DOCKING PROBLEMS OF SEA SURFACE VEHICLES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

İSMAİL ÇAĞDAŞ YILMAZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JANUARY 2019

Approval of the thesis:

DOCKING PROBLEMS OF SEA SURFACE VEHICLES

submitted by **İSMAİL ÇAĞDAŞ YILMAZ** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Tolga Çiloğlu
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Mehmet Kemal Leblebicioğlu
Supervisor, **Electrical and Electronics Eng. Dept., METU** _____

Examining Committee Members:

Prof. Dr. Mehmet Önder Efe
Computer Eng. Dept., Hacettepe University _____

Prof. Dr. Mehmet Kemal Leblebicioğlu
Electrical and Electronics Eng. Dept., METU _____

Prof. Dr. Asım Egemen Yılmaz
Electrical and Electronics Eng. Dept., Ankara University _____

Prof. Dr. Çağatay Candan
Electrical and Electronics Eng. Dept., METU _____

Assist. Prof. Dr. Mustafa Mert Ankaralı
Electrical and Electronics Eng. Dept., METU _____

Date: 31/01/2019

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: İsmail Çağdaş Yılmaz

Signature :

ABSTRACT

DOCKING PROBLEMS OF SEA SURFACE VEHICLES

Yılmaz, İsmail Çağdaş
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Mehmet Kemal Leblebicioğlu

January 2019, 132 pages

This thesis covers parallel docking (parallel parking) problem for unmanned surface vehicles (USVs). First, a mathematical model for a USV with two propellers is constructed by using Newton-Euler formulation. Kinematics and dynamic equations create 6 degrees-of-freedom model. A hierarchical motion control approach is implemented on this model. Two kinds of guidance laws, line-of-sight (LOS), and pure pursuit (PP) are employed for way-point travelling at the strategic level of the hierarchy. At the control allocation level, a finite horizon model predictive controller (MPC) and a cascaded PID controller are designed and tuned to optimize path following the performance. These guidance and control methods are implemented for parallel docking, which is treated as a way-point generation problem. Path generation for docking is handled in two stages. In the first stage, by solving a constrained optimal control, a path is found which provides that the vehicle reaches the port of the parking region with minimum control demands. By using a continuous curvature path function, the vehicle is taken from port to parking slot. The path following and energy consumption performances of the USV under the parallel docking manoeuvres are evaluated for different combinations of guidance laws and controller designs

at the second stage. Finally, experimental validation has been realized on a scaled boat with model predictive control and pure-pursuit guidance methods.

Keywords: unmanned surface vehicles, mathematical modeling, line-of-sight guidance, pure-pursuit guidance, model predictive control, cascaded PID control, parallel docking, constrained optimal control

ÖZ

SU ÜSTÜ ARAÇLARI İÇİN KENETLENME PROBLEMİ

Yılmaz, İsmail Çağdaş
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Mehmet Kemal Leblebicioğlu

Ocak 2019, 132 sayfa

Bu tez, insansız yüzey araçları (İYA) için paralel kenetlenme (paralel park etme) problemini kapsamaktadır. İlk olarak, iki tane pervanesi bulunan bir İYA için Newton-Euler metodu kullanılarak bir matematiksel model elde edilir. Kinematik ve dinamik denklemler kullanılarak 6 serbestlik derecesine sahip bir model oluşturulur. Bu model üzerinde bir hiyerarşik hareket kontrol yaklaşımı uygulanır. Hiyerarşinin stratejik seviye kısmında iki çeşit güdüm yöntemi; görüş hattı (GH) ve katıksız takip (PP) yol noktası takibi için kullanılmaktadır. Hiyerarşinin kontrol dağıtım seviyesinde ise, sonlu ufka sahip model öngörülü denetleyici ve bir katlı PID denetleyici tasarlanmış ve aracın yol izleme performansını en iyilemek için parametreleri ayarlanmıştır. Daha sonra, bu güdüm ve kontrol metotları yol noktaları oluşturma problemi olarak görülen paralel kenetlenmede uygulanır. Kenetlenme için yol oluşturma iki aşamada ele alınır. İlk aşamada, kısıtlı bir en iyi kontrol çözümü aracın park bölgesinin kıyısına gelmesi sağlanır. İkinci aşamada, sürekliliğe sahip eğimli bir yol fonksiyonu kullanarak, araç kıyıdan park alanına yöneltilir. İYA'nın yol takip ve enerji harcama performansı farklı güdüm ve kontrol yöntemlerinin kombinasyonları ile değerlendirilir. Son olarak, model öngörülü denetleyeci ve katıksız takip güdüm yöntemini kullanılarak

ölçeklendirilmiş bir tekne üzerinde deneysel doğrulama gerçekleştirilmiştir.

Anahtar kelimeler: insansız yüzey araçları, matematiksel modelleme, görüş hattı güdümü, katıksız takip güdümü, model öngürülü denetleyici, katlı PID denetleyici, paralel kenetlenme, kısıtlı en iyi kontrol

To my family

ACKNOWLEDGMENTS

I would like to express my special thanks to my supervisor Prof. Dr. Mehmet Kemal Leblebicioğlu for his remarkable guidance, valuable advices, and supports during the this study. His feedbacks and comments broadened my horizon.

I would like to present my gratitude to Dr. Kenan Ahıska who helped me like a co-supervisor throughtout the thesis study. Without his help and contribution, this study could not be completed.

I would like to express my very great appreciation to Sıdıka Bengür and Mustafa Burak Gürcan for their support and tolerance.

I owe a very important debt to Murat Kumru and İzzet Kağan Erünsal for their master studies and papers.

I am so grateful to my colleagues Mete Erbay, Gökhan Özdoğan, Özkan Yılmaz and my friend Hasan Mercimek for their kind help and contributions.

I would also like to thank Ali Galip Beydilli and employees of Desistek Inc. for their technical solutions.

Advice and comments given by Onur İlhan, Yaser Yurtcan and Raha Shabani have been a great help in software development and hardware setup.

I would like to acknowledge my company ASELSAN Inc. for financial and technical supports especially for experimental validation and conference supports.

The last but not least, I am really thankful to my family, notably my mother Hafize Yılmaz, for their endless support and always believing in me.

TABLE OF CONTENTS

APPENDICES

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ALGORITHMS

ALGORITHMS

LIST OF ABBREVIATIONS

ASV	Autonomous Surface Vehicle
DoF	Degree of Freedom
COLREGs	International Regulations for Avoiding Collisions at Sea
ESC	Electronic Speed Controller
GA	Genetic Algorithm
IMU	Inertial Measurement Unit
LMPC	Linear Model Predictive Control
LOS	Line-of-Sight
MPC	Model Predictive Control
NGC	Navigation, Guidance and Control (NGC)
NED	North-East-Down
OS	Operating System
PID	Proportional Integral Derivative
PSO	Particle Swarm Optimization
PWM	Pulse with Modulation
PP	Pure Pursuit
RPM	Revolutions Per Minute
USV	Unmanned Surface Vehicles
ZOH	Zero Order Hold

CHAPTER 1

INTRODUCTION

1.1 Motivation of the Study

Demands of full autonomy are increasing for unmanned surface vehicles (USVs), in other words, autonomous surface vehicles (ASVs), with each passing day which are employed in observing environmental changes and abnormalities, handling military tasks, conducting scientific research in lakes, seas, and oceans [32]. More robust and reliable systems have been made thanks to ever-improving controller designs which reduce human intervention on surface crafts to the point of disappearing.

