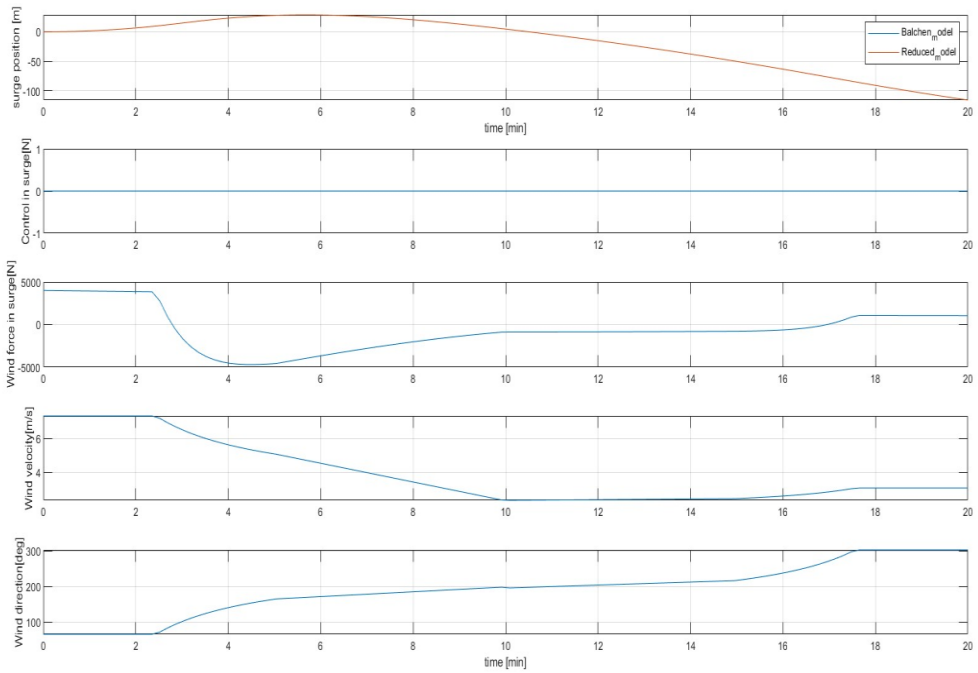

Figure 4.7: Comparison between Balchen and reduced Balchen model in sway direction.

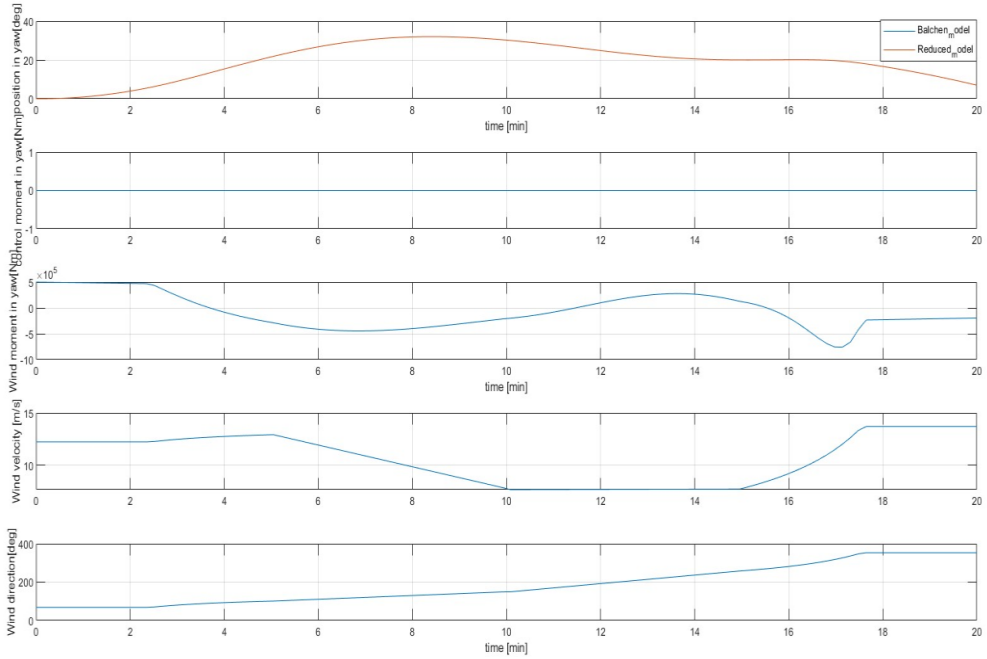

Figure 4.6: Comparison between Balchen and reduced Balchen model in yaw direction.

The reduced model's behavior is almost identical to that of the original model. The experiment's model parameters are based on the 'Seaway Eagle', where drag and momentum coefficients are very small. So, the coefficient reduction did not affect the overall results.

4.1.4 Stability, Controllability and Observability analysis

Control theory is the study of controlling the behavior of dynamic systems. Three fundamental concepts in the field are stability, controllability, and observability. For finding the stability of the system, eigenvalues of the system matrix A of the discrete time model should be located inside the unity circle. Equation 4.4 shows the eigenvalue of the matrix A calculated by using *eig.m* function in MATLAB.

$$eig(A) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{4.4}$$

In the system matrix *A*, each of its eigenvalues equals 1. The system is stable if the eigenvalues have magnitudes that are smaller than 1. However, this system is unstable since at least one eigenvalue is outside the unity circle. This implies that instead of settling down after a disturbance occur, the system's behavior may diverge or oscillate.

The MATLAB *ctrb.m* function is used to determine the system's controllability, and the *rank.m* function is used to determine the rank of the controllability matrix (*A*,*B*). According to the findings, the matrix's rank is 6, which corresponds to the order of the system. This indicates that there are enough control inputs in the system to guide it from any starting condition to any desired end state in a specific period of time. Even with the instability of the system, the simplified model can regulate the behavior of the system using control input.

Using the MATLAB function *obsv.m*, the observability matrix pair (*A*,*D*) is obtained. The rank is also found applying the *rank.m*. The rank is 6, as indicated by the system is observable. So, the system can accurately monitor its states and behavior, regardless of a situation that it shows unstable behavior.

4.2 MPC

This section displays the experimental results from the Standard MPC and Reduced-size MPC.

4.2.1 Standard MPC

Initially, it was assumed that the states were measurable. This means that the system variables may be directly observed or measured, allowing accurate information on their values. The second type of experiment was done by predicting the immeasurable states using Kalman filter.

Experiments with assuming the measurable states

In this experiment, the setpoint of the vessel changes twice in the surge direction. The total duration of the simulation is 20 minutes. During the initial 5 minutes, the set point is at 0 m. For the next 5 minutes, the set point shifts to 15 m. The second shift in the setpoint occurs at the 10-minute mark, when the position decreases to 3 m.

In Figure 4.8 , the wind force began to decrease around 2.3 minutes. The greatest variation happens when it declines from $-0.1 \times 10^4\ N\ to\ -2..5 \times 10^4\ N$ between 11 and 16 minutes. In both cases, controlling the surge results in an increase in the thrust forces to maintain the vessel's position at the aimed set point.

The control input began to increase to $1 \times 10^5\ N$ at 4 minutes, initially to compensate for the wind decrease, and later to reflect changes in the setpoint. This leads to an increase in the vessel's position in the surge by approximately 17 m. After that controller noticed the overshoot and thruster force decreases to $-0.8 \times 10^5\ N$. It takes almost 2 minutes for the controller to reach the steady state. For the second setpoint change the thruster output decrease again to $-0.5 \times 10^5\ N$. From 11 to 20 minutes, the thruster output is increased again to compensate for wind disturbances and keep the vessel's position at 3 m.

For the sway direction, as shown in Figure 4.10, the controller increases the thruster force approximately $4.8 \times 10^5\ N$ at 4 minutes to adjust the setpoint change from 0 m to 23 m. In order to counteract the overshoot, the controller reduces the thruster force once more. It took 3 minutes to reach a stable position. For the second setpoint change, the position shifts from 23 m to 9 m. So, the controller decreases the thruster output again. Between 10 and 17 minutes, the wind is decreased to $-3 \times 10^4\ N$. The controller also addressed this change by increasing the thruster force to maintain the vessel position in sway at 8 m.

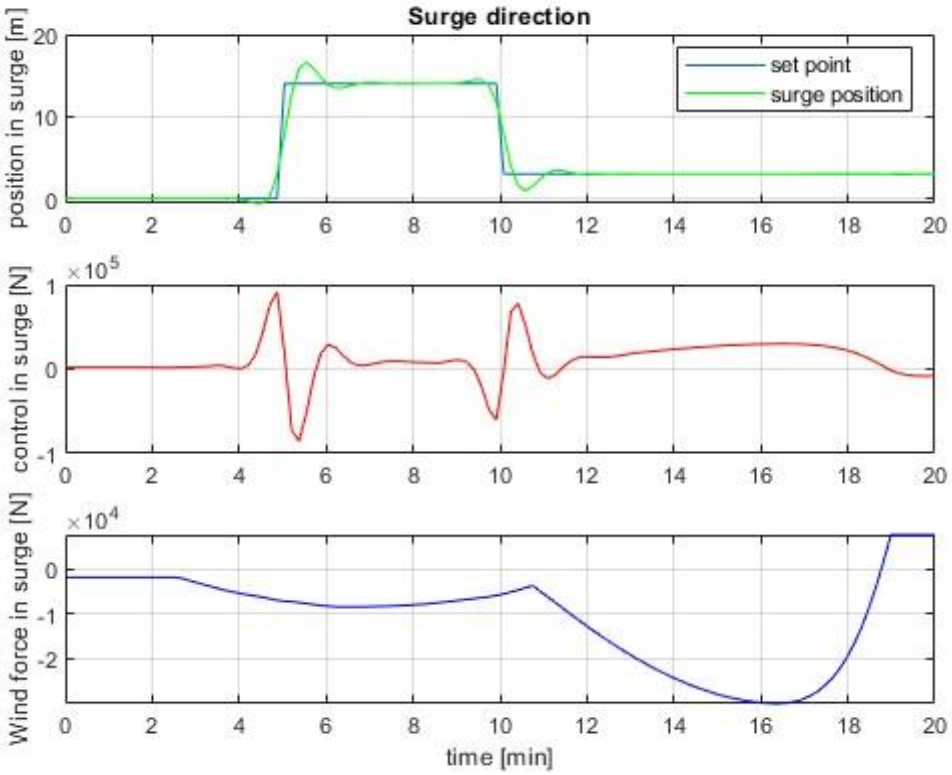

Figure 4.8: Simulation output in surge direction for standard MPC.

The vessel is heading in a yaw direction, as shown in Figure 4.9, there is one setpoint change from 0° to 14° at 5 minutes. To stabilize the vessel's yaw heading, the MPC controller generates thruster moment of $2.2 \times 10^7\ Nm$. To counter the reducing wind and the overshot of 3°, it reduces the thruster moment again. It takes the controller approximately 2.2 minutes to stabilize the heading at 15°.

Figure 4.11 shows the vessel position in the NED coordinate system. At first vessel position is in the origin. After 5 minutes the new setpoint in body coordinate system is (15, 23) *m* and the heading is changed to 14°, which in NED coordinate system is (8.9, 25.9) *m*. The MPC controller correctly move the vessel to the desired location. For the second setpoint change, the heading was same but the position in body coordinate moved to (3,8) *m*. The new North-East

position is (0.9, 8.4) *m* and the controller again successfully moved the vessel into the desired position.

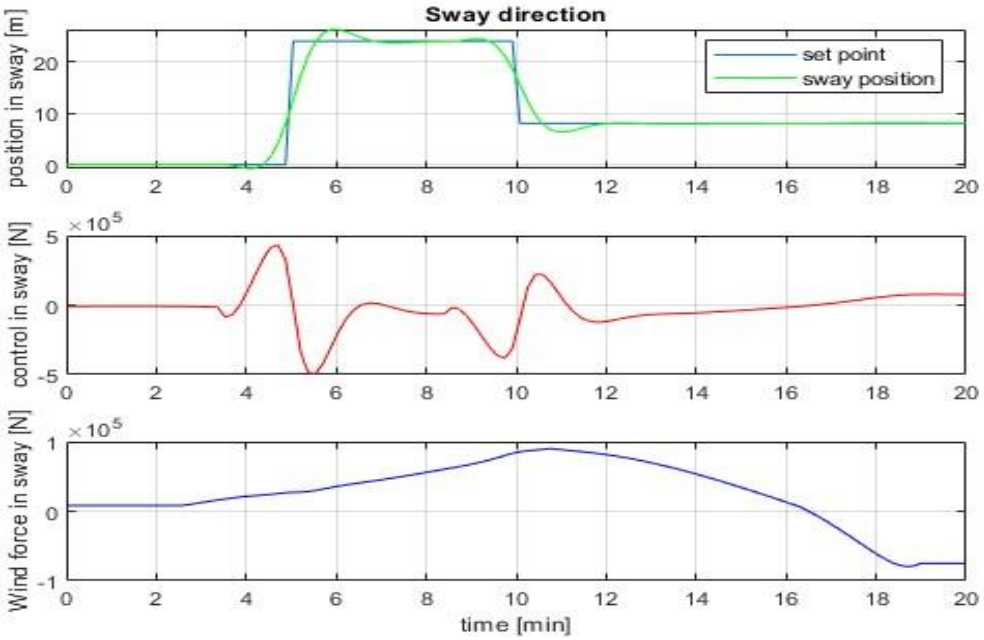

Figure 4.10: Simulation output in sway direction for standard MPC.

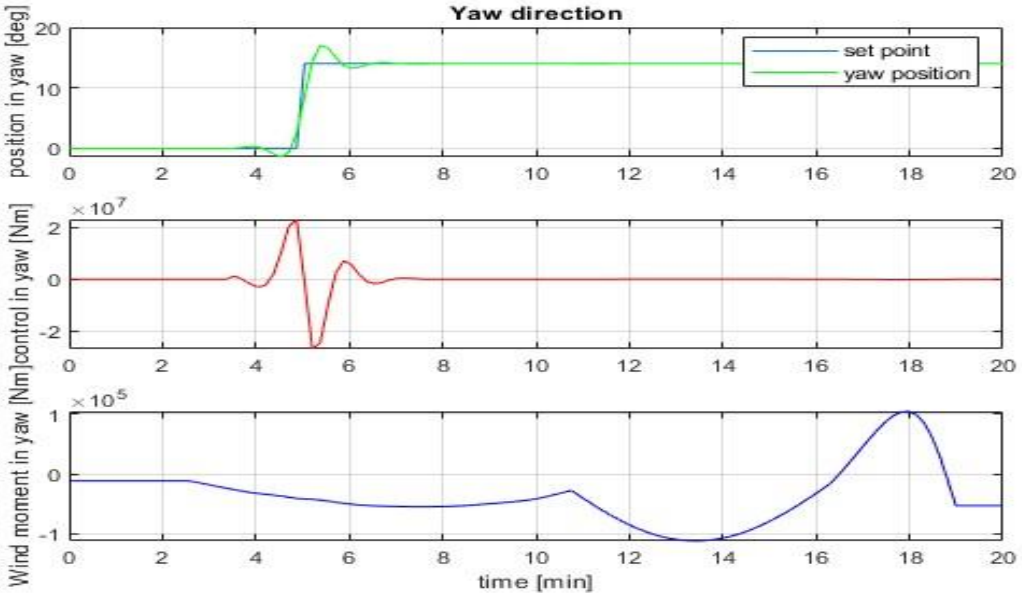

Figure 4.9: Simulation output in yaw direction for standard MPC.

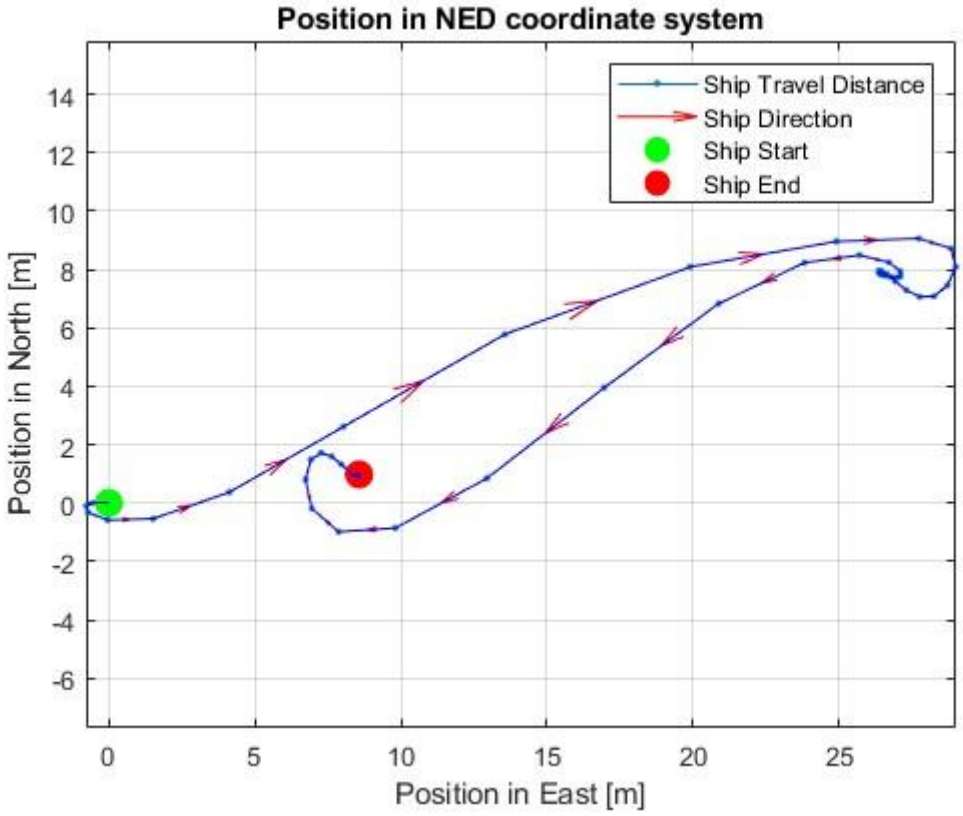

Figure 4.11: Vessel's position in NED coordinate system.

Experiments using Kalman filter to estimate the immeasurable states

For this experiment the prediction of immeasurable states is done by the Kalman filter. Figure 4.12 shows that the MPC controller neatly handles the set point change in 5 and 10 minutes of the simulation by providing sufficient proportional force to increase and decrease the vessel's surge position. Throughout the simulation period, the wind gradually decreases. The controller notices this change and takes appropriate action by steadily increasing the control signal from 11 to 20 minutes to keep the vessel in the desired 3 *m* position.

Comparing Figure 4.8 and Figure 4.12 shows that using Kalman filter decreases the overshoot of the vessel. But the noticeable thing is that it takes slightly more time to come to the stable position.

Figure 4.13 shows that the vessel is effectively adapting to changes in the setpoint. The main difference from Figure 4.10 is that though the overshoot is less, the time to get to the stable position is increased. It took almost 4 minutes when the Kalman filter was used for predicting the states, compared to one minute less when the states were assumed to be known. Additionally, the MPC controller is performing well in steady state, as evidenced by the sudden change in wind force from 16 to 20 minutes into the simulation. The controller increases the control input to keep the vessel in the desired position.

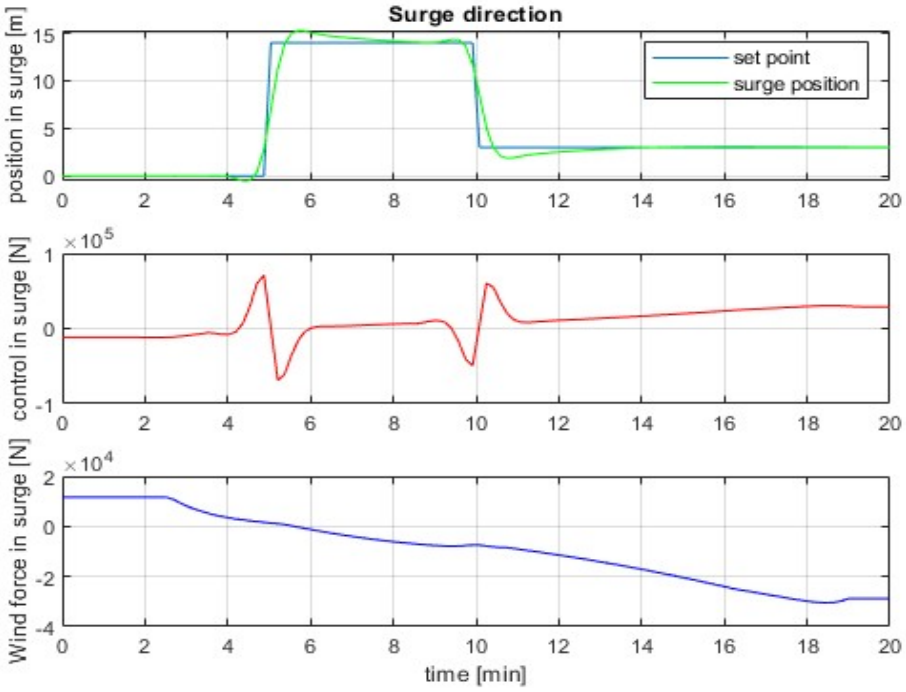

Figure 4.12: Output in surge direction when Kalman filter is used for estimating immeasurable states.

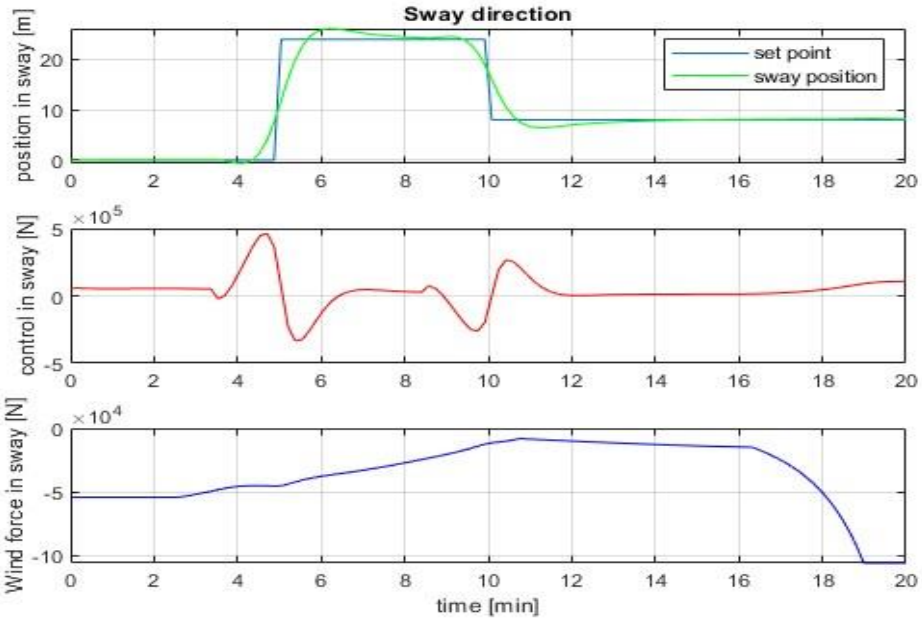

Figure 4.13: Output in sway direction when Kalman filter is used for estimating immeasurable states.

Figure 4.14 demonstrates that, when utilizing the Kalman filter to predict an immeasurable state, the overshoot is almost the same to Figure 4.9 where the states are assumed.

Figure 4.15 shows the vessel's position in the NED coordinate system. The vessel is at the origin for the first 5 minutes; after that, its position in the body coordinate system is (14,24) *m*, and at 10 minutes, it is (3,9) *m*. The heading in body position is 14°. Therefore, in the NED coordinate system, the vessel will first move to (7.7, 26.7) *m*. Then again, move to (0.7, 9.4) *m*. When compared with Figure 4.11, it becomes evident that the vessel motion is smoother and the deviation from the setpoint is less when using the Kalman filter.

4.2.2 Reduced size MPC

Reduced-size MPC reduces unnecessary calculations by directly substituting equality constraints into the objective function. This simplification decreases the amount of computational work required.

For this experiment, the setpoint change occurs twice in the 30-minute simulation. Figure 4.16 shows that, for the first 7.5 minutes, the set point is set to 0 *m*. During the following 7.5 minutes, the surge position changes to 43 *m*. At the 15-minute mark, there is a second change in the setpoint, with the position decreasing to 3 *m*. At 7.5 minute the MPC controller first produces a thrust of $3 \times 10^5\ N$ to move the vessel position in 43 *m*. However, when it encounters an overshoot, it decreases the control input in order to tackle it. The vessel reached the steady position in under 1 minute which gives better than the Standard MPC. In the next set point change, the MPC controller first decreased the proportional force due to the position being shifted to 3 *m*. Also, as the wind started to decrease a small input force of $0.1 \times 10^5\ N$ is applied to keep the vessel in desired position.

Figure 4.17 shows the performance of the MPC controller in the sway direction. The vessel position is in origin before the first setpoint change. The vessel's new position at 7.5 minute is 36 *m* and the controller provided a force of $0.8 \times 10^5\ N$ to reach the positive sway direction. The overshoot also compensated for the both set point change at 7.5 and 15 minutes of the simulation.

The vessel is heading in a yaw direction, as shown in Figure 4.18, there is one setpoint change from 0° to 14° at 7.5 minutes. To stabilize the vessel's yaw heading, the Reduced-size MPC controller generates thruster moment of $1.2 \times 10^7\ Nm$. To counter the reducing wind and the overshot of 2°, it reduces the thruster moment again. It takes the controller approximately 3 minutes to stabilize the heading at 15°.

Figure 4.19 shows the vessel position in the NED coordinate system. At first vessel position is in the origin. After 7.5 minutes the new setpoint in body coordinate system is (43, 36) *m* and the heading is changed to 14°, which in NED coordinate system is (33.1, 45.3) *m*. The MPC controller correctly move the vessel to the desired location. For the second setpoint change, the heading was same but the position in body coordinate moved to (3,8) *m*.

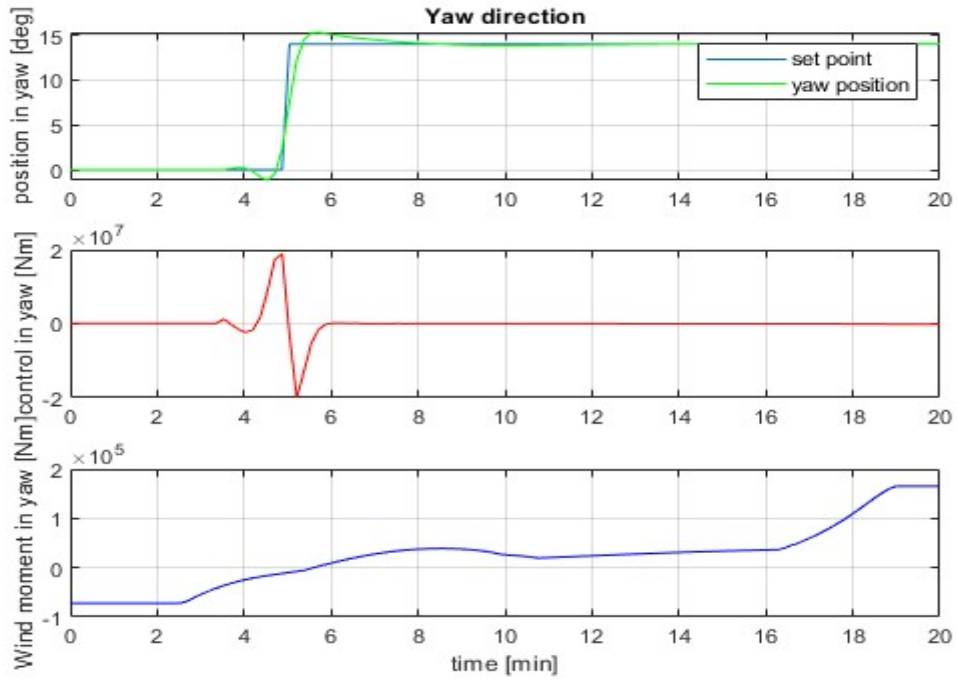

Figure 4.14: Vessel heading in yaw direction when Kalman filter is used for estimating immeasurable states.

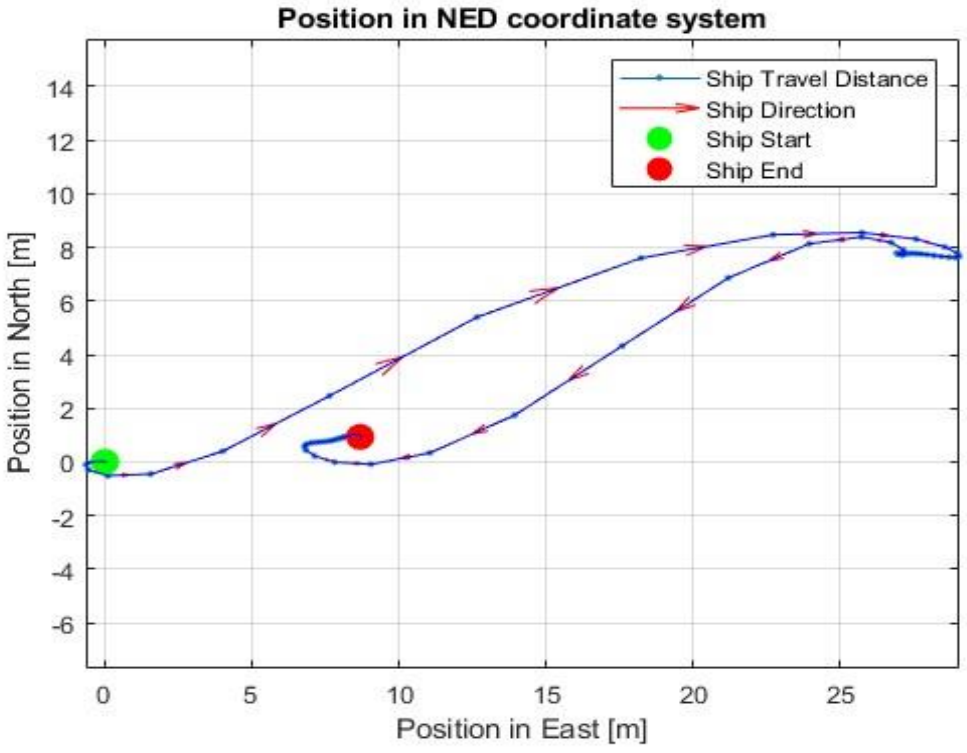

Figure 4.15: Vessel's position in NED coordinate system.

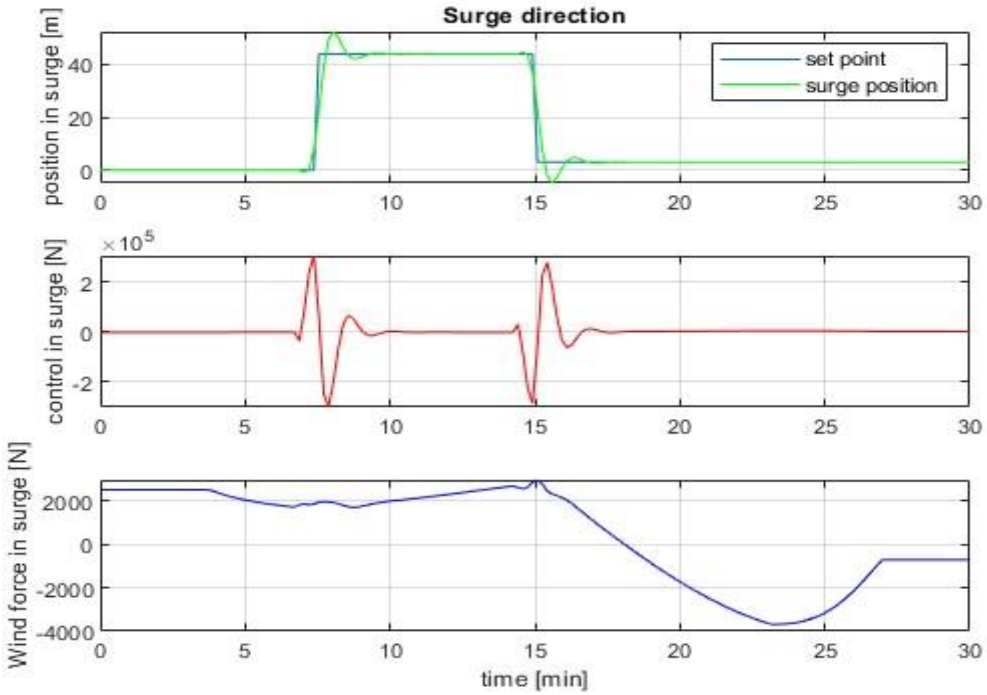

Figure 4.16: Simulation output in surge direction for Reduced-size MPC.

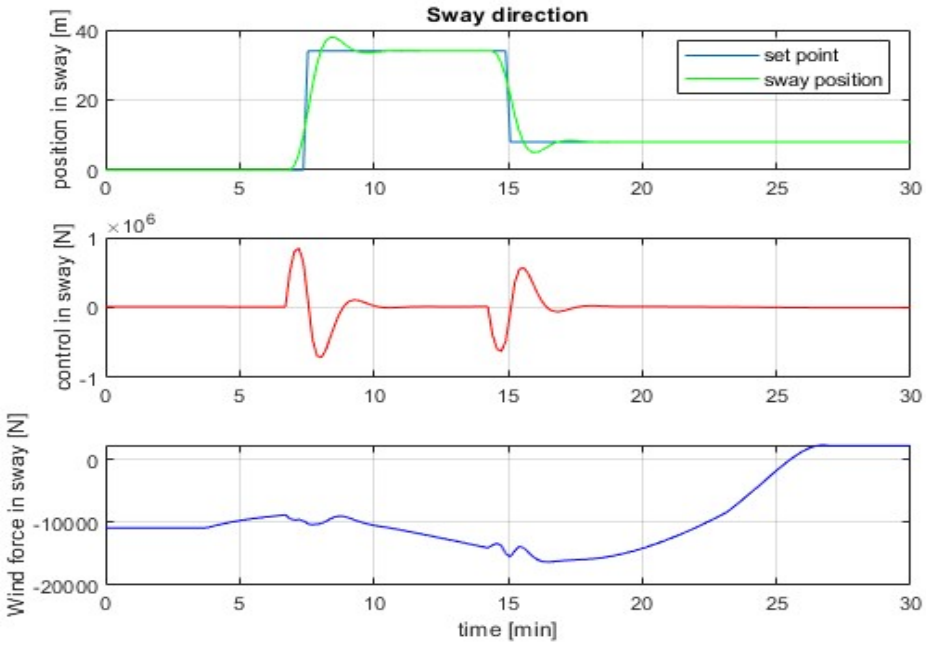

Figure 4.17: Simulation output in sway direction for Reduced-size MPC.

The new position at 15 minutes in NED coordinate system is (0.9, 8.5) *m*. From the figure it is clear that the was able to move to the desired place.

The Reduced-size MPC controller is used for evaluating the performance of the vessel, while the Kalman filter is used for measuring the immeasurable states. The results indicate that the amount of overshooting is reduced when there is a change in the setpoint in the surge, sway, or yaw direction. Furthermore, when observing the vessel movement from the NED coordinate system, it is noticeable that the motion appears smoother and faster. The simulation results are given in Appendix D.

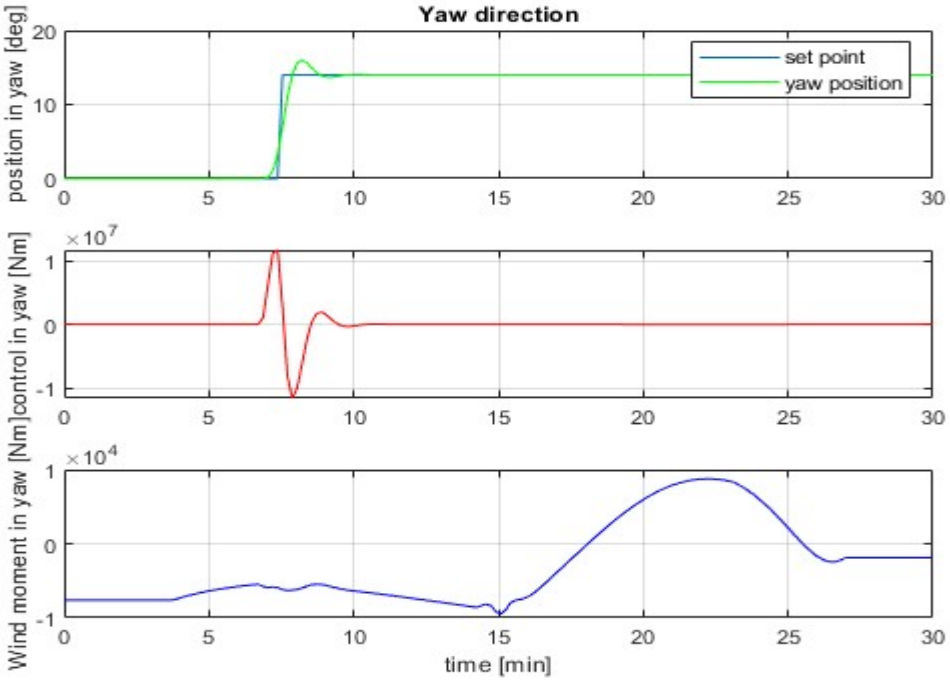

Figure 4.18: Vessel heading in yaw direction for Reduced-size MPC.