Design and implementation procedures of navigation, guidance and control (NGC) systems for a USV have become crucial according to the International Regulations for Avoiding Collisions at Sea (COLREGs) [13]. COLREGs defines possible scenarios for collisions and describes some manoeuvring techniques to prevent potential crashes [3]. Errors and failures such as amateur manoeuvring by human conduct cause marine collisions and accidents the percentages of which are approximately estimated to be between 89% and 96% [43]. All of these issues can be prevented or minimized by making an autonomous vehicle with the aid of proven, commercially available, compact hardware equipment which includes global navigation satellite system (GNSS) receiver, inertial measurement units (IMUs), communication network etc. [33].

One of the challenging aspects of designing and modeling a USV is to deal with coupled and nonlinear dynamics. Some physical parameters are generally not known in a precise manner [36]. However, their values can be assumed to be in a certain range by considering dependencies of model parameters on each other. Consequently,

this difficulty puts NGC systems in an important role.

The marine research community has furthered rapid improvement and state-of-art control applications to make the autonomous USVs for many years. These applications which depend on the vehicle dynamic model, offer robust, stable and accident-free motion [13].

In this study, the main aim is to dock a USV in parallel to a parking spot autonomously. To achieve this purpose, it is decided that effective autopilot and guidance algorithms are employed for motion control. These challenging algorithms offer highly suitable motion guarantees successful manoeuvring for USVs. The proposed methods are later verified on a scaled boat in physical experiments.

1.2 Literature Review

Great number of kinetics and kinematics model publications are available in the literature [28], [19], [22], [11]. A mathematical model which is composed of kinematics and dynamics equations is important for sophisticated control design. Newton's (second) law of motion in 6 degree-of-freedom (DoF) are the first approach to obtain dynamics of the model as follows:

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} = \sum_{j=1}^{k} \boldsymbol{F}_j \tag{1.1}$$

$\boldsymbol{M}$ represents the mass (inertia) matrix of the system. The generalized acceleration vector which is the derivation of linear and angular velocities with respect to time is denoted by $\dot{\boldsymbol{\nu}}$. In (1.1), there is k number of forces and moments which are acting on the vehicle. Each force and moment vector is indicated by six components:

$$\boldsymbol{F}_j = [X_j \; Y_j \; Z_j \; K_j \; M_k \; N_k]^T \tag{1.2}$$

Generalized position vector is given in (1.3), which is composed of position, Euler angles and velocity vector, (1.4), whose elements are linear and angular velocities which are represented by making use of SNAME notation [48]:

$$\boldsymbol{\eta} \triangleq [x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi]^T \tag{1.3}$$

$$\boldsymbol{\nu} \triangleq [u \;\; v \;\; w \;\; p \;\; q \;\; r]^T \tag{1.4}$$

Keeping (1.1) in mind, Fossen [20] has developed a new demonstration for mathematical modeling of underwater vehicles by using Newton-Euler equations. This representation which is also used for surface vehicles with some changes, is based on the derivation of robot dynamics [15]. Developed kinematic equations are represented in vector form in the following equation:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} \tag{1.5}$$

(1.5) includes Coriolis and centripetal matrix, $\boldsymbol{C}$, in addition to inertia matrix $\boldsymbol{M}$. $\boldsymbol{\tau}$ vector indicates torques acting on the vehicle and $\boldsymbol{q}$ is the joint angle vector. The model has been developed with further research and assumptions to improve the controller design. The followings are assumed in the modeling procedure: mass of the USV is uniformly distributed along the body (1), centre of gravity coincides with body-fixed frame (2), motions of a USV are in an ideal fluid (3), sway-yaw and surge-sway dynamics are decoupled (4) [47], [32]. The final form of the robot-like dynamics representation is written in (1.6):

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) + \boldsymbol{g}_0 = \boldsymbol{\tau} \tag{1.6}$$

where $\boldsymbol{D}(\boldsymbol{\nu})$ represents the summation of the linear and nonlinear damping matrix, $\boldsymbol{g}(\boldsymbol{\eta})$ shows the buoyancy forces and moments, and $\boldsymbol{g}_0$ denotes the static restoring forces and moments. Some physical properties on the system can easily be observed in (1.6). $\boldsymbol{M}$ is known to be the symmetric and positive definite matrix, $\boldsymbol{C}(\boldsymbol{\nu})$ is the skew-symmetric and positive-semidefinite matrix. $\boldsymbol{D}(\boldsymbol{\nu})$ matrix has positive-definite property. Throughout the thesis, (1.6) is adopted as the dynamic equation of the system for controller designs.

The kinematic model does not include any specific rule for USVs. The rate of change in positions and in Euler angles with respect to time are easily obtained by making linear and angular velocity transformations. In a number of researches, it can be shown that a 6 DoF model is to be reduced to a 3 DoF by considering only surge-sway-yaw dynamics in planar motion [28], [19].

Control of USVs motion is generally divided into three levels [10]. The first level is named as strategic (organizational) level. Second is the tactical (task) level which

is composed of advanced controllers. The last stage is known as execution (servo) level. Strategical level includes guidance algorithms which are implemented for path following, path manoeuvring, trajectory tracking and set-point regulation (or point stabilization) [8]. In this study, guidance algorithms are employed for path following purpose. In literature, plenty of research has been published for path following task. Caccia *et al.* [12], have used a combination of (PD) and (PI) guidance methods to produce suitable references for surge speed and heading commands to track a straight line. Control task is implemented in the dual-loop feedback loop on an unmanned catamaran. Naeem's study [38] includes a guidance algorithm which is employed for the simple line-of-sight (LOS) method besides way-point tracking plan for differential drive Springer USV. However, its controller only takes the heading angle as a reference. Oh and Sun [40] have used a LOS guidance scheme to produce yaw angle reference for a 3 DoF USV. Apart from other LOS approaches, reference heading angle of the USV is calculated without using $atan()$ function. This guidance method which is called a linear-like LOS approach is performed in simulations to follow a curved path. Pearson *et al.* [41] have utilized high-level fuzzy logic way-point heading guidance method for a USV. An autonomous underwater vehicle (AUV) is launched and recovered via this guidance law which is implemented on a USV. Proposed guidance law generates the desired speed and heading angle for low-level controllers. USV tries to minimize the cross-track error between the USV and AUV by following the intended path. Pure-pursuit (PP) and line-of-sight (LOS) guidance methods were employed to determine surge speed and yaw angle references for task level controllers.