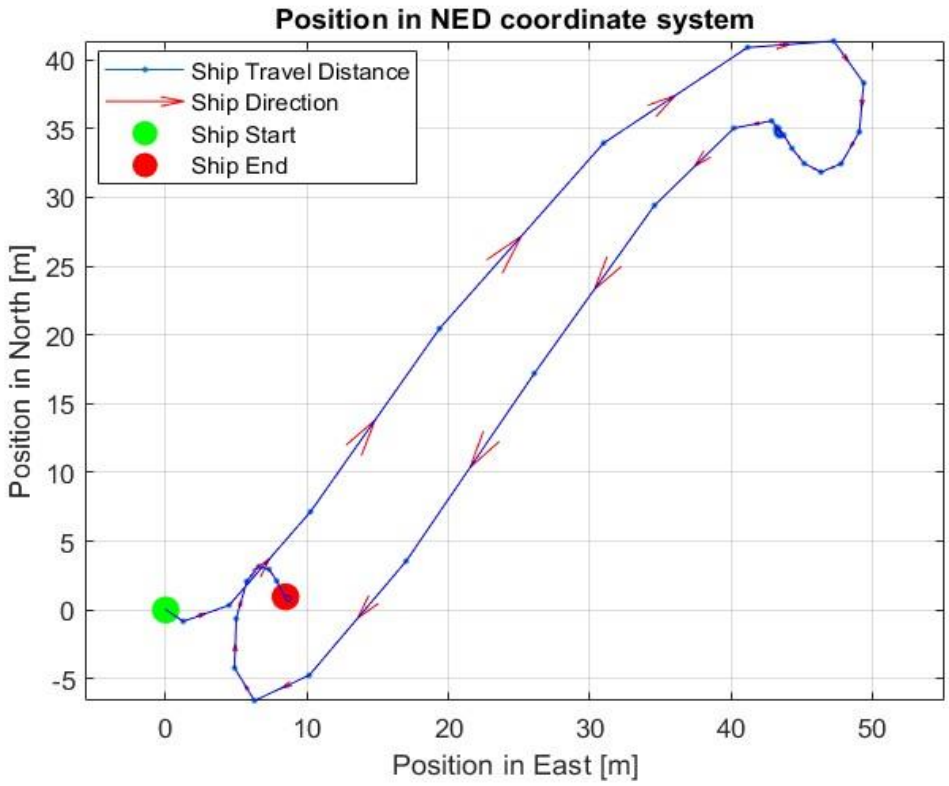

Figure 4.19:Vessel's position in NED coordinate system.

4.3 LQ optimal control with integral action

An LQ optimal control system with integral action used for this experiment to use the additional element in the cost function that penalizes the integral of the error over time, with the goal of eliminating steady-state errors. The integral action enhances performance by constantly adjusting the control signal according to the accumulated error, guaranteeing accurate tracking of desired reference signals even in the presence of disruptions or uncertainties in the model. For this experiment *dlqdu_pi.m* function is used which is developed by David Di Ruscio [20] [22].

Optimal control without any input constrains

In this experiment, the setpoint of the vessel changes twice in the surge direction. The simulation will last for a total of 20 minutes. During the first 5 minutes, the set point is set to 0 *m*. During the following 5 minutes, the target value changes to 15 *m*. At the 10-minute mark, there is a second change in the setpoint, with the position decreasing to 8 *m*.

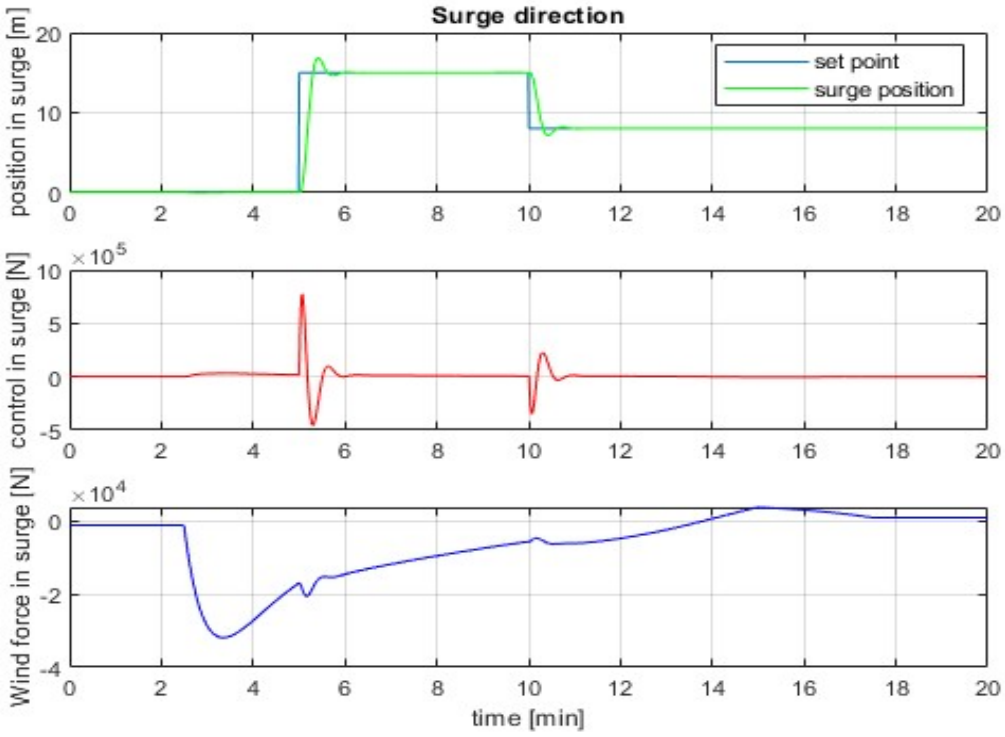

Figure 4.20: Vessel's position in surge direction using LQ optimal control with integral action.

Figure 4.20 shows that the control input began to increase to $0.1 \times 10^5\ N$ at 2.5 minutes, initially to compensate for the wind decrease, and later to reflect changes in the setpoint at 5 minutes. A thruster force of $7 \times 10^5\ N$ is applied to increase vessel's position in the surge by 15 *m.* After that controller noticed the overshoot and thruster force decreases to $-5 \times 10^5\ N$. It takes almost 1.5 minutes for the controller to reach the steady state. For the second setpoint change the thruster output decrease again to $-3 \times 10^5\ N$. After that the overshoot is encountered with a control input of $2 \times 10^5\ N$ to keep the vessel position at 8 *m*.

For the sway direction, as shown in Figure 4.21, the controller increases the thruster force approximately $5 \times 10^5\ N$ at 5 minutes to adjust the setpoint change from 0 *m* to 20 *m*. In order to counteract the overshoot, the controller reduces the thruster force once more. It took almost 3 minutes to reach a stable position. For the second setpoint change, the position shifts from 20 *m* to 6 *m*. So, the controller decreases the thruster output again. The wind started to decreased at 2.3 minute from $20 \times 10^4\ N$ to $-1 \times 10^4\ N$ at 5 minutes. The controller also addressed this change by increasing the thruster force. Also, at 13 minute a slight increase of wind also encountered perfectly by the LQ controller to maintain the vessel position in sway at 6 *m.*

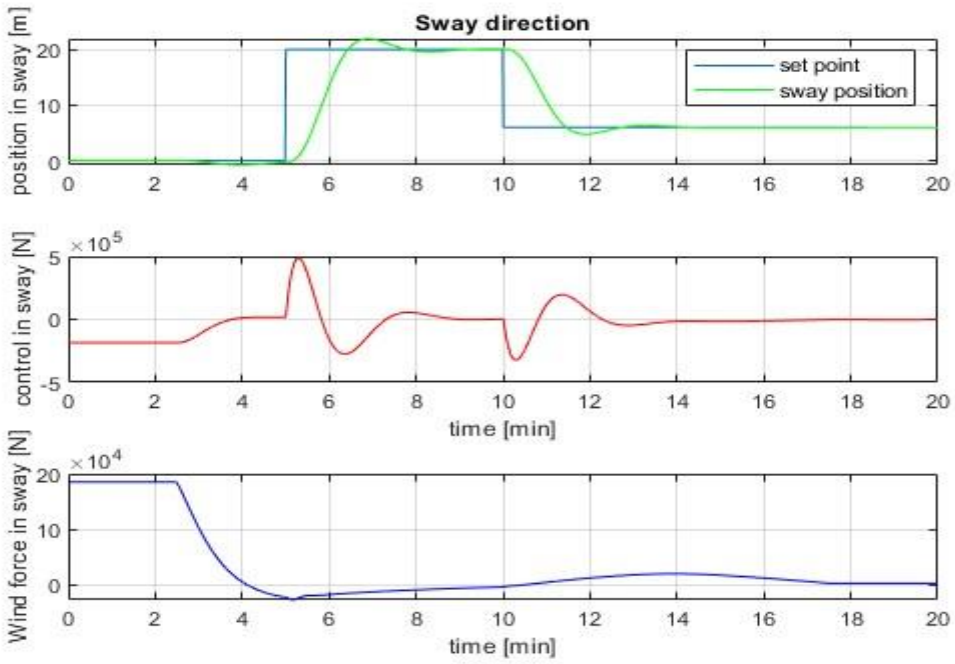

Figure 4.21: Vessel's position in sway direction using LQ optimal control with integral action.

The vessel is heading in a yaw direction, as shown in Figure 4.22, there is one setpoint change from 0° to 14° at 5 minutes and another at 10 minutes from 14° to 6°. To stabilize the vessel's yaw heading, the LQ controller generates thruster moment of $7 \times 10^6\ Nm$ at 5 minute and later decreases to $-3 \times 10^6\ Nm$ for encountering the overshoot. For the second heading change also successfully performed by the controller. For both setpoint change it took almost 3 minutes to go to the stable position. To counter the reducing and increasing wind throughout the whole 20 minutes of the simulation the controller produced appropriate thruster moment to stabilize the heading of the vessel.

Figure 4.23 shows the vessel position in the NED coordinate system. At first vessel position is in the origin. After 5 minutes the new setpoint in body coordinate system is (15, 20) *m* and the heading is changed to 14°, which in NED coordinate system is (9.7, 23.03) *m*. The MPC controller correctly move the vessel to the desired location. For the second setpoint change, the heading is 6° and the position in body coordinate moved to (3,8) *m*. The new North-East position is (7.3, 6.8) *m* and the controller again successfully moved the vessel into the desired position.

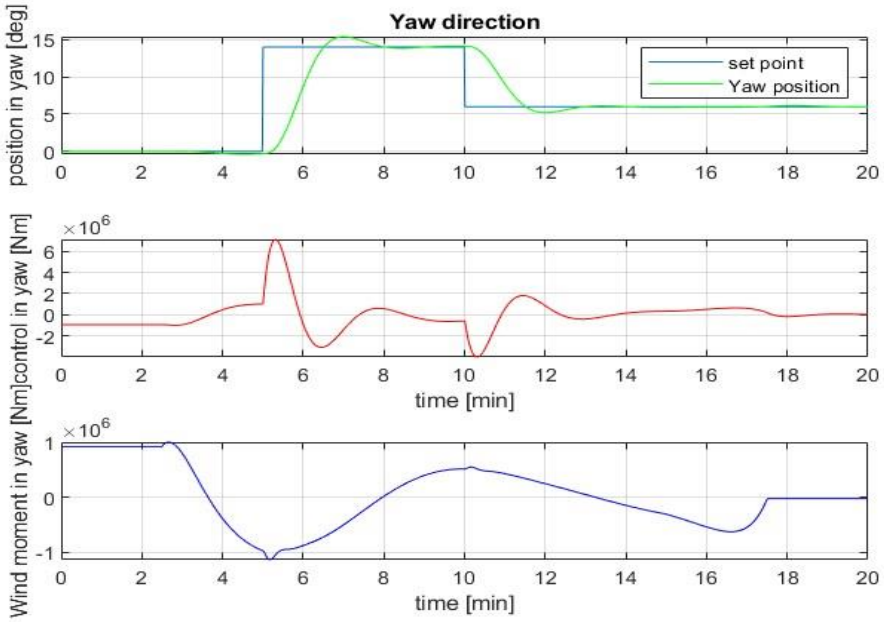

Figure 4.22: Vessel's heading in yaw direction using LQ optimal control with integral action.

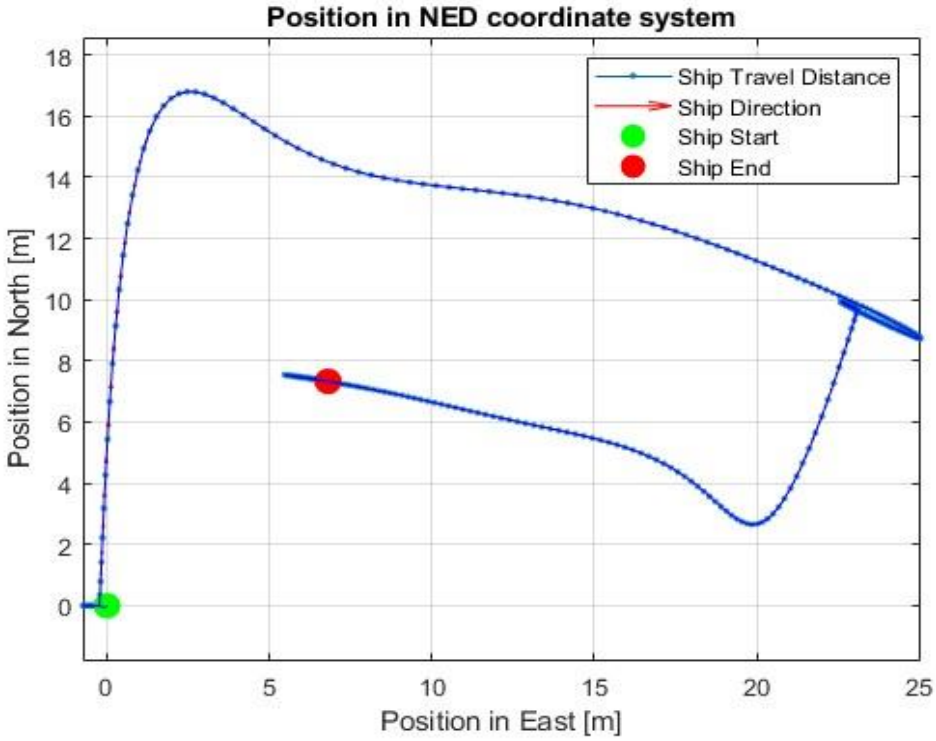

Figure 4.23: Vessel's position in NED coordinate system.

Optimal control with input constrains

An additional experiment is conducted to test the LQ controller with integral action, with the input control signal being restricted. The control force is constrained from $-1 \times 10^5\ N$ to $1 \times 10^5\ N$.

As shown in Figure 4.24, when the setpoint is changed to 15 *m*, the controller successfully moves the vessel to the desired location in 1 minute. At the 10-minute mark in the simulation, there are several factors influencing the vessel. The setpoint was adjusted again from 15 *m* to 8 *m*, and the wind decreased to a minimum of $-6 \times 10^4\ N$. Both of these factors contributed to a significant overshoot in the output. The vessel moved almost 6 *m* away from the desired position. Without the constrained input it took the vessel approximately 20 seconds to stabilized as described before in Figure 4.20. By comparing with the Figure 4.24 it takes more than1.5 minutes.

Figure 4.25 shows that when the first setpoint change occurred at the 5-minute mark, the vessel took a considerable amount of time to reach the desired setpoint of 20 *m* from its initial position of 0 *m*, due to its limited control input. After the overshoot, the controller attempts to decrease the input in order to address it. However, before reaching a stable state, a second setpoint change occurs at the 10-minute mark. The controller required 4 minutes to attain the stable position. It is evident that the first deviation from the desired value is smaller than the later deviation, which can be related to the influence of wind force. At 7.5 minutes, the wind force decreases to $-6 \times 10^4\ N$, indirectly aiding the controller in reducing overshoot. However, on the second occasion, the controller was faced with a rising wind and a change in the setpoint, resulting in a relatively longer period for the controller to effectively manage the situation.
For the heading of the vessel the applied is torque limited from $-6 \times 10^5\ Nm$ to $6 \times 10^5\ Nm$ as shown in Figure 4.27. The overall response of the controller is slow comparing to the Figure 4.22.

Figure 4.26 shows the vessel motion in NED coordinate system. It shows the vessel is able to move to the desired position first at (9.7, 23.03) *m* and later at (7.3, 6.8) *m.* When comparing it with Figure 4.23 it shows with constrained control input, the vessel took two large deviations in the NED coordinate plane. That describes the overshoot of the vessel position in surge, sway and yaw. So, with constrained input the vessel can move to the desired position but the controller takes more time to perform the task.

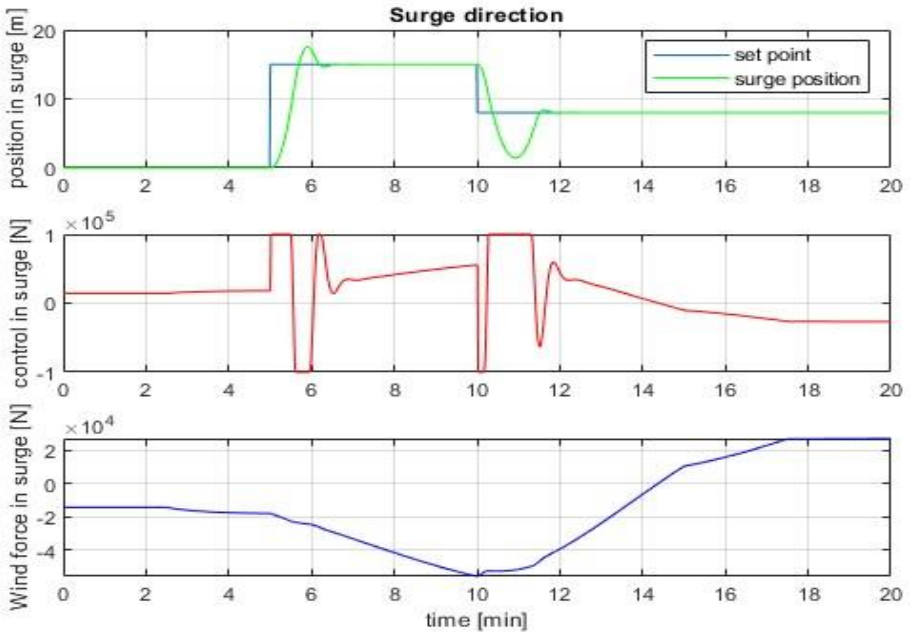

Figure 4.24: Vessel's position in surge direction using LQ optimal control with constrained input.

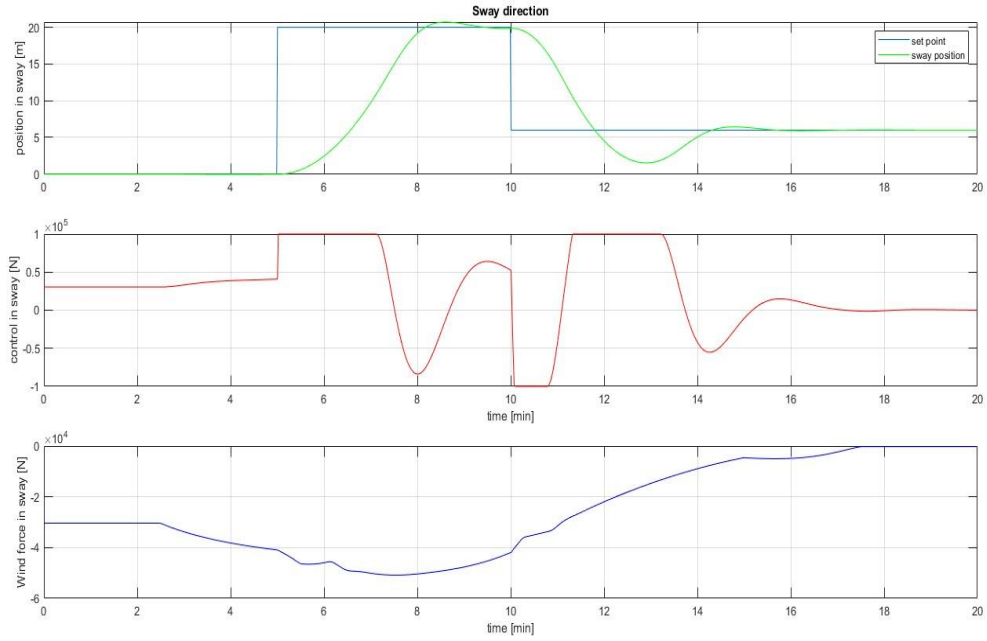

Figure 4.25: Vessel's position in sway direction using LQ optimal control with constrained input.

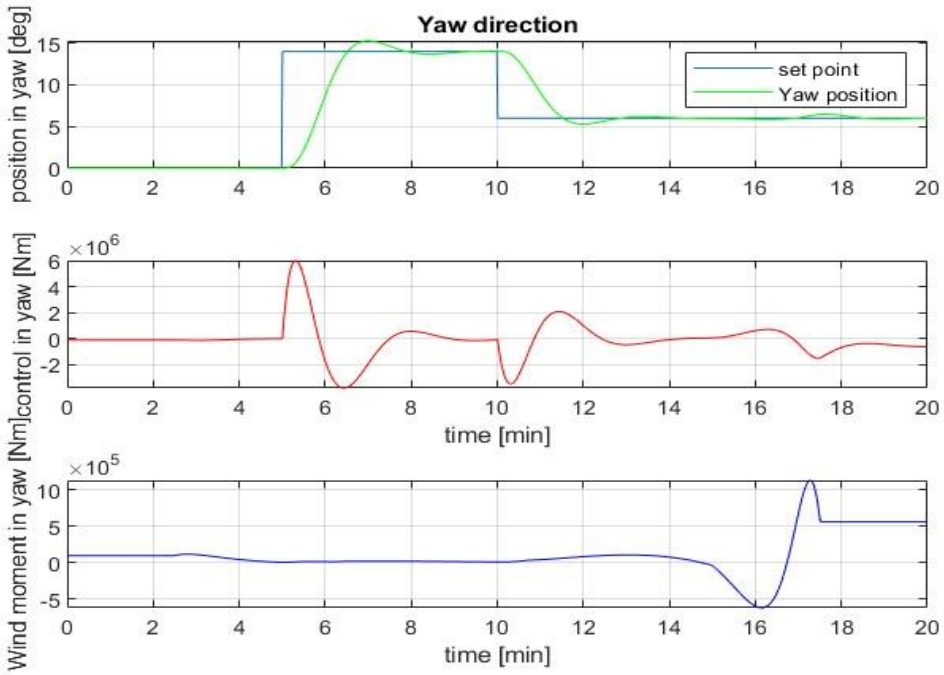

Figure 4.27: Vessel's heading in yaw direction using LQ optimal control with constrained input.

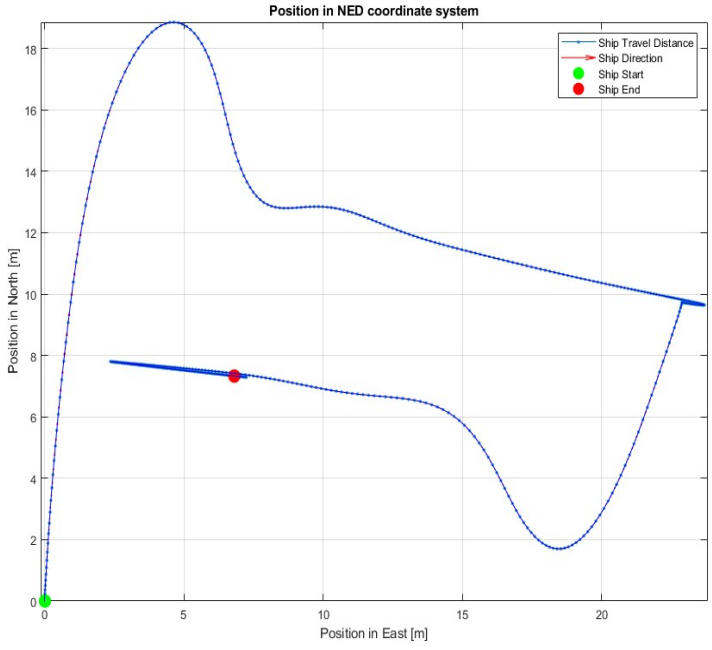

Figure 4.26: Vessel's position in NED coordinate system.

4.4 Simple MPC

This section illustrates an experiment that was conducted using the Balchen reduced model and the Simple MPC to control the position of the vessel.

4.4.1 Simple MPC without integral action

The overall duration of the simulation is 20 minutes. There are two separate instances of setpoint change throughout the entire simulation duration.
Figure 4.28 vessel's motion in the surge direction. Before reaching the initial setpoint change, the vessel deviates from its original position of 0 m due to a declining wind. The wind force decreases gradually over 10 minutes. Despite the controller's attempt to provide a sufficient response by boosting the control output to 15×10^4 N, it was unable to move the vessel to the desired 3 m location. There is always a lag before reaching the next setpoint change. The wind provided assistance to the controller by giving a force of 1×10^4 N to bring the vessel to the 1 m mark. However, the force continued to increase and the vessel once again deviated from its intended position by the end of the simulation time.

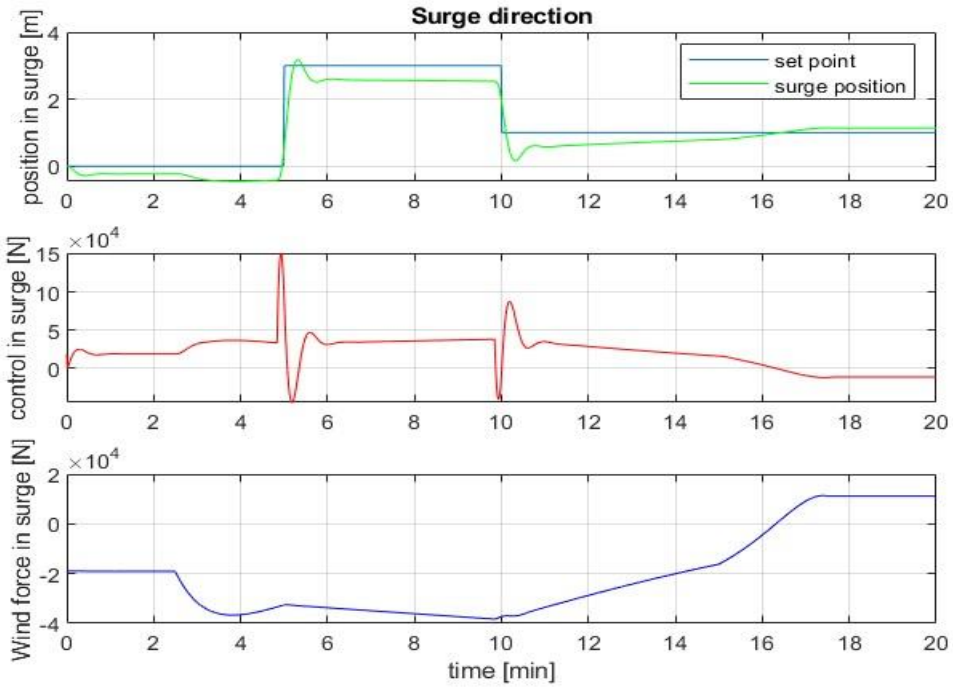

Figure 4.28: Simulation output in surge direction for Simple MPC

So, it shows the Simple MPC without the integral action is very sensitive to the wind force change in surge direction. For further investigation, the simulation time has been increased to 40 minutes. Figure 4.29 indicates the wind force applied to the vessel in the surge direction is nearly negligible, equivalent to 0 N. The controller's performance is excellent, as it effectively

handles setpoint changes with minimal overshoot and quickly reaches a stable position. However, when the wind force reaches $4 \times 10^4\ N$, the controller attempted to counteract it by decreasing the control input. Despite this effort, there is still an error in the vessel's location at the end of the simulation. So, without effect of integral action controller was not able to fix error for the wind disturbances.

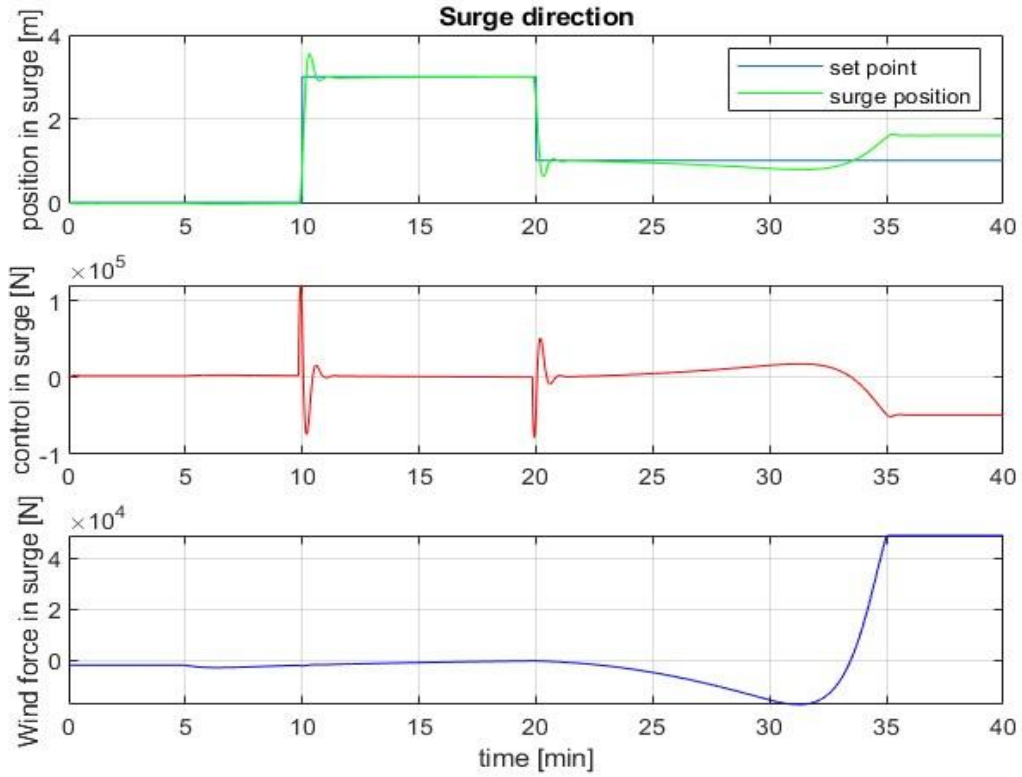

Figure 4.29: Simulation output in surge direction for Simple MPC for investigating wind effects.

For the sway direction, as shown in Figure 4.30 the controller increases the thruster force approximately $2\ \times 10^6\ N$ at 4 minutes to adjust the setpoint change from 0 *m* to 5 *m*. In order to counteract the overshoot, the controller reduces the thruster force once more. It took 1 minute to reach a stable position. For the second setpoint change, the position shifts from 5 *m* to 8 *m*. So, the controller decreases the thruster output again. Between 5 and 16 minutes, the wind is increased to $5\ \times 10^4\ N$. The controller seems counter this force to maintain the vessel position in sway at 8 *m*.

The vessel is heading in yaw direction, as shown in Figure 4.31, there is one setpoint change from 0° to 3° at 5 minutes. To stabilize the vessel's yaw heading, the MPC controller generates thruster moment of $2.5 \times 10^7\ Nm$. To counter the reducing wind and the overshot, it reduces the thruster moment again. It takes the controller approximately 1.5 minutes to stabilize the

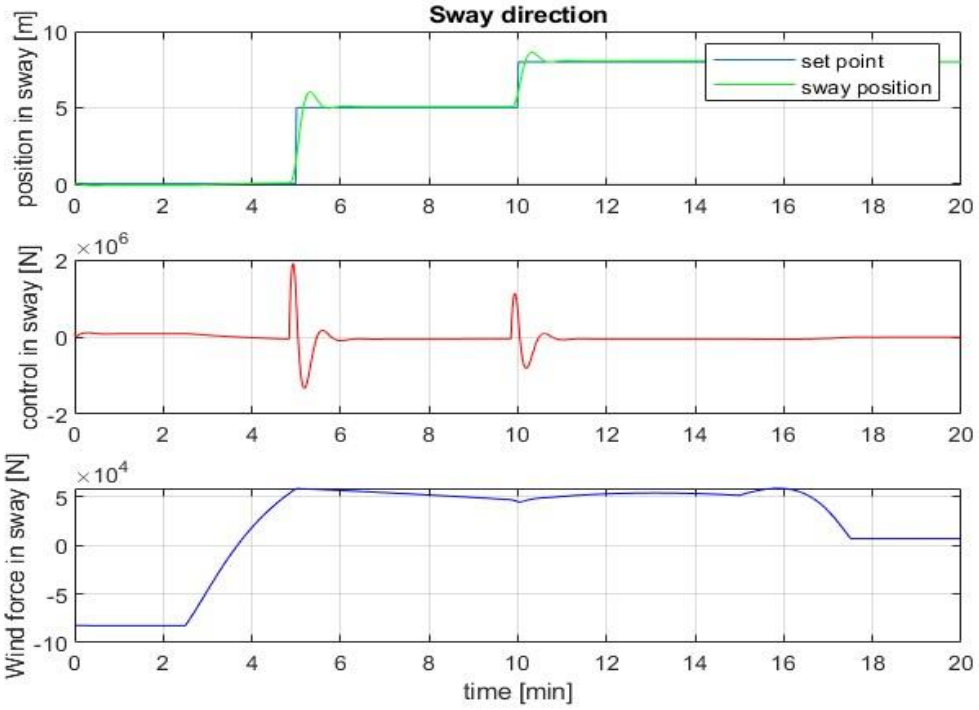

Figure 4.30: Simulation output in sway direction for Simple MPC.

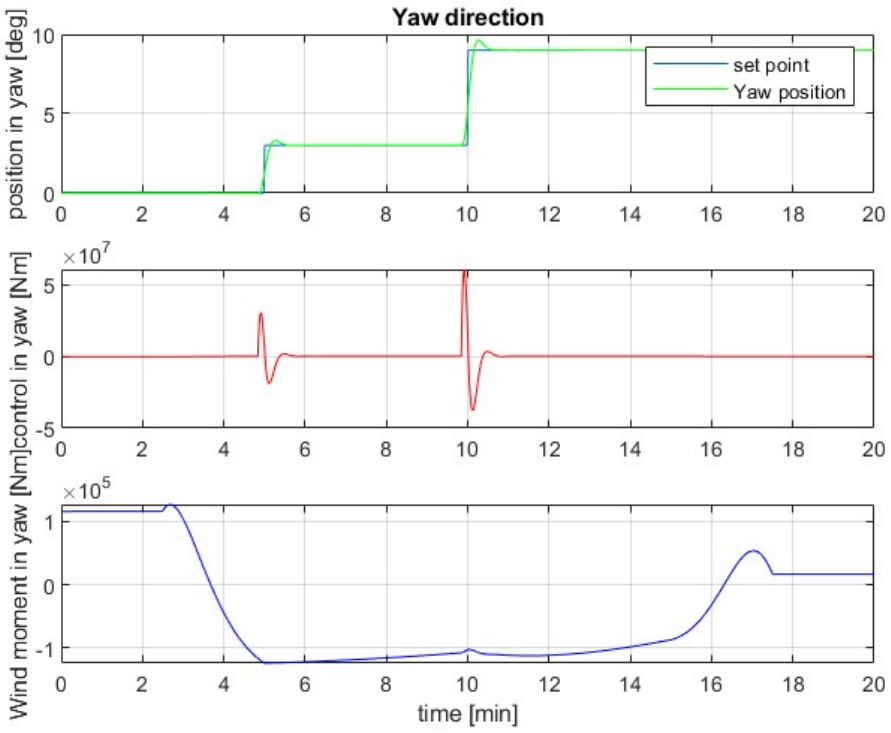

Figure 4.31: Vessel heading in yaw direction for Simple MPC.

heading at 15°. For the next heading change from 3° to 9°, the controller produces a larger thrust to reach the desired heading.

Figure 4.32 shows the vessel position in the NED coordinate system. At first vessel position is in the origin. After 5 minutes the new setpoint in body coordinate system is (3, 5) *m* and the heading is changed to 3°, which in NED coordinate system is (2.7, 5.1) *m*. The MPC controller correctly move the vessel to the desired location. For the second setpoint change, the heading is now 3° and the position in body coordinate moved to (1,8) *m*. The new North-East position is $(-0.26, 8.05)$ *m* and the controller again successfully moved the vessel into the desired position.

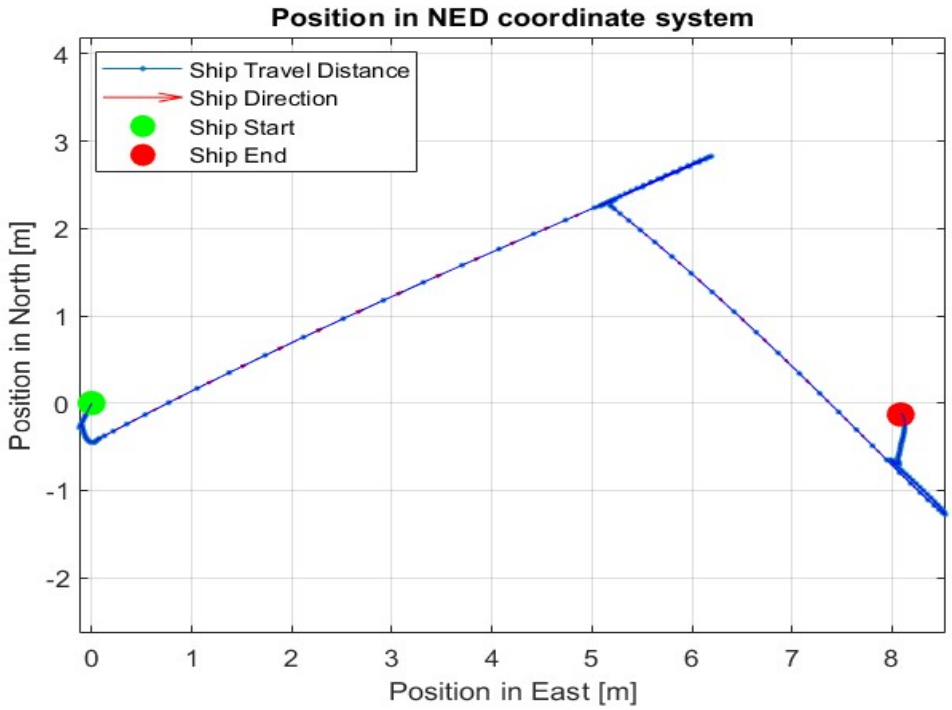

Figure 4.32: Vessel's position in NED coordinate system.