At the tactical level, controllers take references generated from guidance algorithms and produce torque commands for actuators. Design procedures of controllers cover some essential features. To be able to realize autonomous driving, controllers should handle environmental disturbances, model uncertainties and short-term failures of sensors. Most of the USVs in the market are underactuated since actuation in sway motion is not economically and practically feasible in the general sense. In the literature, a significant number of controllers have been suggested to meet the need caused by the difficulties of handling an underactuated control. Surge speed, yaw angle (yaw rate), collaboration with other unmanned systems, roll stabilization, course keeping, manoeuvring, positioning aspects and features are among the manipulated

variables for USVs [38]. Kumru *et al.* [30] have suggested a linearized and interpolated PID technique in a simulation environment to control surge speed and yaw angle of the USV which is also employed in this study. The proposed PID controller parameters have been optimized in accordance with the thruster characteristic of the system. Breivik *et al.* [10] have studied on straight line target tracking using surge speed and yaw rate controller instead of yaw angle control for USVs. Controller design procedure covers a novel velocity control method which is similar to agility and manoeuvrability concepts of the aircraft vehicle. Pereira *et al.* [42] have designed a weighted controller design for a USV which composes of two propellers and a single rudder to realize station keeping. Heading autopilot includes a PD controller only. While proportional term tries to minimize yaw angle error, derivative term is multiplied by yaw rate. It has been found that putting an integral term improves the performance of error minimization. Oh, and Sun [40] have utilized standard quadratic programming (QP) to solve linearized MPC controller outputs. In their study, all the states are included in the QP problem and limited between specific values. Degree-of-freedom is incremented by one to enhance the path tracking performance. Zheng *et al.* [55] have constructed a nonlinear and a linear model predictive controllers to track a circular path without employing any guidance rule. Effects of the prediction horizon are calculated at different sampling rates. Comparison of linear and nonlinear MPC is demonstrated on multiple figures. Li and Sun [31] have applied a novel disturbance rejection MPC (DC-MPC) method to control ship heading dynamics. The controller is implemented in the simulation environment in the presence of modeled environmental disturbances. Disturbances are constituted from two different of sinusoidal signals. Performance of the DC-MPC is compared for different prediction horizons. In this study, surge speed and yaw angle dynamics of the USV have been controlled via model predictive and cascaded proportional-derivative-integral (PID) control techniques.

This thesis study is a follow-up of some previously realized dissertations, and publications under the supervision of Prof. M. Kemal Leblebicioğlu. First of all, Ahıska [4] has derived a mathematical model for a USV which is composed of only one thruster and one rudder. In his study, PID and LQR controllers are designed to attain desired yaw angle and surge speed. Guidance and obstacle avoidance algorithms are

implemented to drive the USV safely. Experimental validation is performed for PID controllers and experimental, and simulation results are compared with each other. In Erünsal's study [17], successive system identification methods have been experimentally utilized to obtain mathematical model parameters of the USV which is employed in this thesis study as well. PID based piecewise controllers and Sliding Mode Controller (SMC) techniques are executed in a simulation environment to reach desired yaw angle and surge speed. Erünsal *et al.* [18] have also published a paper to obtain the model parameters of USVs. The final values of the model parameters are found by conducting consecutive experiments for the proposed identification scheme. Kumru [29] has investigated navigation and control algorithms in his thesis study as well. A mathematical model whose parameters are identified in Erünsal's thesis is used. Integration of Inertial Navigation System (INS) and Global Navigation Satellite System (GNSS) are provided to design a suitable navigation algorithm. A loosely couple navigation scheme suitable for additional magnetometer measurements is developed to obtain a better navigation solution. LQR and feedback linearization based controllers are utilized and compared with each other. Kumru *et al.* [30] have surveyed some task level control allocation methods for achieving the path following the performance of USVs. The study covers pole placement, feedback linearization, PID and sliding mode controller techniques. Disturbance rejection abilities of all these employed controllers are discussed by Monte Carlo simulations.

1.3 Outline of the Dissertation

This study covers mathematical modeling, guidance and control methods, docking manoeuvres strategies of the USV whose parameters are previously obtained. Introduction chapter is comprised of the motivation of the thesis and literature surveys where selected publications are referenced for the methods in the following chapters.

In Chapter 2, derivation of a 6 DoF mathematical model is explained in detail. The modeled USV has two identical left and right thrusters without any actuators such as rudder. First of all kinematic equations are obtained for position updates. Rigid body inertia, Coriolis and centripetal matrices are found in the rigid body dynamics section. Simplified added mass dynamics are attached to rigid body dynamics. Linear

and nonlinear hydrodynamic damping forces and moments are presented. Buoyancy, gravitational, air drag forces and moments acting on the rigid body are obtained in vector forms. Torques produced by motors are modeled as input. Wave disturbances generated by wind is taken as an environmental disturbance effect. MATLAB simulations of the proposed model are demonstrated at the end of this chapter.

Motion control hierarchy is introduced in Chapter 3. Generation of the reference signals from two types of guidance algorithms which are line-of-sight (LOS) and pure-pursuit (PP) take part in this chapter.

In Chapter 4, first of all, the linearization technique of the nonlinear model and selection of the sampling time are briefly described. The design procedure of MPC and solution of the quadratic programming problem are introduced. Then, the second controller technique, cascaded PID, is employed and tuned by using the particle swarm optimization technique. Parameters of the MPC and cascaded PID controller are optimized on the "S" shape curved path, and their performances are illustrated.

In Chapter 5, the docking problem for a USV is solved in two stages. Optimal control rule is employed for obtaining the path which is to be followed by the vehicle between an initial point and the parking point. Then, a geometric approach is utilized for a backward motion to take the vehicle into the parking slot. Next, way-points are generated from these paths. Finally, combinations of guidance and controller algorithms are compared in simulations.

In Chapter 6, physical and hardware components section includes features of the model boat and the selected hardware equipment. Later, the software architecture of the autopilot card is investigated in detail. At the end of this chapter, experimental validation of the study for MPC and PP combination is expressed, and results are presented in figures.

The last chapter briefly summarizes the whole thesis and gives information about future work. Some suggestions for follow-up studies are listed.

In the appendix part, the details of the yaw angle calculation in Chapter 3 are expressed to overcome the discontinuity problem of the arctangent function at some values.

CHAPTER 2

MATHEMATICAL MODELING OF A SEA SURFACE VEHICLE

2.1 Introduction

A mathematical model for an unmanned surface vehicle, (USV), which has two thrusters is constructed in the scope of this chapter. The model is obtained from the kinematics of a physical system representing a USV and comprises external forces acting on the hull including torque commands for controlling the vehicle and the disturbances due to waves. The body of the boat is taken as rigid, and equations of motion which are used to construct the mathematical model are based on rigid body theory.

A mathematical model can be derived by using different kinds of techniques. In this thesis, similar to that in robot model which is later adopted for marine vehicles by [22], a model with a set of equations in vector form is used. This model includes rigid body dynamics: mass and inertia, centripetal and Coriolis forces, gravitational and buoyancy forces, and damping forces. Furthermore, the vehicle can be exited by some external forces and moments due to various sources: thrusters, air drag due to vehicle speed and wind, and disturbances due to waves and currents. Air drag forces act on the aerial or buoyant part of the vehicle. Control commands are applied to the vehicle via thrusters located at its aft. The longitudinal motion of the vehicle is obtained with the cumulative product of the thrusters, and yaw motion is acquired with uneven thruster commands. In addition to the forces mentioned above, there are disturbance forces due to environmental effects, like wave, wind, sea and ocean currents. They are essential to be used in the stability analysis of the system. There is no rudder on the vehicle in contrary to traditional surface vehicles; therefore rudder

forces and moments are not investigated in this thesis.

This chapter covers the following subjects in detail. First of all, Fossen's robot-like vectorial model which is used to define the kinetic equations of the sea surface vehicle will be explained briefly. After that, kinematics will be expressed for vector transformation from body to inertial frame, and coordinate transformation between these two frames is shortly described. Rigid body dynamics and forces will be discussed. In the end, simulation of the model with a parameter set found by using system identification techniques [17] is run in MATLAB software environment. According to some initial conditions, simulation results obtained from MATLAB will be demonstrated and discussed.

2.2 Vectorial Model

Robot-like vectorial model is adopted as a standard model by the international community due to its easy implementation [22]. The system can be easily examined for stability and motion characteristics because of the properties of inertia, Coriolis and centripetal matrices. These properties are also useful for computational purposes to reduce the number of coefficients to be considered in the design of controllers. The vectorial model (2.1) can be directly written as follows:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} \tag{2.1}$$

Six degrees-of-freedom (DoFs) vectorial representation is used in this equation. Inertia and Coriolis matrices of the system are demonstrated by $\boldsymbol{M}$ and $\boldsymbol{C}$, respectively. For the vector of joint angles, $\boldsymbol{q}$ is used. The term $\boldsymbol{\tau}$ represents moment and forces acting on the vehicle. This simple equation is extended and modified for equations of motion of sea surface vehicles by using [24], [21] and [7]:

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) + \boldsymbol{g}_0 = \boldsymbol{\tau} + \boldsymbol{\tau}_{wind} + \boldsymbol{\tau}_{wave} \tag{2.2}$$

In addition to the above matrices $\boldsymbol{M}$ and $\boldsymbol{C}$, $\boldsymbol{D}$ matrix describes hydrodynamic damping. $\boldsymbol{g}(\boldsymbol{\eta})$ stands for the buoyancy and gravity forces. Static restoring forces and moments are indicated with $\boldsymbol{g}_0$. Disturbance forces due to environmental effects which

are wind and wave forces are $\boldsymbol{\tau}_{wind}$ and $\boldsymbol{\tau}_{wave}$, respectively. States or variables for equations (2.2) are represented in vectorial form as follows:

$$\boldsymbol{\eta} \triangleq [x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi]^T \tag{2.3}$$

$$\boldsymbol{\nu} \triangleq [u \;\; v \;\; w \;\; p \;\; q \;\; r]^T \tag{2.4}$$

Model shown in (2.2) may be converted to the following equation to bring forces right-hand side of the equation:

$$\boldsymbol{M\dot{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_d + \boldsymbol{\tau}_g + \boldsymbol{\tau}_t + \boldsymbol{\tau}_a + \boldsymbol{\tau}_{dis} \tag{2.5}$$

Inertia, centripetal and Coriolis matrices include the added mass dynamics in the left-hand side of (2.5). Right-hand side consists of hydrodynamic damping forces and moments $\boldsymbol{\tau}_d$, buoyancy and gravitational forces and moments $\boldsymbol{\tau}_g$, thruster forces and moments $\boldsymbol{\tau}_t$, air drag forces and moments $\boldsymbol{\tau}_a$, and disturbance forces and moments $\boldsymbol{\tau}_{dis}$. (2.5) will be considered as a reference to derive equations of motion and is inclusively explained in this chapter.

2.3 Kinematics

Kinematics is described as a motion of objects without knowledge of the causes of forces [25]. In navigation problems, linear and angular positions, velocities, accelerations and angular rates of a coordinate frame must be aligned with respect to another frame to update these values in both frames for equations of motion. In kinematics, geometrical aspects of motion are examined to find relations among variables defined in both coordinate frames. A coordinate frame is identified with its origin and axes orientations. Any vector, including positions, velocities and accelerations can be represented in different coordinate frames. Local (reference) and body-fixed coordinate frames, and transformations are briefly explained in the scope of this section.

2.3.1 Coordinate Frames and Notations

Two main navigation frames are utilized to implement guidance and controller methods in the scope of this study. One of them is called the body-fixed coordinate frame,

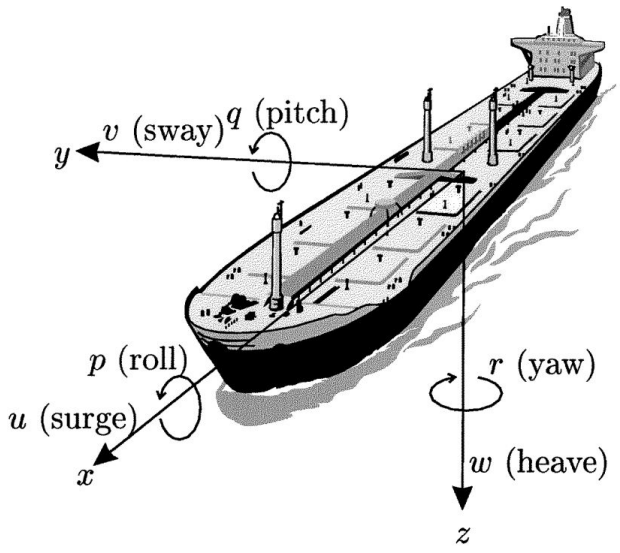

Figure 2.1: Body-fixed frame of a surface vehicle [22].

and the other is North-East-Down (NED) reference frame. Body-fixed coordinate frame is located on the vehicle, as shown in Fig. 2.1. Notations of this figure are based on Society of Naval Architects and Marine Engineers (SNAME) standards [48]. Forces, moments, linear and angular velocities, position and Euler angles are given with their symbols in Table 2.1.

Table 2.1: SNAME notations for surface vehicles, [48].

DoF	Forces and Moments	Linear and Angular Velocities	Position and Euler's Angles
(**1**) motion in x-direction (surge)	X	u	x
(**2**) motion in y-direction (sway)	Y	v	y
(**3**) motion in z-direction (heave)	Z	w	z
(**4**) rotation in x-axis (roll)	K	p	ϕ
(**5**) rotation in y-axis (pitch)	M	q	θ
(**6**) rotation in z-axis (yaw)	N	r	ψ

Body-fixed coordinate frame is represented by $\boldsymbol{b} = \{x_b, y_b, z_b\}$ axes and o_b, which is the origin of it. Angular and linear velocities and forces and torques are expressed

in body frame. Then they are transformed into a reference frame which is defined as NED frame for calculation of equation of motion. NED frame has axes with $\boldsymbol{n} = \{x_n, y_n, z_n\}$ and origin o_n. The North is pointed toward x_n axis, y_n axis indicates East, and in order to represent downwards of the normal to the Earth's surface, z_n is used. This reference frame is also called as a local navigation frame. The position and Euler angles of the vehicle are calculated in the NED frame. Fig. 2.2 illustrates the NED frame with its axes and the origin. State vector includes 12 variables which consist of position, Euler angles, linear and angular velocities in x, y, and z directions. This vector can be represented as follows:

$$\boldsymbol{state} \triangleq [x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi \;\; u \;\; v \;\; w \;\; p \;\; q \;\; r]^T \tag{2.6}$$

Following notations that are listed below are used to derive equations throughout this chapter.