4.4.2 Simple MPC with integral action

The integral action is added with Simple MPC to simulates the next experiments. Figure 4.33 points out the controller's effective response to the change in setpoints and the overshoot. The vessel is stabilized to its intended position within 1.5 minutes for both setpoint changes. The variation in wind force decreases from $1 \times 10^4\ N$ to $-3 \times 10^4\ N$ after 10 minutes. In response, the controller slightly increases the thruster force to maintain the vessel's location. The most noticeable observation is that the inclusion of integral action effectively rectifies the error induced by the wind force.

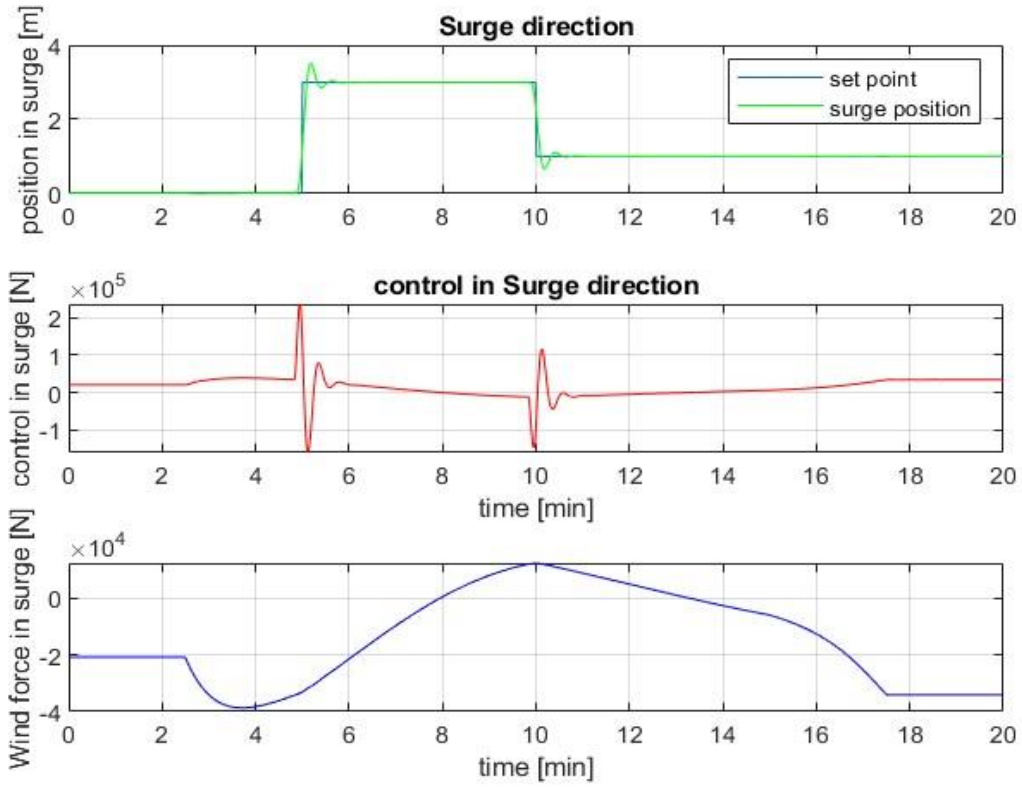

Figure 4.33: Simulation output in surge direction for Simple MPC.

Simple MPC with integral action in both the sway and yaw directions demonstrates that the controller is capable of exhibiting great performances. The rapid and accurate changing of position in sway and heading in yaw was achieved with a small overshoot from the desired trajectory. However, there is no significant difference in results between the controllers that did not use the integral action to correct the error. The Appendix E documents the simulation results.

The vessel's position in NED coordinate system is shown in Figure 4.34. Initially, the vessel remained in its original position. After 5 minutes, the vessel moved 3 *m* in surge and 5 *m* in sway. The vessel is heading in the yaw direction by 3°. So, the vessel's position is 2.7 *m* in the north and 5.1 *m* in the east. For the next positional shift, the vessel moved (1,8) *m* in the body coordinate system. The position, when converted to NED, is (-0.2, 8.05) *m*. When using the integral action, it becomes obvious that the vessel motion is smoother and the deviation from the setpoint is less.

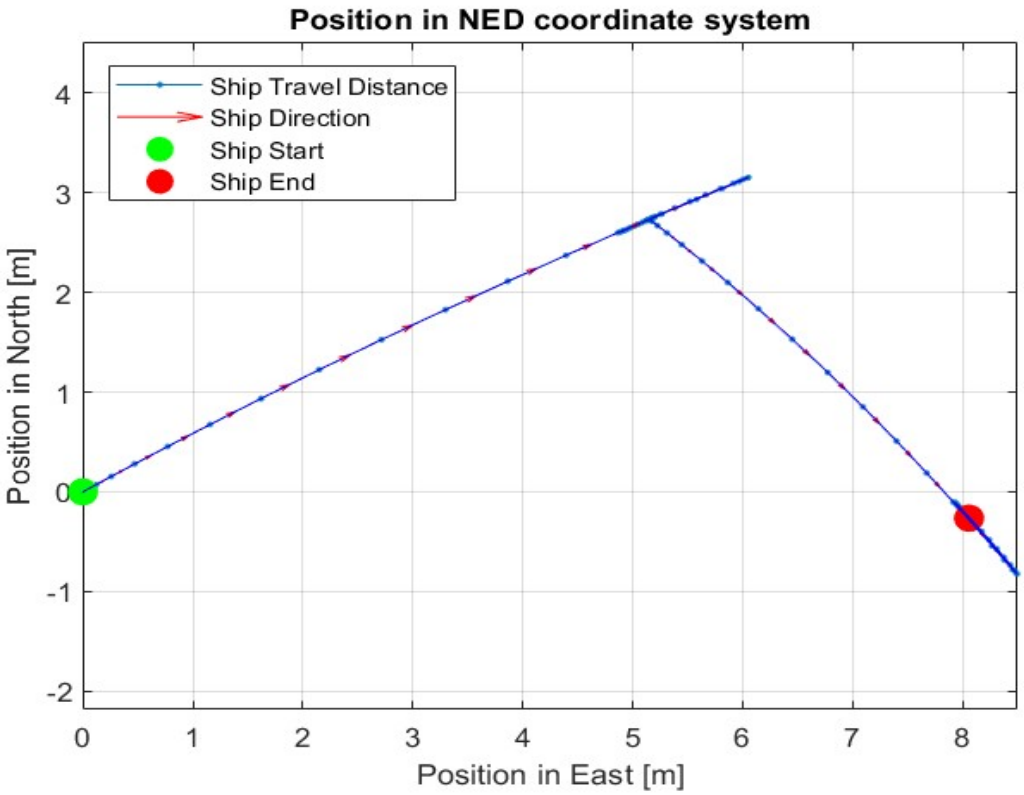

Figure 4.34: Vessel's position in NED coordinate system.

4.5 Comparison between Simple MPC and LQ optimal control

The comparison of two controllers is conducted by limiting the control input within the range of $-5 \times 10^5\ N$ to $5{\times}10^5$ N. The objective was to evaluate the performance of the controller in maneuvering the vessel to a desired position while limiting the power of the thruster force. Figure 4.36 shows that the implementation of the Simple MPC required a reduced amount of thruster force to effectively maneuver the vessel. Additionally, when compared to the LQ optimum controller, the performance is slightly better while reaching the setpoint.
Figure 4.35 demonstrates that the LQ controller displayed much better performance in the sway direction. The amount of control force required and the time taken to reach the setpoint were lower in LQ controller. In the case of the Simple MPC, the setpoint was reached within 4 minutes, however it is important to note that there was oscillation before achieving the setpoint, which is known as integral windup. The control system's integral term accumulates error as a result of significant changes in the setpoint. But the controller is unable to adequately correct for this due to limitations in the control input.

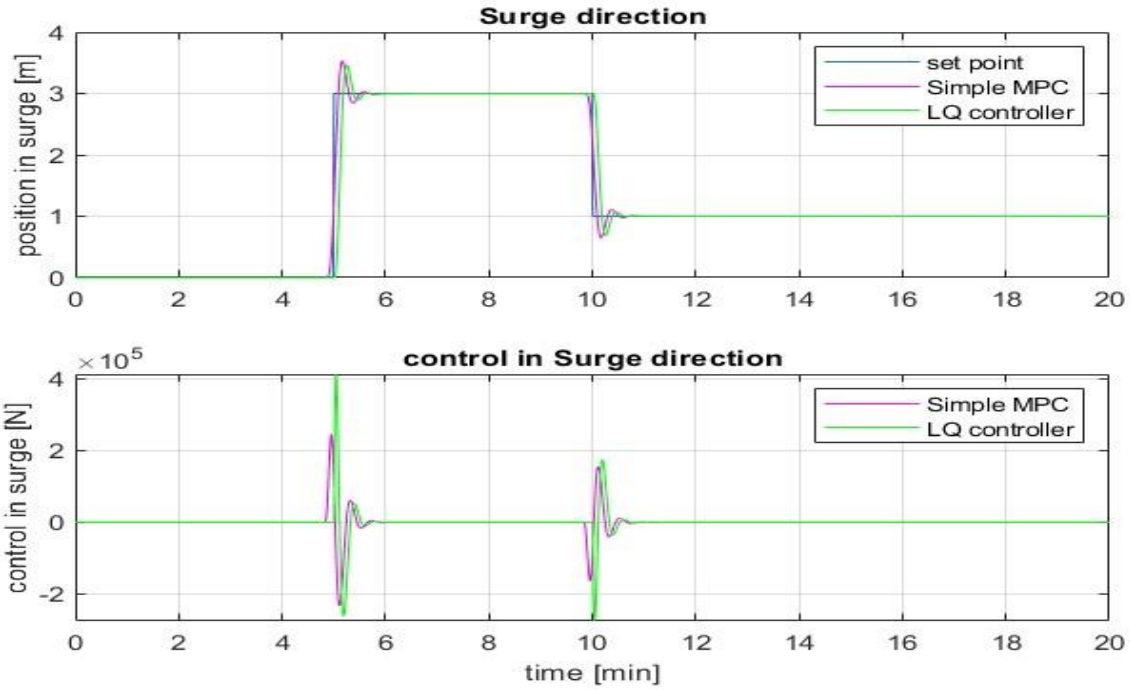

Figure 4.36: Comparison between Simple MPC with integral action and LQ optimal control with integral action in surge direction.

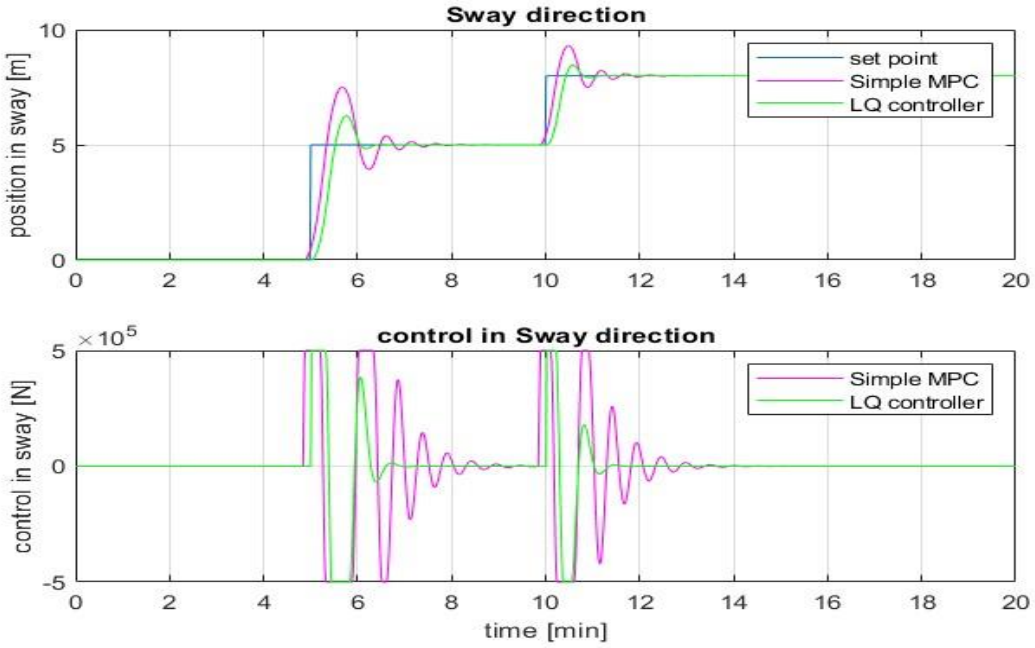

Figure 4.35: Comparison between Simple MPC with integral action and LQ optimal control with integral action in sway direction.

Figure 4.37 displays the comparison in the yaw direction. The LQ controller exhibits poor performance due to oscillations and lack of stability during the entire simulation period. The performance of the Simple MPC was better, effectively maintaining control over the vessel's heading in yaw direction.
While controlling the vessel's position with a limited input thruster force, the Simple MPC's overall performance is better than that of the LQ controller.

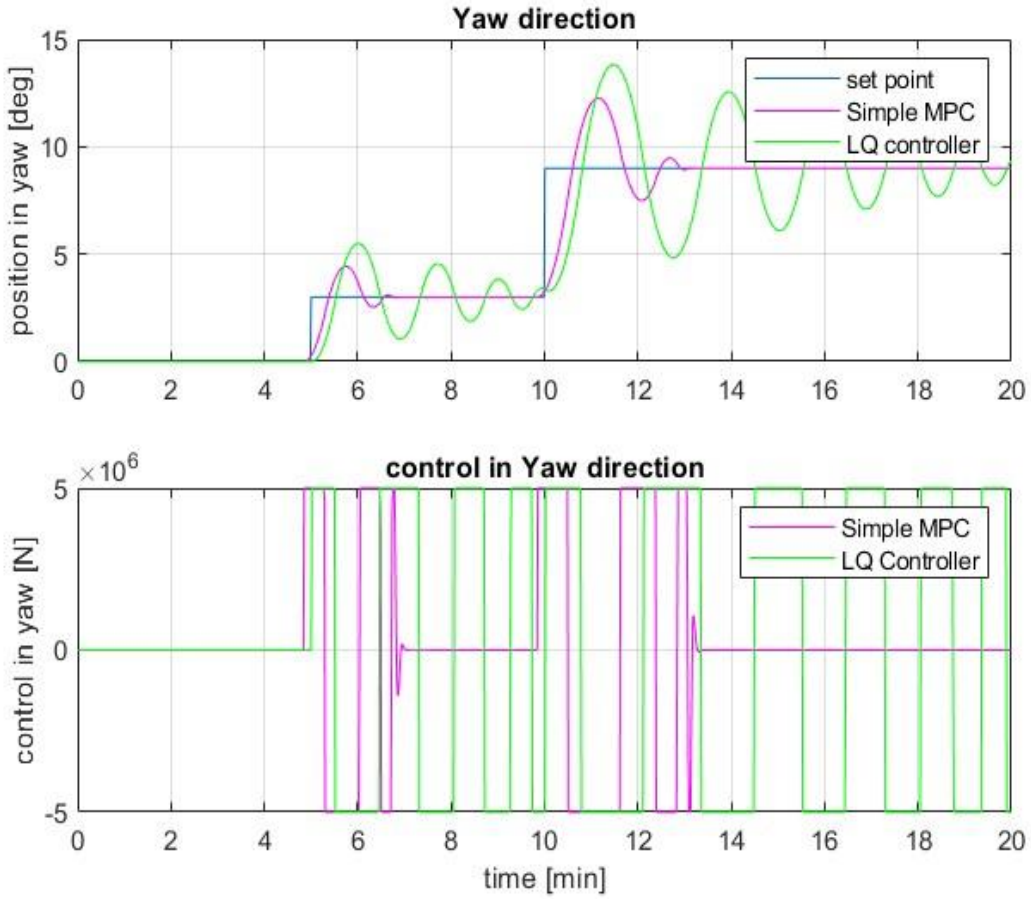

Figure 4.37: Comparison between Simple MPC with integral action and LQ optimal control with integral action in yaw direction.

4.6 Controller performance analysis using IAE and TV index

The Integrated Absolute Error (IAE) is a metric used in control systems to evaluate the overall efficacy of a controller by measuring the magnitude of error over a given time period. The IAE is determined by integrating the absolute value of the error function $e(t)$ throughout the time interval t [23]. Mathematically it can be shown,

$$IAE = \int_0^T |e(t)|\, dt \qquad 4.5$$

In equation 4.5 T represents the sampling time and the $e(t)$ is the absolute value of error at any given time.

The Total Variation (TV) index is a performance metric used to evaluate the efficiency of the controller's actions. In particular, it provides a numerical value for the total amount of time that control actions have changed [23]. It can be calculated using the equation 4.6,

$$TV = \int_{K=1}^{N} |\Delta u_K| \qquad 4.6$$

Both the IAE and TV indexes are essential to assessing the efficiency of a control system. The IAE helps in evaluating the controller's ability to reduce error over time, which directly affects the stability and accuracy of the system. The TV index offers an insight on the mobility and efficiency of the control actions, which might affect the longevity of actuators and their probability of producing oscillations.

The simulation time is set at a fixed duration of 15 minutes to evaluate the performance of multiple controllers in three different directions. During the simulation, there is a single setpoint change that occurs at the midpoint of the simulation. The following tables show the IAE and TV index values of different controllers.

Table 4.1: IAE and TV index for Standard MPC.

Direction	IAE Index	TV index
Surge	120.2	15.85×10^4
Sway	175.5	55.39×10^4
Yaw	1.4	2482.9×10^4

Table 4.2: IAE and TV index for Reduced size MPC.

Direction	IAE Index	TV index
Surge	115.4	14.34×10^4
Sway	164.60	45.41×10^4
Yaw	1.6	1085.6×10^4

Table 4.3: IAE and TV index for LQ optimal control with integral action.

Direction	IAE Index	TV index
Surge	268.9	25.62×10^4
Sway	886.6	14.7×10^4
Yaw	9.5	134.7×10^4

Table 4.4: IAE and TV index for Simple MPC.

Direction	IAE Index	TV index
Surge	199.1	47.9×10^4
Sway	125.7	452.2×10^4
Yaw	1.1	823.8×10^4

Table 4.5: IAE and TV index for Simple MPC with integral action.

Direction	IAE Index	TV index
Surge	151.2	36×10^4
Sway	146.2	354.7×10^4
Yaw	1.8	654.8×10^4

Based on the results, Table 4.6 is created that shows the performance of the controllers.