$\boldsymbol{v}_{b|n}^{b} \triangleq [u \;\; v \;\; w]^T$: Linear velocities of o_b with respect to $\{n\}$ expressed in $\{b\}$,

$\boldsymbol{w}_{b|n}^{b} \triangleq [p \;\; q \;\; r]^T$: Angular velocities of o_b with respect to $\{n\}$ expressed in $\{b\}$,

$\boldsymbol{f}_{b}^{b} \triangleq [X \;\; Y \;\; Z]^T$: Forces with acting point o_b in $\{b\}$,

$\boldsymbol{m}_{b}^{b} \triangleq [K \;\; M \;\; N]^T$: Moments about o_b in $\{b\}$,

$\boldsymbol{\Theta_{nb}} \triangleq [\phi \;\; \theta \;\; \psi]^T$: Euler angles between $\{n\}$ and $\{b\}$.

The attitude and position vectors of the reference frame, local navigation (NED)

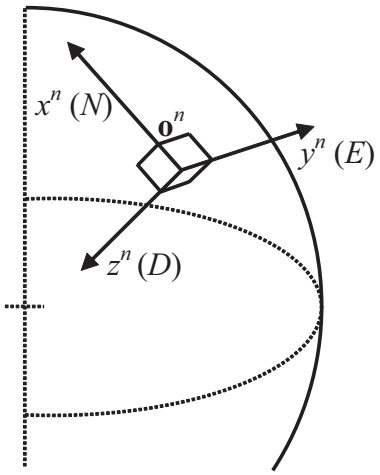

Figure 2.2: Axes and origin of NED (local navigation frame) [25].

frame, is expressed as $\boldsymbol{\eta} = [\boldsymbol{p}_{b|n}^{b}{}^{T} \quad \boldsymbol{\Theta}_{\boldsymbol{nb}}{}_{b}^{bT}]^T$. The body-fixed velocity vector, which is constituted from linear and angular velocities is represented $\boldsymbol{\nu} = [\boldsymbol{v}_{b|n}^{b}{}^{T} \; \boldsymbol{w}_{b|n}^{b}{}^{T}]^T = [\boldsymbol{v}_1{}^{T} \; \boldsymbol{v}_2{}^{T}]^T$. Generalized forces and moments vector is described as $\boldsymbol{\tau} = [\boldsymbol{f}_{b}^{bT} \; \boldsymbol{m}_{b}^{bT}]^T$.

2.3.2 Transformations

Position and orientation of the vehicle are calculated in North-East-Down (NED) frame. Inertial sensors whose measurements are angular and linear velocities, the orientation of the body are located on the rigid body. These values in the body-fixed frame must be converted to NED frame (local navigation frame). In the next part, the transformation of the linear and angular velocities vector from body-fixed to NED frame will be explained.

2.3.2.1 Linear Velocity Transformation

Rotation matrix (also called coordinate transformation matrix) is used to convert the presenter of a vector such as a linear velocity of a USV between two frames. This matrix is denoted as $\boldsymbol{R}_{\alpha}^{\beta}$. By multiplying this matrix with a linear velocity vector in α frame, a related velocity vector is found in β frame. Inversely, related velocity vector in α frame is obtained by multiplying the transformation matrix $\boldsymbol{R}_{\beta}^{\alpha}$ with a linear velocity vector in β frame. The final form of the transformation matrix is a function of Euler angles and is indicated in the following equation.

$$\boldsymbol{R}_{b}^{n}(\boldsymbol{\Theta}_{nb}) = \begin{bmatrix} c(\theta)c(\psi) & -c(\theta)s(\psi) + s(\phi)s(\theta)c(\psi) & s(\phi)s(\psi) + c(\phi)s(\theta)c(\psi) \\ c(\theta)s(\psi) & c(\phi)c(\psi) + s(\phi)s(\theta)s(\psi) & -s(\phi)c(\psi) + c(\phi)s(\theta)s(\psi) \\ -s(\theta) & s(\phi)s(\theta) & c(\phi)c(\theta) \end{bmatrix} \tag{2.7}$$

where $c(\cdot)$ is the cosine operator whereas $s(\cdot)$ is the sine operator. Linear velocity vector defined in the reference frame is obtained from the following equation:

$$\boldsymbol{v}_{b|n}^{n} = \dot{\boldsymbol{p}}_{b|n}^{n} = \boldsymbol{R}_{b}^{n}(\boldsymbol{\Theta}_{nb})\boldsymbol{v}_{b|n}^{b} \tag{2.8}$$

$\dot{\boldsymbol{p}}_{b|n}^{n}$ is time derivative of the position of o_b with respect to {n} expressed in {n}.

2.3.2.2 Angular Velocity Transformation

When angular velocity vector of the body frame, $\boldsymbol{w}_{b|n}^{b}$, is converted to the reference frame, rate of change of the Euler angles with respect to time is obtained in the reference frame. The angular velocity transformation matrix which is the function of Euler's angle is represented as $\boldsymbol{T}_{\Theta}(\boldsymbol{\Theta}_{nb})$:

$$\boldsymbol{T}_{\Theta}(\boldsymbol{\Theta}_{nb}) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)/\cos(\theta) & \cos(\phi)/\cos(\theta) \end{bmatrix} \tag{2.9}$$

The related rate of changes in Euler angles is mathematically indicated by:

$$\dot{\boldsymbol{\Theta}}_{nb} = \boldsymbol{T}_{\Theta}(\boldsymbol{\Theta}_{nb})\boldsymbol{w}_{b|n}^{b} \tag{2.10}$$

2.3.2.3 6 DoFs Kinematic Equations

(2.8) is combined with (2.10) to write a single equation for kinematic calculations. Rate of changes of the position and orientation with respect to time in vector form are described at the following equation:

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}_{\boldsymbol{\Theta}}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{2.11}$$

Explicitly,

$$\begin{bmatrix} \dot{\boldsymbol{p}}_{b|n}^{n} \\ \dot{\boldsymbol{\Theta}}_{nb} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{b}^{n}(\boldsymbol{\Theta}_{nb}) & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{T}_{\Theta}(\boldsymbol{\Theta}_{nb}) \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{b|n}^{b} \\ \boldsymbol{w}_{b|n}^{b} \end{bmatrix} \tag{2.12}$$

where $\boldsymbol{J}_{\boldsymbol{\Theta}}$ is a modified transformation matrix that is used for transforming linear and angular velocity of o_b with respect to the reference frame expressed in the body-fixed frame to rate of change of position and Euler angles.

2.4 Rigid Body Dynamics

Rigid body dynamics of surface vehicles are the essential part of the mathematical modeling. Inertia $\boldsymbol{M}_{RB}$ and Coriolis and centripetal matrices $\boldsymbol{C}_{RB}(\boldsymbol{\nu})$ of the rigid body are found from these dynamics. Lagrangian formalism with using Kirchoff's

equation or Newton-Euler equations is derived from satisfying the following equation which is in the vector form [24]:

$$\boldsymbol{M}_{RB}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau} \tag{2.13}$$

RB is the abbreviation of the rigid body. $\boldsymbol{\nu}$ represents the vector of angular and linear velocities with variables $\boldsymbol{\nu} = [u\ v\ w\ p\ q\ r]^T$. $\boldsymbol{\tau}$ stands for the torque and moment vector, and it is written as $\boldsymbol{\tau} = [X\ Y\ Z\ K\ M\ N]^T$. Origin of the body frame and centre of gravity which are used later are abbreviated as CO and CG, respectively. Newton's second law is written in popular form for surface vehicles according to the following equation:

$$m\dot{\overrightarrow{\boldsymbol{v}_{g|i}}} = \overrightarrow{\boldsymbol{f}_g} \tag{2.14}$$

where $\overrightarrow{\boldsymbol{f}_g}$ is a vector which acts on the centre of gravity CG of the vehicle, $\dot{\overrightarrow{\boldsymbol{v}_{g|i}}}$ stands for acceleration of the CG with respect to the inertial frame, and m is the total amount of mass of the vehicle. According to Newton's first law, if $\overrightarrow{\boldsymbol{f}_g} = 0$, vehicle rests (if its velocity is zero) or keeps its velocity in constant speed. Euler's axioms which were derived from Newton's second law are defined as conservation of linear, $\overrightarrow{\boldsymbol{p}_g}$, and the angular momentum, $\overrightarrow{\boldsymbol{h}_g}$. Euler's first and second axioms are described in (2.15) and (2.16), respectively.

$$\frac{{}^i\partial\overrightarrow{\boldsymbol{p}_g}}{\partial t} = \overrightarrow{\boldsymbol{f}_g} = m\dot{\overrightarrow{\boldsymbol{v}_{g|i}}} \tag{2.15}$$

$$\frac{{}^i\partial\overrightarrow{\boldsymbol{h}_g}}{\partial t} = \overrightarrow{\boldsymbol{m}_g} = \boldsymbol{I}_g\dot{\overrightarrow{\boldsymbol{w}_{g|i}}} \tag{2.16}$$

where $\overrightarrow{\boldsymbol{m}_g}$ is moment vector acting on CG of the body. Linear and angular velocity of the CG with respect to i, inertial frame, are represented as $\overrightarrow{\boldsymbol{v}_{g|i}}$ and $\overrightarrow{\boldsymbol{w}_{g|i}}$, respectively. $\frac{{}^i\partial}{\partial t}$ is a derivative operator that can be expressed time differentiation of a vector in the inertial frame. Inertia dyadic, $\boldsymbol{I}_g$, is described by inertia matrix or tensor. Elements of this matrix are calculated using the equations on following Table 2.2. Positive definite inertia matrix, $\boldsymbol{I}_g$ that satisfies $\boldsymbol{I}_g = \boldsymbol{I}_g{}^T > \mathbf{0}$, is represented with its elements as follows:

$$\boldsymbol{I}_g = \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{yx} & I_y & -I_{yz} \\ -I_{zx} & I_{zy} & I_z \end{bmatrix} \tag{2.17}$$

Table 2.2: Calculation of the elements of the inertia matrix.

Moments of inertia	Products of inertia
$I_x = \int_V (y^2 + z^2)\rho_m dV$, about x_b axis	$I_{xy} = \int_V xy\rho_m dV = \int_V yx\rho_m dV = I_{yx}$
$I_y = \int_V (x^2 + z^2)\rho_m dV$, about y_b axis	$I_{xz} = \int_V xz\rho_m dV = \int_V zx\rho_m dV = I_{zx}$
$I_z = \int_V (x^2 + y^2)\rho_m dV$, about z_b axis	$I_{yz} = \int_V yz\rho_m dV = \int_V zy\rho_m dV = I_{zy}$

In order to derive equations of motion for surface vehicles, two essential assumptions are adopted [22]. First of all, the craft is rigid. It means that craft has no individual elements of mass, so there is no force acting on an individual mass. The second assumption that NED frame $\{n\}$ is inertial. This assumption ignores forces because of Earth's motion relative to the star-fixed inertial reference system. These forces can be negligible when it is compared with hydrodynamic forces. So linear velocity of CG with respect to $\{i\}$ and angular velocity of $\{b\}$ with respect to $\{i\}$ are written again:

$$\overrightarrow{\boldsymbol{v}_{g|i}} \approx \overrightarrow{\boldsymbol{v}_{g|n}} \tag{2.18}$$

$$\overrightarrow{\boldsymbol{w}_{b|i}} \approx \overrightarrow{\boldsymbol{w}_{b|n}} \tag{2.19}$$

Differentiation of a vector, $\overrightarrow{\boldsymbol{a}}$, with respect to time in reference frame has following relation with differentiation of $\overrightarrow{\boldsymbol{a}}$ with respect to time in the body-fixed frame (or moving frame) and angular velocity of CO with respect to $\{i\}$:

$$\frac{{}^{i}\partial \overrightarrow{\boldsymbol{a}}}{\partial t} = \frac{{}^{b}\partial \overrightarrow{\boldsymbol{a}}}{\partial t} + \overrightarrow{\boldsymbol{w}_{b|i}} \times \overrightarrow{\boldsymbol{a}} \tag{2.20}$$

2.4.1 Translational Motion about CG

Position vector of CG with respect to $\{i\}$, $\overrightarrow{\boldsymbol{r}_{g|i}}$ (or it is assumed according to assumption $\overrightarrow{\boldsymbol{r}_{g|n}}$), is expressed in the following equation and relation can be shown in Fig. 2.3:

$$\overrightarrow{\boldsymbol{r}_{g|i}} = \overrightarrow{\boldsymbol{r}_{b|i}} + \overrightarrow{\boldsymbol{r}_{g}} \tag{2.21}$$

or,

$$\overrightarrow{\boldsymbol{r}_{g|n}} = \overrightarrow{\boldsymbol{r}_{b|n}} + \overrightarrow{\boldsymbol{r}_{g}} \tag{2.22}$$

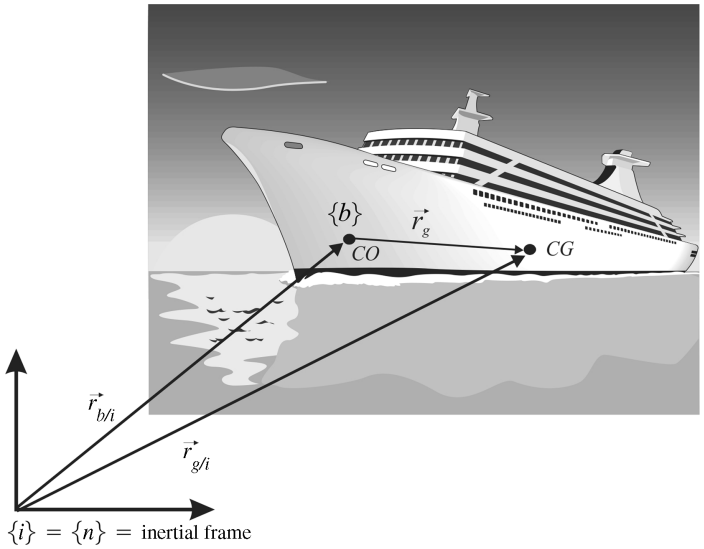

Figure 2.3: Explanation of $\overrightarrow{\boldsymbol{r}_{g|i}}$, $\overrightarrow{\boldsymbol{r}_{b|i}}$, $\overrightarrow{\boldsymbol{r}_g}$ and origin of CO and CG [22].