Table 4.6: Performance of the Controllers.

Controller	Direction	IAE Performance	TV Performance	Combined Assessment
Standard MPC	Surge	Moderate	Low	Low
	Sway	Moderate	Low	Low
	Yaw	Good	Moderate	Moderate
Reduced Size MPC	Surge	Moderate	Moderate	Moderate
	Sway	Moderate	Moderate	Moderate
	Yaw	Good	Moderate	Moderate
LQ Optimal Controller with integral action	Surge	Low	Good	Moderate
	Sway	Low	Moderate	Low
	Yaw	Good	Moderate	Moderate
Simple MPC	Surge	Low	Low	Low
	Sway	Moderate	Moderate	Moderate
	Yaw	Good	Moderate	Moderate
Simple MPC with integral action	Surge	Moderate	Good	Moderate
	Sway	Moderate	Good	Moderate
	Yaw	Good	Moderate	Moderate

Based on the summary table and considering the overall performance in managing both Integral Absolute Error (IAE) and Total Variation (TV) in all directions, the performance of the Standard MPC is rather low in the surge and sway directions because of its high TV indexes. However, it performs pretty well in the yaw direction. The reduced size MPC exhibits overall good performance in each direction, successfully achieving a good score for IAE and TV.
The LQ optimal control with integral action gives mixed results, demonstrating effective control in the surge direction for TV, but not as effective in sway and the performance in yaw is moderate. The simple MPC gives poor performance in the surge direction, but moderate performance in the sway and yaw directions. The simple MPC with integral demonstrates a satisfactory stability in every aspect, possibly offering the most stable performance among the all controllers.

The LQ optimal control with integral action is the most preferable choice among the evaluated controllers. It displays stable control capabilities and efficiently minimizes errors and control variations. The strong performance across multiple measures indicates the effectiveness for maintaining system stability and accuracy. The reduced size MPC can also be a suitable option, as it requires less computational power and shows balanced performance for both indexes in all directions.

5 Discussion

The thesis's primary goal is to reconstruct the Balchen model and experiment with the new model. The Balchen model, developed in 1980, was one of the earliest mathematical models used for Dynamic Positioning (DP) of ships. The new model is a reduced version developed by removing the drag and momentum coefficient from the original model.

The reduced model displays the characteristics of a three double integrator model. When comparing between the simplified model and the original model in the presence of wind disturbances and without any input thruster force, it can be observed that both models exhibit nearly identical behavior. The parameter values of the drag and momentum coefficients are really small, which were used to simulate the original model. So, eliminating these did not cause any significant differences in the reduced model.

The reduced model has been evaluated with various control systems. The performance of the controllers is tested using Integrated Absolute Error (IAE) and Total Variation (TV) index. Standard MPC performed good with the use of Kalman filter. Reduced-size MPC is developed by simplifying the objective function. The results of Reduced-size MPC is much better than the Standard MPC as it has lower IAE and TV index.

Simple MPC controller performed well when there were no environmental disturbances. The results indicate that the disturbances cause the vessel position to deviate from the setpoint in the surge direction. However, using the Simple MPC with integral action can effectively solve this problem. The controller gives balanced performance in all three directions (surge, sway, and yaw). The IAE index shows that the Simple MPC with integral action has very good performance overall. The TV index in Simple MPC is quite high. But this algorithm can reduce this large control input by adjusting the corresponding weight. Simple MPC with integral action delivers the most efficient results among all controllers, ensuring perfect vessel control.

The LQ optimal controller with integral action showed poor performance in the IAE index compared to the other controllers. In the event of a setpoint change, the controller took longer to adjust the vessel's position in surge and sway. Aslo, the experiment result showed that when the input thruster forces are limited, the controller response is very slow and performance is not that good. The tuning of P and Q matrices in the LQ optimal controller can be used to find better performance. It is a trial-and-error process to find the optimal control input for the vessel.

It can be said that the MPC is simply the implementation of the LQ optimal controller in each iteration of the control loop along with the sliding horizon. Because a MPC controller uses inequality and equality constraints to find an optimized response, which gives an advantage over the LQ controller. However, this increases the computational costs.

In real life scenario, more than one control method is used for the control of a complex system like marine vessel. For Standard MPC, it will create a challenge for the high computational costs when more constrains will be added or the prediction horizon length is increased. So Reduced size MPC and Simple MPC with integral action can be used to conduct high-level

control. This will help to create the optimal operational window for the vessel. On the other hand, the LQ optimal control with integral action can be used for set point tracking.

The future work can be suggested as:

- Use System Identification to develop adaptive control algorithms that can adjust to different weather conditions, utilizing data-driven models derived from input-output data.
- Applying matrices for determining each of the forces and torques generated by each thruster, compared to only calculating the total forces and torques from all thrusters.
- Proposed simplified model can be tested on a real vessel.

6 Conclusion

This report gives a theoretical explanation of the coordinate system a vessel uses for navigation, along with the external and internal factors that affect the vessel's movement. The Balchen low-frequency model is simplified by eliminating the drag and momentum coefficients in this report. The reduced model was tested with different control methods, such as Standard MPC, Reduced-size MPC, Simple MPC, Simple MPC with integral action, and LQ-optimal control with integral action.

The behavior of the simplified model and the original Balchen model is almost same due to low value of drag and momentum coefficient. Among all the control methods tested on the model, the Simple MPC with integral action shows the best performance under wind disturbances and various positional changes. Also, Reduced-size MPC controller gives a good performance while using less computational power.

References

[1] A. J. Sørensen, S. I. Sagatun, and T. I. Fossen, "Design of a dynamic positioning system using model-based control," *Control Eng. Pract.*, vol. 4, no. 3, pp. 359–368, Mar. 1996, doi: 10.1016/0967-0661(96)00013-5.

[2] A. J. Sørensen, "A survey of dynamic positioning control systems," *Annu. Rev. Control*, vol. 35, no. 1, pp. 123–136, Apr. 2011, doi: 10.1016/j.arcontrol.2011.03.008.

[3] "The story behind dynamic positioning." [Online]. Available: https://www.kongsberg.com/kmagazine/2014/3/story-behind-dynamic-positioning/

[4] J. G. Balchen, N. A. Jenssen, E. Mathisen, and S. Sælid, "A Dynamic Positioning System Based on Kalman Filtering and Optimal Control," vol. 1, no. 3, pp. 135–163, Jan. 1980, doi: 10.4173/mic.1980.3.1.

[5] S. Sælid and N. A. Jenssen, "Adaptive ship autopilot with wave filter," vol. 4, no. 1, pp. 33–45, Jan. 1983, doi: 10.4173/mic.1983.1.3.

[6] P. Fung and M. Grimble, "Dynamic ship positioning using a self-tuning Kalman filter," *IEEE Trans. Autom. Control*, vol. 28, no. 3, pp. 339–350, Mar. 1983, doi: 10.1109/TAC.1983.1103226.

[7] J. P. Strand, A. J. Sørensen, and T. I. Fossen, "Design of automatic thruster assisted mooring systems for ships," vol. 19, no. 2, pp. 61–75, Jan. 1998, doi: 10.4173/mic.1998.2.1.

[8] A. J. Sørensen, S. I. Sagatun, and T. I. Fossen, "Design of a dynamic positioning system using model-based control," *Control Eng. Pract.*, vol. 4, no. 3, pp. 359–368, Mar. 1996, doi: 10.1016/0967-0661(96)00013-5.

[9] T. I. Fossen and A. Grovlen, "Nonlinear output feedback control of dynamically positioned ships using vectorial observer backstepping," *IEEE Trans. Control Syst. Technol.*, vol. 6, no. 1, pp. 121–128, Jan. 1998, doi: 10.1109/87.654882.

[10] M. F. Aarset, J. P. Strand, and T. I. Fossen, "Nonlinear Vectorial Observer Backstepping with Integral Action and Wave Filtering for Ships," *IFAC Proc. Vol.*, vol. 31, no. 30, pp. 77–82, Oct. 1998, doi: 10.1016/S1474-6670(17)38420-3.

[11] T. I. Fossen, "A survey on Nonlinear Ship Control: from Theory to Practice," *IFAC Proc. Vol.*, vol. 33, no. 21, pp. 1–16, Aug. 2000, doi: 10.1016/S1474-6670(17)37044-1.

[12] O. M. Aamo and T. I. Fossen, "Controlling line tension in thruster assisted mooring systems," in *Proceedings of the 1999 IEEE International Conference on Control Applications (Cat. No.99CH36328)*, Kohala Coast, HI, USA: IEEE, 1999, pp. 1104–1109. doi: 10.1109/CCA.1999.801126.

[13] T. Perez, Ø. N. Smogeli, T. I. Fossen, and A. J. Sørensen, "An Overview of the Marine Systems Simulator (MSS): A Simulink Toolbox for Marine Control Systems," *Model. Identif. Control Nor. Res. Bull.*, vol. 27, no. 4, pp. 259–275, 2006, doi: 10.4173/mic.2006.4.4.

[14] T. I. Fossen, "cybergalactic/MSS." Apr. 05, 2024. [Online]. Available: https://github.com/cybergalactic/MSS

[15] C. G. Källström, “Autopilot and Track-Keeping Algorithms for High Speed Craft,” *IFAC Proc. Vol.*, vol. 31, no. 30, pp. 229–234, Oct. 1998, doi: 10.1016/S1474-6670(17)38445-8.

[16] D. Di Ruscio, “Model Predictive Control with Integral Action: A simple MPC algorithm,” *Model. Identif. Control Nor. Res. Bull.*, vol. 34, no. 3, pp. 119–129, 2013, doi: 10.4173/mic.2013.3.2.

[17] “Handbook of Marine Craft Hydrodynamics and Motion Control by Fossen, Thor I.. 9781119575054. Hardcover - 2021 | Akademika.no.” [Online]. Available: https://www.akademika.no/teknologi/teknisk-kybernetikk/handbook-marine-craft-hydrodynamics-and-motion-control/9781119575054,https://github.com/cybergalactic/FossenHandbook

[18] N. Bargouth, “Dynamic positioning, system identification and control of marine vessels,” Master thesis, University of South-Eastern Norway, 2022. Accessed: Apr. 19, 2024. [Online]. Available: https://openarchive.usn.no/usn-xmlui/handle/11250/3000363

[19] R. Sharma, “Lecture notes for the course IIA 4117: Model Predictive Control,” 2019, [Online]. Available: https://web01.usn.no/~roshans/mpc/downloads/lecture_notes_MPC.pdf

[20] D. D. Ruscio, “iscrete LQ optimal control with integral action: A simple controller on incremental form for MIMO systems,” *Model. Identif. Control Nor. Res. Bull.*, vol. 33, no. 2, pp. 35–44, 2012, doi: 10.4173/mic.2012.2.1.

[21] F. A. Haugen, “Modeling, Simulation and Control”, [Online]. Available: https://techteach.no/control/modsimcon/mod_sim_con_2023_08_24.pdf

[22] David Di Ruscio, “dlqdu_pi.m.” 2018. [Online]. Available: https://web01.usn.no/~davidr/sec4106/ovinger/integrator_ex/matlab/dlqdu_pi.m.

[23] D. D. Ruscio, “n Tuning PI Controllers for Integrating Plus Time Delay Systems,” *Model. Identif. Control Nor. Res. Bull.*, vol. 31, no. 4, pp. 145–164, 2010, doi: 10.4173/mic.2010.4.3.

Appendices

Appendix A

Task Description

University of South-Eastern Norway

Faculty of Technology, Natural Sciences and Maritime Sciences, Campus Porsgrunn

FMH606 Master's Thesis

Title: Dynamic Positioning, system identification and control system of marine vessels – based on Balchen model

USN supervisor: David Di Ruscio

External partner: None

Task background:
One of the first mathematical models used for Dynamic Positioning (DP) of ships was the Balchen et al 1980 model. This model was used for building a DP system based on Kalman filtering and optimal control. It is of interest to reconstruct this DP system and possibly make a modified and simplified model based on three double integrators (putting the drag and momentum coefficients equal zero) in surge, sway and yaw (the three moving directions), and from this make a modified DP system.

Task description:

1. Perform a literature research about DP systems of ships.
2. Implement the modified Balchen et al DP system in MATLAB or similar. Some m-file solution proposals exists.
3. Perform simulation experiments.

Student category: IIA students

Is the task suitable for online students (not present at the campus)? Yes

Practical arrangements: None

Supervision:
As a general rule, the student is entitled to 15-20 hours of supervision. This includes necessary time for the supervisor to prepare for supervision meetings (reading material to be discussed, etc).

Signatures:

Supervisor (date and signature): 29.01.2024 David Di Ruscio

Student (write clearly in all capitalized letters): Abu Tammam Bin Nuruddin

Student (date and signature): Tammam 29/01/24

Appendix B

Reduced Balchen model

```
% Reduced low-frequency model without drag and momentam coefficient

function x_dot = nonlinear_model(x, F_t, F_w, M, neu)
    % x:The state vector contains the vessel's position and velocity.
    % F_t: thruster forces in surge, sway, and moment in yaw
    % F_w: wind forces in surge, sway, wind moment in yaw
    % M: initial coefficients from Balchen paper
    % neu: zero mean gaussian white noise
    % parameters
    m1 = M(1);
    m2 = M(2);
    m3 = M(3);
    % state equations
    dx_su_dt = x(4);
    dx_sw_dt = x(5);
    dpsi_dt = x(6);
    dv_su_dt = 1/m1 * (F_w(1) + F_t(1)) + neu(1);
    dv_sw_dt = 1/m2 * (F_w(2) + F_t(2)) + neu(2);
    dv_psi_dt = 1/m3 * (F_w(3) + F_t(3)) + neu(3);

    % Output state derivative
    x_dot = [dx_su_dt; dx_sw_dt; dpsi_dt; dv_su_dt; dv_sw_dt; dv_psi_dt];

end
```

Appendix C

Balchen low frequency non-linear model

```
%%  Balchen non-linear low frequrncy model
function x_dot = Balchen(x, F_t, F_w, F_c, M, D, neu)
%% definitions:
% x = [x_su, x_sw, psi, v_su, v_sw, v_psi]' vessel position and vessel velocities
in surge, sway and yaw
% F_t = [F_t_su, F_t_sw, N_t] thruste forces in surge and sway and and the
thruster moment in yaw
% F_w = [F_w_su, F_w_sw, N_w]' wind forces in surge and sway and wind moment in
yaw (Body frame)
% F_c = [v_c_su, v_c_sw, N_c]' water current velocities in surge and sway and
water current moment in yaw (Body frame)
%D = [d1, d2, d3, d4] drag and moment coefficients
% M = [m1, m2, m3] initial coefficients

%% equations:
dx_su_dt = x(4);
dx_sw_dt = x(5);
dpsi_dt = x(6);
dv_su_dt = - D(1)/M(1)*abs(x(4) - F_c(1))*(x(4) - F_c(1)) + 1/M(1)*(F_w(1)  +
F_t(1)) + neu(1);
dv_sw_dt = - D(2)/M(2) * abs(x(5) - F_c(2))*(x(5) - F_c(2)) + 1/M(2)*(F_w(2) +
F_t(2)) + neu(2);
dv_psi_dt = -D(3)/M(3)*abs(x(6))*x(6) - D(4)/M(3)*abs(x(5) - F_c(2))*(x(5) -
F_c(2)) + 1/M(3)*(F_w(3) + F_t(3) + F_c(3)) + neu(3) ;
x_dot = [dx_su_dt, dx_sw_dt, dpsi_dt, dv_su_dt, dv_sw_dt, dv_psi_dt]';
r = [cos(x(3)) -sin(x(3)) 0; sin(x(3))  cos(x(3)) 0; 0 0 1]; % Matrix to transform
from body frame to NED frame
end
```

Appendix D

Simulation results of Reduced-size MPC where Kalman filter is used for measuring the immeasurable states.

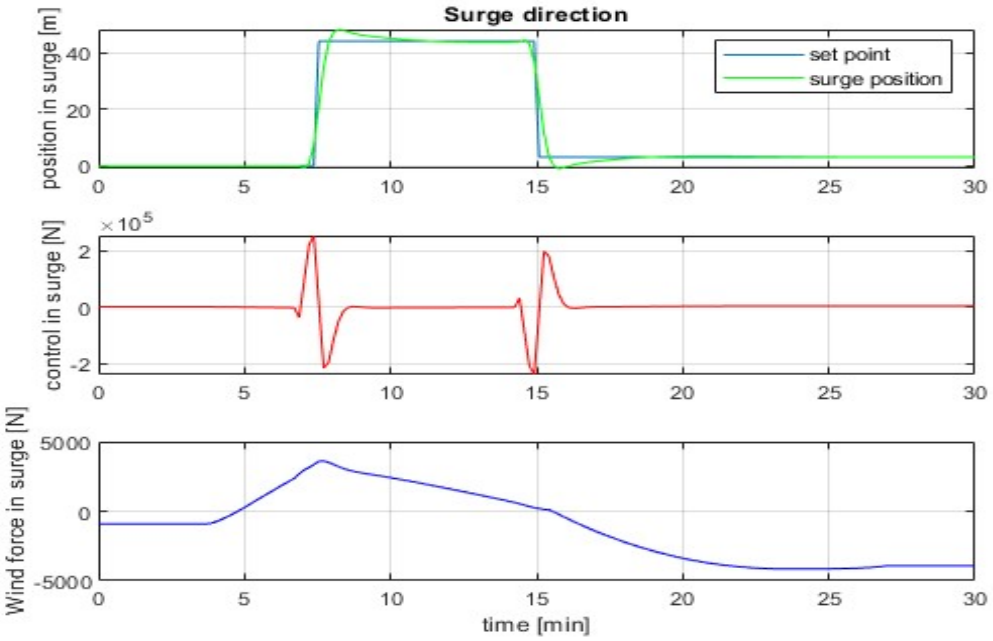

Figure D.1: Output in surge direction when Kalman filter is used for estimating immeasurable states.

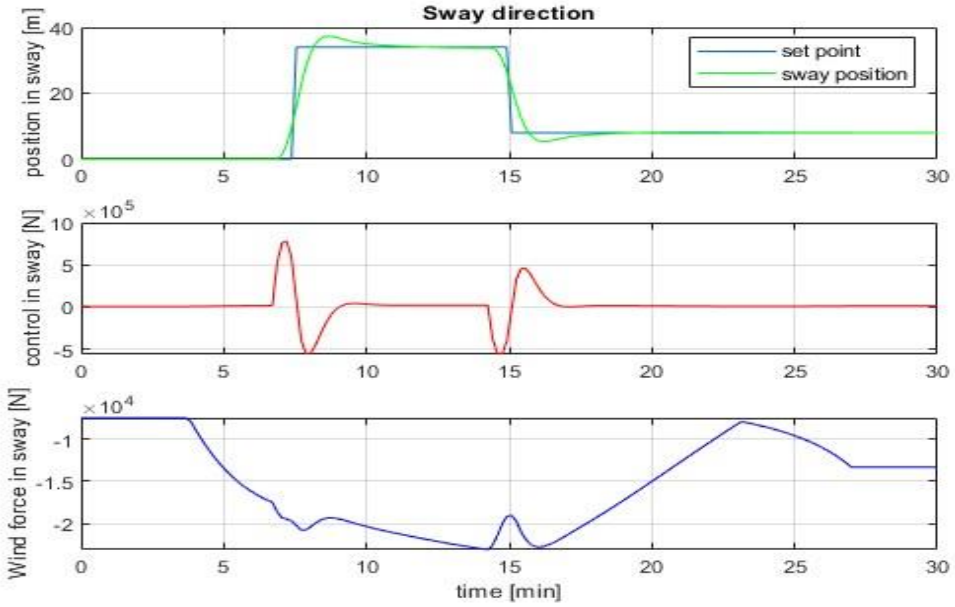

Figure D.2: Output in sway direction when Kalman filter is used for estimating immeasurable states.