$\overrightarrow{\boldsymbol{r}_g}$ is the vector from CO to CG , and it is indicated by component form as $\boldsymbol{r}_g = [x_g \;\; y_g \;\; z_g]^T$. By using (2.21), the time differentiation of the $\overrightarrow{\boldsymbol{r}_g}$ is written as:

$$\frac{{}^i\partial \overrightarrow{\boldsymbol{r}_{g|n}}}{\partial t} = \frac{{}^b\partial}{\partial t}(\overrightarrow{\boldsymbol{r}_{b|n}} + \overrightarrow{\boldsymbol{r}_g}) \tag{2.23}$$

Furthermore,

$$\overrightarrow{\boldsymbol{v}_{g|n}} = \overrightarrow{\boldsymbol{v}_{b|n}} + (\frac{{}^i\partial \overrightarrow{\boldsymbol{r}_g}}{\partial t} + \overrightarrow{\boldsymbol{w}_{b|n}} \times \overrightarrow{\boldsymbol{r}_g}) \tag{2.24}$$

Due to the fact that vehicle is a rigid body and CG does not change in time, i.e., $\frac{{}^i\partial \overrightarrow{\boldsymbol{r}_g}}{\partial t} = \mathbf{0}$, so the final form of (2.23) is formulated:

$$\overrightarrow{\boldsymbol{v}_{g|n}} = \overrightarrow{\boldsymbol{v}_{b|n}} + \overrightarrow{\boldsymbol{w}_{b|n}} \times \overrightarrow{\boldsymbol{r}_g} \tag{2.25}$$

The equation of translational motion is derived by the help of (2.15) and (2.25). It follows that

$$\begin{aligned} \overrightarrow{\boldsymbol{f}_g} &= \frac{{}^i\partial}{\partial t} m \overrightarrow{\boldsymbol{v}_{g|n}} \\ &= \frac{{}^b\partial}{\partial t} m \overrightarrow{\boldsymbol{v}_{g|n}} + m \overrightarrow{\boldsymbol{w}_{b|n}} \times \overrightarrow{\boldsymbol{v}_{g|n}} \\ &= m(\dot{\overrightarrow{\boldsymbol{v}_{g|n}}} + \overrightarrow{\boldsymbol{w}_{b|n}} \times \overrightarrow{\boldsymbol{v}_{g|n}}) \end{aligned} \tag{2.26}$$

and

$$\boldsymbol{f}_g^b = m(\dot{\boldsymbol{v}}_{g|n}^b + \boldsymbol{S}(\boldsymbol{w}_{b|n}^b)\boldsymbol{v}_{g|n}^b) \tag{2.27}$$

Skew-symmetric matrix operator, $\boldsymbol{S}(\cdot)$, which is substituted for cross product calculation, is defined with arbitrary vectors, $\boldsymbol{a} = [a_1 \quad a_2 \quad a_3]^T$ and $\boldsymbol{b} = [b_1 \quad b_2 \quad b_3]^T$ in $\mathbb{R}^3$:

$$\boldsymbol{S}(\boldsymbol{a}) = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{2.28}$$

$$\boldsymbol{a} \times \boldsymbol{b} = \boldsymbol{S}(\boldsymbol{a})\boldsymbol{b} \tag{2.29}$$

2.4.2 Rotational Motion about CG

The equation of rotational motion is written using (2.20) and Euler's second axiom in (2.16). Following steps are done to derive rotational motion:

$$\begin{aligned} \overrightarrow{\boldsymbol{m}_g} &= \frac{{}^i\partial}{\partial t}\boldsymbol{I}_g\overrightarrow{\boldsymbol{w}_{b|n}} \\ &= \frac{{}^b\partial}{\partial t}(\boldsymbol{I}_g\overrightarrow{\boldsymbol{w}_{b|n}}) + \overrightarrow{\boldsymbol{w}_{b|n}} \times (\boldsymbol{I}_g\overrightarrow{\boldsymbol{w}_{b|n}}) \\ &= \frac{{}^b\partial}{\partial t}(\boldsymbol{I}_g\overrightarrow{\boldsymbol{w}_{b|n}}) - (\boldsymbol{I}_g\overrightarrow{\boldsymbol{w}_{b|n}}) \times \overrightarrow{\boldsymbol{w}_{b|n}} \end{aligned} \tag{2.30}$$

and

$$\boldsymbol{m}_g^b = \boldsymbol{I}_g\dot{\boldsymbol{w}}_{b|n}^b - \boldsymbol{S}(\boldsymbol{I}_g\boldsymbol{w}_{b|n}^b)\boldsymbol{w}_{b|n}^b \tag{2.31}$$

Let $\boldsymbol{a}$ and $\boldsymbol{b}$ be two arbitrary vectors in $\in \mathbb{R}^3$. Cross-product of two vectors has a relation:

$$\boldsymbol{a} \times \boldsymbol{b} = -\boldsymbol{b} \times \boldsymbol{a} \tag{2.32}$$

In this way, the statement $\overrightarrow{\boldsymbol{w}_{b|n}} \times (\boldsymbol{I}_g\overrightarrow{\boldsymbol{w}_{b|n}})$ can be written as $-(\boldsymbol{I}_g\overrightarrow{\boldsymbol{w}_{b|n}}) \times \overrightarrow{\boldsymbol{w}_{b|n}}$. The equation of motion with respect to the centre of gravity is written in component form according to (2.27) and (2.31):

$$\begin{bmatrix} m\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_g \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}}_{g|n}^b \\ \dot{\boldsymbol{w}}_{b|n}^b \end{bmatrix} + \begin{bmatrix} m\boldsymbol{S}(\boldsymbol{w}_{b|n}^b) & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & -\boldsymbol{S}(\boldsymbol{I}_g\boldsymbol{w}_{b|n}^b) \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{g|n}^b \\ \boldsymbol{w}_{b|n}^b \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_g^b \\ \boldsymbol{m}_g^b \end{bmatrix} \tag{2.33}$$

where the inertia matrix of rigid body is $\boldsymbol{M}_{RB}^{CG} = \begin{bmatrix} m\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_g \end{bmatrix}$, the Coriolis and centripetal matrix is $\boldsymbol{C}_{RB}^{CG} = \begin{bmatrix} m\boldsymbol{S}(\boldsymbol{w}_{b|n}^b) & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & -\boldsymbol{S}(\boldsymbol{I}_g\boldsymbol{w}_{b|n}^b) \end{bmatrix}$ with respect to center of

gravity of the vehicle. The implicit form is indicated by:

$$\boldsymbol{M}_{RB}^{CG}\begin{bmatrix}\dot{\boldsymbol{v}}_{g|n}^{b}\\ \dot{\boldsymbol{w}}_{g|n}^{b}\end{bmatrix}+\boldsymbol{C}_{RB}^{CG}\begin{bmatrix}\boldsymbol{v}_{g|n}^{b}\\ \boldsymbol{w}_{b|n}^{b}\end{bmatrix}=\begin{bmatrix}\boldsymbol{f}_{g}^{b}\\ \boldsymbol{m}_{g}^{b}\end{bmatrix} \tag{2.34}$$