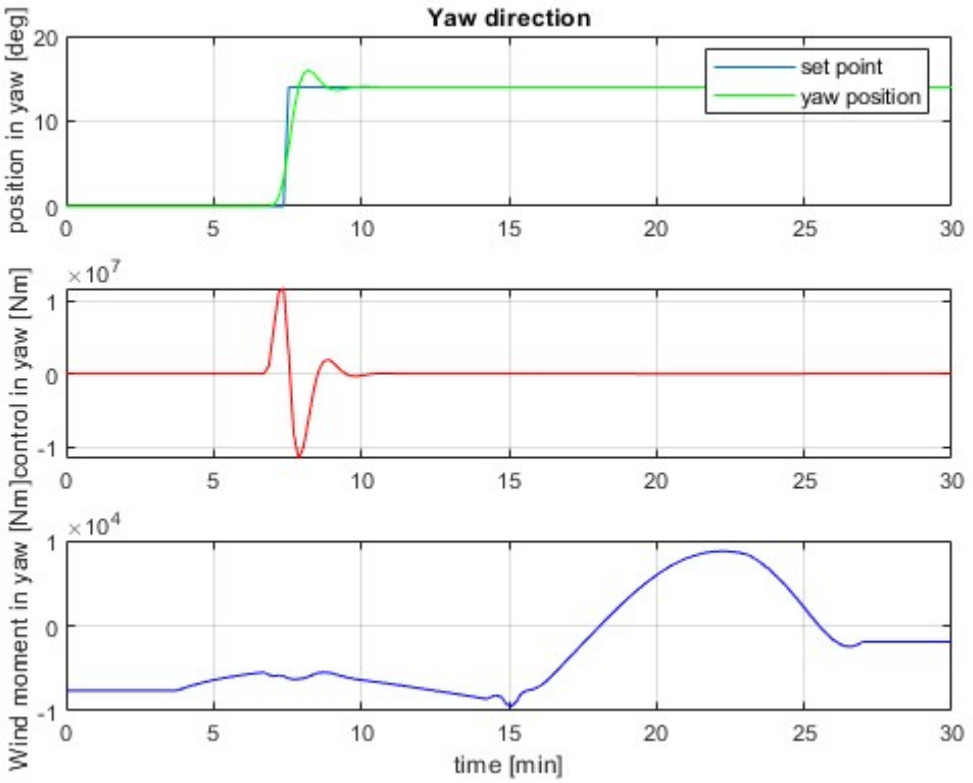

Figure D.3: Vessel heading in yaw direction when Kalman filter is used for estimating immeasurable states.

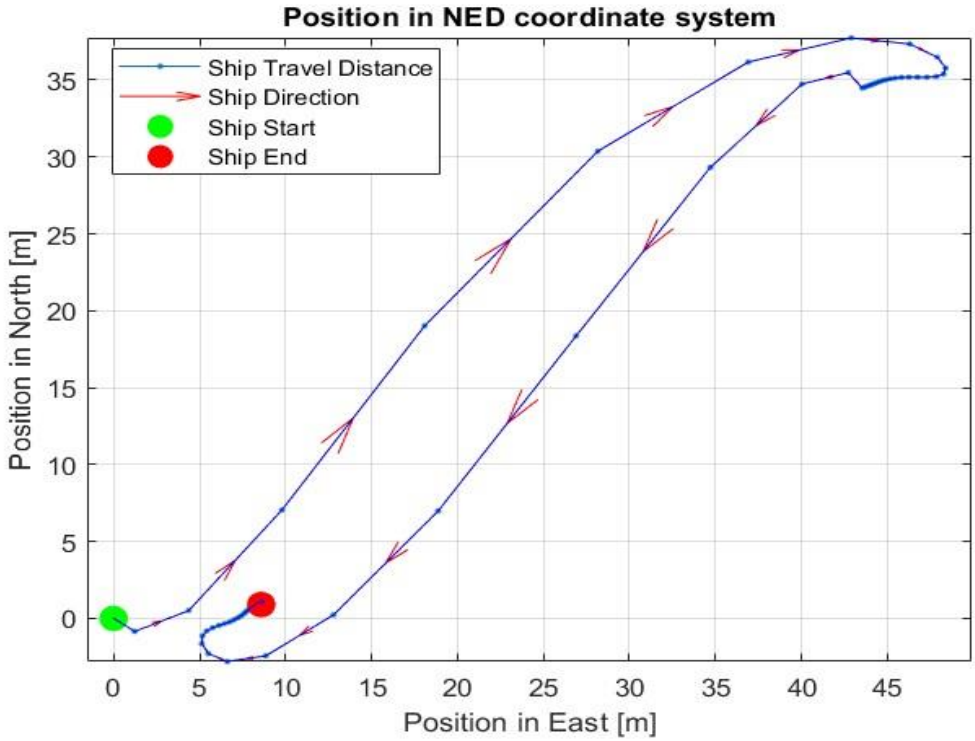

Figure D.4: Vessel's position in NED coordinate system.

Appendix E

Simulation results of Simple MPC with integral action in sway and yaw direction.

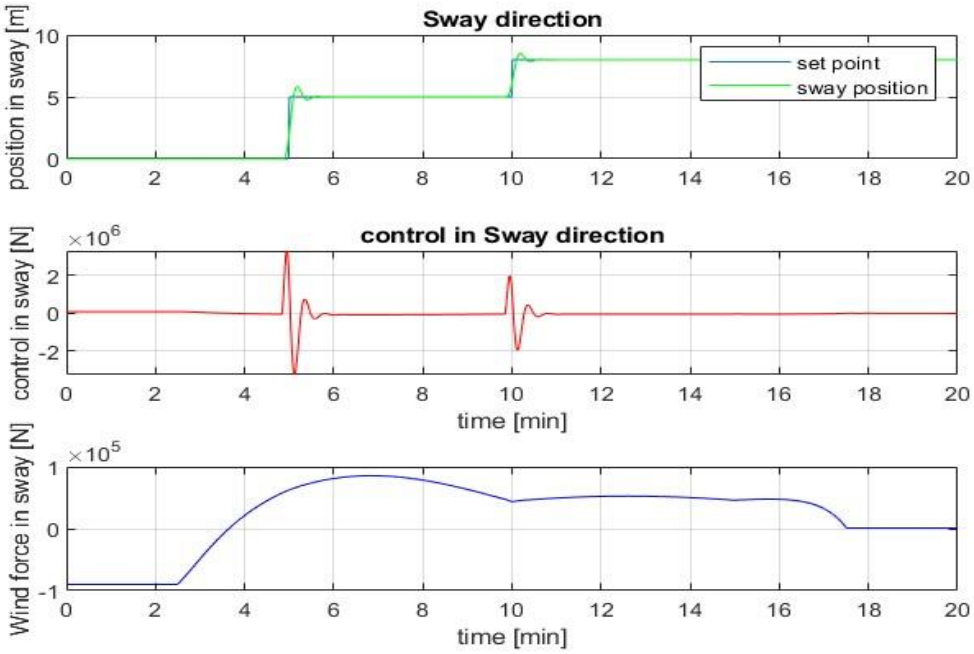

Figure E.1: Vessel direction in sway for Simple MPC with integral action.

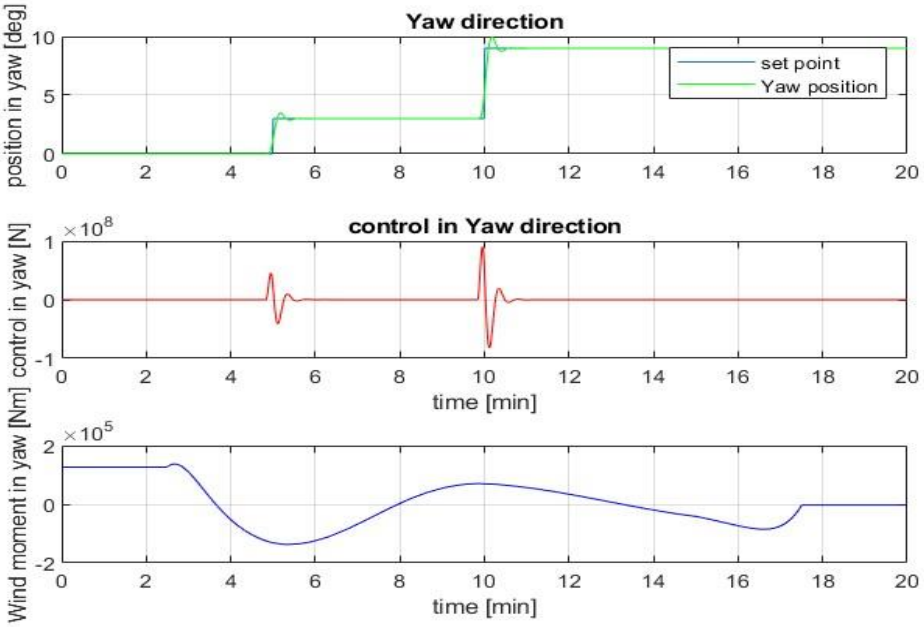

Figure E.2: Vessel heading in yaw direction for Simple MPC with integral action.

Appendix F

Standard MPC algorithm [19]

```
%Standard MPC Algorithm
function [u_opt, x_opt,e_opt, y_opt] = standard_algorithm(A,B,D,r,W,x0,N)
%number of states,system outputs and system inputs:
nx = 6; ny = 3; nu = 3;
%size of the unknow vector z
nz = N*(nx + nu + 2*ny);% N*15
% transformation matrix to tansform from body to NED coordinate system
R = [cos(x0(3)) -sin(x0(3)) 0; sin(x0(3))  cos(x0(3)) 0; 0 0 1];
% weighting matrices
Q = [1e4 0 0; 0 1e4 0; 0 0 1e8]; %tuning weight for errors
P = [1e-5 0 0; 0 1e-6 0; 0 0 1e-15];%tuning weight for inputs

%% Hessian Matrices in the standard form
H11 = kron(eye(N),P);
H22 = zeros(N*nx,N*nx);
H33 = kron(eye(N),Q);
H44 = zeros(N*ny,N*ny);
H = blkdiag(H11,H22,H33,H44);
c = zeros(nz,1);
%% equality constraints
Ae1u = -kron(eye(N),B);
Ae1x = eye(N*nx)-kron(diag(ones(N-abs(-1),1),-1),A);
Ae1e = zeros(N*nx,N*ny);
Ae1y = zeros(N*nx,N*ny);
Fw = W(:,2:N);
Fw = Fw(:);
be1 = [A*x0 + B*W(:,1);kron(eye(N-1),B)*Fw];
Ae2u = zeros(N*ny,N*nu);
Ae2x = -kron(eye(N),D);
Ae2e = zeros(N*ny,N*ny);
Ae2y = eye(N*ny);
be2 = zeros(N*ny,1);
Ae3u = zeros(N*ny,N*nu);
Ae3x = zeros(N*ny,N*nx);
Ae3e = eye(N*ny);
Ae3y = eye(N*ny);
be3 = r(:);
Ae=[Ae1u Ae1x Ae1e Ae1y;
     Ae2u Ae2x Ae2e Ae2y;
     Ae3u Ae3x Ae3e Ae3y];
be=[be1;be2;be3];
%% bounds
ZL=(-Inf*ones(nz,1));
ZU=(Inf*ones(nz,1));
%% solving the QP
options = optimoptions('quadprog','Display','off');
z_opt = quadprog(H,c,[],[],Ae,be,ZL,ZU,x0,options);
%% extract results
Ua = z_opt(1+N*(0) :N*(nu),1); %control inputs
Xa = z_opt (1+N*(nu) :N*(nu+nx),:); %states
Ea = z_opt (1+N*(nu+nx) :N*(nu+nx+ny),:); %error in tracking
Ya = z_opt (1+N*(nu+nx+ny) :N*(nu+nx+ny+ny),:); %outputs
```

```
%Rearrenging the results
u_opt = reshape(Ua,nu,N); %control inputs
x_opt = reshape(Xa,nx,N); %states
e_opt = reshape(Ea,ny,N); %errors
y_opt = reshape(Ya,ny,N); %outputs
end
```

Appendix G

Standard MPC to control the reduce Balchen model

```
% Using standard MPC to control reduced Balchen model
clc;
clear all;
close all;

% Load parameters
parameters;
%% simulation setting
dt = 10; % time step [sec]
t_start = 0; starting time [ min]
t_stop = 15; % stopping time [min]
N = ceil(60*(t_stop - t_start)/ dt ); % steps
L = 10; % prediction horizon

%% setpoint
sp = zeros(3, (N+L));
for m = 1:N+L
   if m <= N/2
        sp(:,m) = [0; 0 ; 0*pi/180];
        sp(:,m) = [6; 6 ; 4*pi/180];
   end
end
% Initialization
x_mod = [sp(:, 1); 0; 0; 0]; % Initial state
x_est = x_mod;

% Disturbances
V_w = slowly(N+L)*20; % Wind speed
gamma = slowly(N+L)*360; % Wind direction
F_w = wind_force(V_w(1), gamma(1), x_mod(3), x_mod(4), x_mod(5), A_F, A_L, L);
u = -F_w;
% Preallocation of arrays
position = zeros(3, N);
position_NED = zeros(3, N);
u_array = zeros(3, N);
W = zeros(3, N);
time = linspace(0, N*dt, N)';
IAE = zeros(3, 1); % Initialize IAE for surge, sway, yaw
TV = zeros(3, 1); % Initialize TV for surge, sway, yaw
%% linear State space model
[A, B, D] = linear_SSM(M, dt);
%% kalman filter gain:
G=0.01*eye(6);
Q_k = diag([0.09, 0.11, 0.09,0.8,0.5,1.5]);
R_k = diag([1e2,1e2,1e2]);
K=dlqe(A,G,D,Q_k,R_k);
% Simulation loop
for k = 1:N
    R = [cos(x_mod(3)) -sin(x_mod(3)) 0; sin(x_mod(3)) cos(x_mod(3)) 0; 0 0 1];
    y = D*x_mod;
    r = sp(:, k+1:min(k+L, N+L));
```

```
    % wind disturbances
    for n = 1:L
        if (k+n-1) <= N
            Fw(:, n) = wind_force(V_w(n+k-1), gamma(n+k-1), x_mod(3), x_mod(4),
x_mod(5), A_F, A_L, L);
        end
    end

    position(:, k) = y;
    position_NED(:, k) = R*y;
    u_array(:, k) = u;
    W(:, k) = wind_force(V_w(k), gamma(k), x_mod(3), x_mod(4), x_mod(5), A_F, A_L,
L);
    [u_opt, x_opt] = optimal_standard(A,B,D,r,Fw,x_est,L);% using Kalman filter to
estimate immeasurable states
    u = u_opt(:, 1);
    dx = nonlinear_model(x_mod, u, W(:, k), M, neu);
    x_mod = x_mod + dt * dx;
    % Updating IAE and TV
    errors = abs(sp(:, k) - y);
    IAE = IAE + errors * dt;
    if k > 1
        TV = TV + abs(u - u_old);
    end
    u_old = u;
    x_est = A * x_est + B * u + B * W(:, k) + K * (y - D * x_est);
end

% Display IAE results
disp('Integrated Absolute Errors (IAE) for Surge, Sway, and Yaw:');
disp(['Surge: ', num2str(IAE(1))]);
disp(['Sway : ', num2str(IAE(2))]);
disp(['Yaw  : ', num2str(IAE(3))]);

disp('Total Value (TV) index for Surge, Sway, and Yaw:');
disp(['Surge TV: ', num2str(TV(1))]);
disp(['Sway TV : ', num2str(TV(2))]);
disp(['Yaw TV  : ', num2str(TV(3))]);

%% plotting results
figure(1),
subplot(311);plot(time/60,sp(1,1:N),time/60,position(1,:),'g-');ylabel('position
in surge [m]');
legend('set point', 'surge position');grid
title('Surge direction');
subplot(312);plot(time/60,u_array(1,:),'r-'); ylabel('control in surge [N]');grid
subplot(313);plot(time/60,W(1,:),'b-');xlabel('time [min]'); ylabel('Wind force in
surge [N]');grid
figure(2),
subplot(311);plot(time/60,sp(2,1:N),time/60,position(2,:),'g-');ylabel('position
in sway [m]');
legend('set point', 'sway position');grid
title('Sway direction');
subplot(312);plot(time/60,u_array(2,:),'r-'); ylabel('control in sway [N]');grid
subplot(313);plot(time/60,W(2,:),'b-');xlabel('time [min]'); ylabel('Wind force in
sway [N]');grid
figure(3),
```

```
subplot(311);plot(time/60,sp(3,1:N)*180/pi,time/60,position(3,:)*180/pi,'g-
');ylabel('position in yaw [deg]');
legend('set point', 'yaw position');grid
title('Yaw direction');
subplot(312);plot(time/60,u_array(3,:),'r-'); ylabel('control in yaw [Nm]');grid
subplot(313);plot(time/60,W(3,:),'b-');xlabel('time [min]'); ylabel('Wind moment
in yaw [Nm]');grid
figure(4);
plot_dir(position_NED(2,:)', position_NED(1,:)');
xlabel('Position in East [m]');
ylabel('Position in North [m]');
title('Position in NED coordinate system');
hold on;
plot(position_NED(2,1), position_NED(1,1), 'go', 'MarkerSize', 10,
'MarkerFaceColor', 'g');
plot(position_NED(2,end), position_NED(1,end), 'ro', 'MarkerSize', 10,
'MarkerFaceColor', 'r');
quiver(position_NED(2,end-1), position_NED(1,end-1), position_NED(2,end)-
position_NED(2,end-1), position_NED(1,end)-position_NED(1,end-1), 0, 'Color', 'r',
'LineWidth', 2)
plot(position_NED(2,:), position_NED(1,:), 'b-');
hold off;
legend('Ship Travel Distance', 'Ship Direction', 'Ship Start', 'Ship End');
grid on;
```

Appendix H

The optimal feedback matrix *G* is calculated using the *dlqdu_pi*

```
function [G1,G2]= dlqdu_pi(A,B,D,Q,Rw);
% DLQDU_PI syntax
% [G1,G2,At,Bt,Dt]=dlqdu_pi(A,B,D,Q,R);
% Purpose
% Compute LQ-optimal feedback matrices
% G1 and G2 in the controller
% u=u+G1*(x-x_old)+G2*(y_old-r);
% On input
% A,B,D- discrete state space model matrices.
% Q - Weighting matrix for the output y_k.
% R - Weighting matrix for the control
% increment, Delta u_k=u_k-u_(k-1).
% On output
% G1 and G2 - Matrices in LQ controller
% At, Bt, Dt - Matrices in augmented model
% Make augmented state space model
% matrices.
nx=size(A,1); nu=size(B,2); ny=size(D,1);
At=[A,zeros(nx,ny);D,eye(ny,ny)];
Bt=[B;zeros(ny,nu)];
Dt=[D,eye(ny,ny)];
Qt=Dt'*Q*Dt;
% Solve Riccati-equation and feedback matrices
[K,Rr]=dlqr(At,Bt,Qt,Rw);
G=-K;
G1=G(:,1:nx); G2=G(:,nx+1:nx+ny);
% END dlqdu_pi
```

Appendix I

LQ Optimal control with integral action to control reduce Balchen model

```
%LQ optimal control with integral action
clc
clear all
close all
parameters;

%% simulation settings
dt = 10; % time step [sec]
t_start = 0; % starting time [ min]
t_stop = 15; %  stopping time [min]
N = ceil(60*(t_stop - t_start)/ dt ); %  steps
L = 10; % prediction horizon length

%% setpoint
sp = zeros(3, (N));
for m = 1:N
   if m <= N/2
        sp(:,m) = [0; 0 ; 0*pi/180];
   else
        sp(:,m) = [6; 6 ; 4*pi/180];
   end
end
%% Disturbances:
V_w = slowly(N)*20;% wind speed as a random number
gamma = slowly(N)*360;% wind direction as a random number

%% initialization:
x0 = [sp(1,1) sp(2,1) sp(3,1) 0 0 0]';
x = x0;
F_w = wind_force(V_w(1,1), gamma(1,1), x(3),x(4), x(5), A_F, A_L, L);
u = -F_w;
%% Preallocation of arrays for plotting:
y_array = zeros(3,N);
u_array= zeros(3,N);
Fw = zeros(3,N);
time = linspace(0, N*dt, N)';
position = zeros(3,N);
position_NED = zeros(3,N);
IAE = zeros(3, 1); % Initialize IAE for surge, sway, yaw
TV = zeros(3, 1); % Initialize TV for surge, sway, yaw
%% state space model
[ A, B, D] = linear_SSM(M, dt);
%% calculating control gaing
Q = [1e2 0 0; 0 1e0 0; 0 0 1e3]; %tuning weight for errors
P = [1e-7 0 0; 0 1e-7 0; 0 0 1e-10] ;%tuning weight for inputs
[G1,G2]= dlqdu_pi(A,B,D,Q,P);
%% kalman filter gain
G=0.01*eye(6);
Q_k = diag([0.09, 0.11, 0.09,0.8,0.5,1.5]);
R_k = diag([1e3,1e3,1e3]);
K=dlqe(A,G,D,Q_k,R_k);
```

```
%% simulation loop:
x_est = x;
y = D*x;
x_old = x;
y_old = y;
for i = 1:N
    R = [cos(x(3)) -sin(x(3)) 0; sin(x(3))  cos(x(3)) 0; 0 0 1];
    % wind disturbances
    F_w = wind_force(V_w(i,1), gamma(i,1), x(3),x(4), x(5), A_F, A_L, L);
    y = D*x;
    y_array(:,i) = y;
    y_NED(:,i)= R*y;
    position(:,i) = y;
    position_NED(:,i) = R*y;
    u_array(:,i) = u;
    Fw(:,i) = F_w;
    u = u + G1*(x_est-x_old) + G2*(y_old - sp(:,i));
    x_old = x_est;
    y_old = y;
    dx = nonlinear_model(x, u, F_w, M, neu);
    x = x + dt*dx; % updating
    x_est = A*x_est + B*u + B*F_w+ K*(y- D*x_est); % predicting next state
     IAE = IAE + abs(sp(:, i) - y) * dt;
         if i > 1
        TV = TV + abs(u - u_old); % sum of absolute changes in control signals
    end
    u_old = u;
end
%% Display IAE and TV results
disp('Integrated Absolute Errors (IAE) for Surge, Sway, and Yaw:');
disp(['Surge IAE: ', num2str(IAE(1))]);
disp(['Sway IAE: ', num2str(IAE(2))]);
disp(['Yaw IAE: ', num2str(IAE(3))]);
disp('Total Value (TV) index for Surge, Sway, and Yaw:');
disp(['Surge TV: ', num2str(TV(1))]);
disp(['Sway TV : ', num2str(TV(2))]);
disp(['Yaw TV  : ', num2str(TV(3))]);
%% plotting
figure(1),
subplot(311);plot(time/60,sp(1,:),time/60,y_array(1,:),'g-');ylabel('position in
surge [m]');
title('Surge direction');legend('set point','surge position');grid
subplot(312);plot(time/60,u_array(1,:),'r-'); ylabel('control in surge [N]');grid
subplot(313);plot(time/60,Fw(1,:),'b-');xlabel('time [min]'); ylabel('Wind force
in surge [N]');grid
figure(2),
subplot(311);plot(time/60,sp(2,:),time/60,y_array(2,:),'g-');ylabel('position in
sway [m]');
title('Sway direction');legend('set point','sway position');grid
subplot(312);plot(time/60,u_array(2,:),'r-');xlabel('time [min]'); ylabel('control
in sway [N]');grid
subplot(313);plot(time/60,Fw(2,:),'b-');xlabel('time [min]'); ylabel('Wind force
in sway [N]');grid
figure(3),
subplot(311);plot(time/60,sp(3,:)*180/pi,time/60,y_array(3,:)*180/pi,'g-
');ylabel('position in yaw [deg]');
title('Yaw direction');legend('set point','Yaw position');grid
```

```
subplot(312);plot(time/60,u_array(3,:),'r-');xlabel('time [min]'); ylabel('control
in yaw [Nm]');grid
subplot(313);plot(time/60,Fw(3,:),'b-');xlabel('time [min]'); ylabel('Wind moment
in yaw [Nm]');grid
figure(4);
plot_dir(position_NED(2,:)', position_NED(1,:)');
xlabel('Position in East [m]');
ylabel('Position in North [m]');
title('Position in NED coordinate system');
hold on;
plot(position_NED(2,1), position_NED(1,1), 'go', 'MarkerSize', 10,
'MarkerFaceColor', 'g');
plot(position_NED(2,end), position_NED(1,end), 'ro', 'MarkerSize', 10,
'MarkerFaceColor', 'r');
quiver(position_NED(2,end-1), position_NED(1,end-1), position_NED(2,end)-
position_NED(2,end-1), position_NED(1,end)-position_NED(1,end-1), 0, 'Color', 'r',
'LineWidth', 2)
plot(position_NED(2,:), position_NED(1,:), 'b-');
hold off;
legend('Ship Travel Distance', 'Ship Direction', 'Ship Start', 'Ship End');
grid on;
```

Appendix J

MATLAB code of simple MPC to control the vessel

```
%% implementing simple MPC developed by David Di Ruscio in a DP system
clc
clear all
close all
parameters;

%% simulation settings
dt = 1; % time step [s]
t_start = 0; % simulating starting time [ min]
t_stop = 40; %simulating stopping time[ min]
N = ceil(60*(t_stop - t_start)/ dt ); % simulating steps
L =10;% prediction horizon
%% setpoint
sp = zeros(3, (N+L));
for m = 1:N+L
   if m<=N/4
        sp(:,m) = [0; 0 ; 0*pi/180];% north, east, yaw setpoint
   elseif m> N/4 && m<=N/2
        sp(:,m)= [3 ; 5 ; 3*pi/180];
   elseif m>N/2
        sp(:,m)= [1 ; 8; 9*pi/180];
   end
end
%% Disturbances:
V_w = slowly(N)* 20;% wind speed in NED coordinate system as a random number
gamma = slowly(N)*360;% wind direction in NED coordinate system as a random number
%% initialization:
x0 = [sp(1,1) sp(1,2) sp(1,3) 0 0 0]';
x = x0;
F_w = wind_force(V_w(1), gamma(1), x(3),x(4), x(5), A_F, A_L, L);
u = -F_w;
%% Preallocation of arrays for plotting:
y_pos = zeros(3,N);
u_array= zeros(3,N);
Fw = zeros(3,N);
y_NED= zeros(3,N);
time = linspace(0, N*dt, N)';
position = zeros(3,N);
position_NED = zeros(3,N);
%% calculating linear SSM matrices
[ A, B, D] = linear_SSM ( M, dt);
%% calculating kalman filter gain:
G=0.01*eye(6);
Q_k = diag([0.09, 0.11, 0.09,0.8,0.5,1.5]);
R_k = diag([1e2,1e2,1e2]);
K=dlqe(A,G,D,Q_k,R_k);
%%
Q = [1e4 0 0; 0 1e5 0; 0 0 1e9]; %tuning weight for errors
P = [1e-6 0 0; 0 1e-7 0; 0 0 1e-15] ;%tuning weight for inputs
[HdL,OL,OLB]=ss2h(A,B,D,zeros(3,3),L,0);
FL=[OLB HdL];
Qt=q2qt(Q,L);
Rt=q2qt(P,L);
```

```
H=FL'*Qt*FL+Rt;
%%
x_est = x;
%% simulation loop:
for k = 1:N
    % transformation matrix to tansform from body to NED coordinate system
    R = [cos(x(3)) -sin(x(3)) 0; sin(x(3))  cos(x(3)) 0; 0 0 1];
    F_w = wind_force(V_w(k), gamma(k), x(3),x(4), x(5), A_F, A_L, L);
    y = D*x;
    y_pos(:,k) = y;
    y_NED(:,k) = R*y;
    position(:,k) = y;
    position_NED(:,k) = R*y;
    u_array(:,k) = u(:,1);
    Fw(:,k) = F_w;
    pL=OL*A*x_est;
    r = sp(:,k+1:L+k);
    r = r(:);
    f=FL'*Qt*(pL-r);
    u = quadprog(H,f);
    u = reshape(u,3,L); %arranged control inputs
    dx = nonlinear_model(x, u(:,1), F_w, M ,neu);% update the system by using
first calculated optimal control
    x= x+ dt*dx;
    % using current time output to estimate states by kalman filter
     x_est = A*x_est + B*u(:,1)+B*F_w+ K*(y - D*x_est);
end
%% plotting
figure(1),
subplot(311);plot(time/60,sp(1,1:N),time/60,y_pos(1,:),'g-');ylabel('position in
surge [m]');
title('Surge direction');legend('set point','surge position');grid
subplot(312);plot(time/60,u_array(1,:),'r-'); ylabel('control in surge [N]');grid
subplot(313);plot(time/60,Fw(1,:),'b-');xlabel('time [min]'); ylabel('Wind force
in surge [N]');grid
figure(2),
subplot(311);plot(time/60,sp(2,1:N),time/60,y_pos(2,:),'g-');ylabel('position in
sway [m]');
title('Sway direction');legend('set point','sway position');grid
subplot(312);plot(time/60,u_array(2,:),'r-'); ylabel('control in sway [N]');grid
subplot(313);plot(time/60,Fw(2,:),'b-');xlabel('time [min]'); ylabel('Wind force
in sway [N]');grid
figure(3),
subplot(311);plot(time/60,sp(3,1:N)*180/pi,time/60,y_pos(3,:)*180/pi,'g-
');ylabel('position in yaw [deg]');
title('Yaw direction');legend('set point','Yaw position');grid
subplot(312);plot(time/60,u_array(3,:),'r-'); ylabel('control in yaw [Nm]');grid
subplot(313);plot(time/60,Fw(3,:),'b-');xlabel('time [min]'); ylabel('Wind moment
in yaw [Nm]');grid
figure(4);
plot_dir(position_NED(2,:)', position_NED(1,:)');
xlabel('Position in East [m]');
ylabel('Position in North [m]');
title('Position in NED coordinate system');
hold on;
plot(position_NED(2,1), position_NED(1,1), 'go', 'MarkerSize', 10,
'MarkerFaceColor', 'g');
```

```
plot(position_NED(2,end), position_NED(1,end), 'ro', 'MarkerSize', 10,
'MarkerFaceColor', 'r');
quiver(position_NED(2,end-1), position_NED(1,end-1), position_NED(2,end)-
position_NED(2,end-1), position_NED(1,end)-position_NED(1,end-1), 0, 'Color', 'r',
'LineWidth', 2)
plot(position_NED(2,:), position_NED(1,:), 'b-');
hold off;
legend('Ship Travel Distance', 'Ship Direction', 'Ship Start', 'Ship End');
grid on;
```

Appendix K

MATLAB code of simple MPC with integarl action to control the vessel .

```
%% implementing simple MPC with integral action developed by David Di Ruscio in a
DP control system
clc
clear all
close all
parameters;
%% simulation settings
dt = 1; % time step [s]
t_start = 0; % simulating starting time [ min]
t_stop = 20; %simulating stopping time[ min]
N = ceil(60*(t_stop - t_start)/ dt ); % simulating steps
L =10;% prediction horizon
%% setpoint
sp = zeros(3, (N+L));
for m = 1:N+L
   if m<=N/4
        sp(:,m) = [0; 0 ; 0*pi/180];% north, east, yaw setpoint
   elseif m> N/4 && m<=N/2
        sp(:,m)= [3 ; 5 ; 3*pi/180];
   elseif m>N/2
        sp(:,m)= [1 ; 8; 9*pi/180];
   end
end
%% Disturbances:
V_w = slowly(N)* 20;% wind speed in NED coordinate system as a random number
gamma = slowly(N)*360;% wind direction in NED coordinate system as a random number
%% initialization:
x0 = [sp(1,1) sp(1,2) sp(1,3) 0 0 0]';
x = x0;
F_w = wind_force(V_w(1), gamma(1), x(3),x(4), x(5), A_F, A_L, L);
u = -F_w;
%% Preallocation of arrays for plotting:
x_pos = zeros(6,N);
u_array= zeros(3,N);
Fw = zeros(3,N);
y_array = zeros(3,N);
time = linspace(0, N*dt, N)';
position = zeros(3,N);
position_NED = zeros(3,N);
%% Calculating linear SSM matrices
[ A, B, D] = linear_SSM ( M, dt);
nx = size(A,1); % number of system states
nu = size(B,2);% number of system inputs
ny = size(D,1); % number of system outputs
%% calculating kalman filter gain:
G=0.01*eye(6);
Q_k = diag([0.09, 0.11, 0.09,0.8,0.5,1.5]);
R_k = diag([1e2,1e2,1e2]);
K=dlqe(A,G,D,Q_k,R_k);
Q = [1e4 0 0; 0 1e5 0; 0 0 1e9]; %tuning weight for errors
P = [1e-6 0 0; 0 1e-7 0; 0 0 1e-15];%tuning weight for inputs
```

```
%% augmented system matrices:
At = [A zeros(nx,ny); D eye(ny,ny)];
Bt= [B ; zeros(ny,nu)];
Dt = [D eye(ny,ny)];
[HdL,OL,OLB]=ss2h(At,Bt,Dt,zeros(ny,nu),L,0);
FL=[OLB HdL];
Qt=q2qt(Q,L);
Rt=q2qt(P,L);
H=FL'*Qt*FL+Rt;
%%
y = D*x;
x_est = x;
x_old = x_est;
y_old = y;
xt = [x-x_old;y_old];
%% simulation loop:
for k = 1:N
    % transformation matrix to tansform from body to NED coordinate system
    R = [cos(x(3)) -sin(x(3)) 0; sin(x(3))  cos(x(3)) 0; 0 0 1];
    F_w = wind_force(V_w(k), gamma(k), x(3),x(4), x(5), A_F, A_L, L);
    y = D*x;
    x_pos(:,k) = x;
    position(:,k) = y;
    position_NED(:,k) = R*y;
    u_array(:,k) = u(:,1);
    y_array(:,k) = R*y;
    Fw(:,k) = F_w;
    pL=OL*At*xt;
    r = sp(:,k+1:L+k);
    r = r(:);
    f=FL'*Qt*(pL-r);
    du = quadprog(H,f);
    du = reshape(du,3,L); %arranged control inputs
    u = u + du(:,1);
    x_old = x_est;
    y_old = y;

    dx = nonlinear_model(x, u, F_w, M, neu);
    x= x+ dt*dx;
    % using current time output to estimate states by kalman filter
    x_est = A*x_est + B*u(:,1)+B*F_w+ K*(y - D*x_est);
    xt = [x_est-x_old;y_old];
end
%% plotting
figure(1),
subplot(311);plot(time/60,sp(1,1:N),time/60,x_pos(1,:),'g-');ylabel('position in
surge [m]');
title('Surge direction');
legend('set point','surge position');
grid
subplot(312);plot(time/60,u_array(1,:),'r-');xlabel('time [min]'); ylabel('control
in surge [N]');
title('control in Surge direction');
grid
subplot(313);plot(time/60,Fw(1,:),'b-');xlabel('time [min]'); ylabel('Wind force
in surge [N]');grid
figure(2),
```

```
subplot(311);plot(time/60,sp(2,1:N),time/60,x_pos(2,:),'g-');ylabel('position in
sway [m]');
title('Sway direction');
legend('set point','sway position');
grid
subplot(312);plot(time/60,u_array(2,:),'r-');xlabel('time [min]'); ylabel('control
in sway [N]');
title('control in Sway direction');
grid
subplot(313);plot(time/60,Fw(2,:),'b-');xlabel('time [min]'); ylabel('Wind force
in sway [N]');grid
figure(3),
subplot(311);plot(time/60,sp(3,1:N)*180/pi,time/60,x_pos(3,:)*180/pi,'g-
');ylabel('position in yaw [deg]');
title('Yaw direction');
legend('set point','Yaw position');
grid
subplot(312);plot(time/60,u_array(3,:),'r-');xlabel('time [min]'); ylabel('control
in yaw [N]');
title('control in Yaw direction');
grid
subplot(313);plot(time/60,Fw(3,:),'b-');xlabel('time [min]'); ylabel('Wind moment
in yaw [Nm]');grid
figure(4);
plot_dir(position_NED(2,:)', position_NED(1,:)');
xlabel('Position in East [m]');
ylabel('Position in North [m]');
title('Position in NED coordinate system');
hold on;
plot(position_NED(2,1), position_NED(1,1), 'go', 'MarkerSize', 10,
'MarkerFaceColor', 'g');
plot(position_NED(2,end), position_NED(1,end), 'ro', 'MarkerSize', 10,
'MarkerFaceColor', 'r');
quiver(position_NED(2,end-1), position_NED(1,end-1), position_NED(2,end)-
position_NED(2,end-1), position_NED(1,end)-position_NED(1,end-1), 0, 'Color', 'r',
'LineWidth', 2)
plot(position_NED(2,:), position_NED(1,:), 'b-');
hold off;
legend('Ship Travel Distance', 'Ship Direction', 'Ship Start', 'Ship End');
grid on;
```