2.4.3 Translational and Rotational Motion about CO

Newton-Euler equations are calculated at the centre of gravity of the vehicle with (2.27) and (2.31). These equations are transformed to rigid-body frame thank to kinematic relations. For this purpose, (2.25) may be expressed as:

$$\begin{aligned}\boldsymbol{v}_{g|n}^{b}&=\boldsymbol{v}_{b|n}^{b}+\boldsymbol{w}_{b|n}^{b}\times\boldsymbol{r}_{g}^{b}\\&=\boldsymbol{v}_{b|n}^{b}-\boldsymbol{r}_{g}^{b}\times\boldsymbol{w}_{b|n}^{b}\\&=\boldsymbol{v}_{b|n}^{b}+\boldsymbol{S}^{T}(\boldsymbol{r}_{g}^{b})\boldsymbol{w}_{b|n}^{b}\end{aligned} \tag{2.35}$$

$$\begin{bmatrix}\boldsymbol{v}_{g|n}^{b}\\ \boldsymbol{w}_{b|n}^{b}\end{bmatrix}=\begin{bmatrix}\boldsymbol{I}_{3\times3}&\boldsymbol{S}^{T}(\boldsymbol{r}_{g}^{b})\\ \boldsymbol{0}_{3\times3}&\boldsymbol{I}_{3\times3}\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_{b|n}^{b}\\ \boldsymbol{w}_{b|n}^{b}\end{bmatrix} \tag{2.36}$$

$\boldsymbol{H}(\boldsymbol{r}_{g}^{b})\in\mathbb{R}^{3}$ is a transformation matrix which converts linear and angular velocity from the CG to CO. It and its transpose are explained by:

$$\boldsymbol{H}(\boldsymbol{r}_{g}^{b})=\begin{bmatrix}\boldsymbol{I}_{3\times3}&\boldsymbol{S}^{T}(\boldsymbol{r}_{g}^{b})\\ \boldsymbol{0}_{3\times3}&\boldsymbol{I}_{3\times3}\end{bmatrix},\quad\boldsymbol{H}^{T}(\boldsymbol{r}_{g}^{b})=\begin{bmatrix}\boldsymbol{I}_{3\times3}&\boldsymbol{0}_{3\times3}\\ \boldsymbol{S}(\boldsymbol{r}_{g}^{b})&\boldsymbol{I}_{3\times3}\end{bmatrix} \tag{2.37}$$

(2.34) is multiplied with $\boldsymbol{H}^{T}(\boldsymbol{r}_{g}^{b})$ from left to preserve positive definiteness and symmetry properties of the matrices in the equation of motion, This gives the following formula:

$$\boldsymbol{H}^{T}(\boldsymbol{r}_{g}^{b})\boldsymbol{M}_{RB}^{CG}\boldsymbol{H}(\boldsymbol{r}_{g}^{b})\begin{bmatrix}\dot{\boldsymbol{v}}_{b|n}^{b}\\ \dot{\boldsymbol{w}}_{b|n}^{b}\end{bmatrix}+\boldsymbol{H}^{T}(\boldsymbol{r}_{g}^{b})\boldsymbol{C}_{RB}^{CG}\boldsymbol{H}(\boldsymbol{r}_{g}^{b})\begin{bmatrix}\boldsymbol{v}_{g|n}^{b}\\ \boldsymbol{w}_{b|n}^{b}\end{bmatrix}=\boldsymbol{H}^{T}(\boldsymbol{r}_{g}^{b})\begin{bmatrix}\boldsymbol{f}_{g}^{b}\\ \boldsymbol{m}_{g}^{b}\end{bmatrix} \tag{2.38}$$

Rigid body mass, $\boldsymbol{M}_{RB}^{CO}$, and Coriolis and centripetal, $\boldsymbol{C}_{RB}^{CO}$, matrices with respect to body-fixed frame become:

$$\boldsymbol{M}_{RB}^{CO}\triangleq\boldsymbol{H}^{T}(\boldsymbol{r}_{g}^{b})\boldsymbol{M}_{RB}^{CG}\boldsymbol{H}(\boldsymbol{r}_{g}^{b}),\qquad\boldsymbol{M}_{RB}^{CO}=\boldsymbol{M}^{T}{}_{RB}^{CO}>\boldsymbol{0} \tag{2.39}$$

$$\boldsymbol{C}_{RB}^{CO}\triangleq\boldsymbol{H}^{T}(\boldsymbol{r}_{g}^{b})\boldsymbol{C}_{RB}^{CG}\boldsymbol{H}(\boldsymbol{r}_{g}^{b}),\qquad\boldsymbol{C}_{RB}^{CO}=-\boldsymbol{C}^{T}{}_{RB}^{CO}\geq\boldsymbol{0} \tag{2.40}$$

Explicitly,

$$\boldsymbol{M}_{RB}^{CO}=\begin{bmatrix}m\boldsymbol{I}_{3\times3}&-m\boldsymbol{S}(\boldsymbol{r}_{g}^{b})\\ m\boldsymbol{S}(\boldsymbol{r}_{g}^{b})&\boldsymbol{I}_{g}-m\boldsymbol{S}^{2}(\boldsymbol{r}_{g}^{b})\end{bmatrix} \tag{2.41}$$

$$\boldsymbol{C}_{RB}^{CO} = \begin{bmatrix} m\boldsymbol{S}(\boldsymbol{w}_{b|n}^{b}) & -m\boldsymbol{S}(\boldsymbol{w}_{b|n}^{b})\boldsymbol{S}(\boldsymbol{r}_{g}^{b}) \\ m\boldsymbol{S}(\boldsymbol{r}_{g}^{b})\boldsymbol{S}(\boldsymbol{w}_{b|n}^{b}) & -\boldsymbol{S}\Big(\big(\boldsymbol{I}_{g} - m\boldsymbol{S}^{2}(\boldsymbol{r}_{g}^{b})\boldsymbol{w}_{b|n}^{b}\big)\Big) \end{bmatrix} \tag{2.42}$$

while $\boldsymbol{M}_{RB}^{CO}$ is a unique matrix, $\boldsymbol{C}_{RB}^{CO}$ can be represented in different ways. It may be found by utilizing the Lagrangian approach with Kirchoff's equations. $\boldsymbol{v}_1 = \boldsymbol{v}_{b|n}^{b}$, $\boldsymbol{v}_2 = \boldsymbol{w}_{b|n}^{b}$, and $\boldsymbol{\nu} = [\boldsymbol{v}_1{}^T \quad \boldsymbol{v}_2{}^T]^T$ were defined earlier. The kinetic energy of the vehicle that is defined in quadratic form is described as [44]:

$$T = \frac{1}{2}\boldsymbol{\nu}^T\boldsymbol{M}\boldsymbol{\nu} \tag{2.43}$$

$\boldsymbol{M}_{RB}^{CO}$ can be written with its entries as follows:

$$\boldsymbol{M}_{RB}^{CO} = \begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{M}_{12} \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} \end{bmatrix} \tag{2.44}$$

where $\boldsymbol{M}_{11} = m\boldsymbol{I}_{3x3}$, $\boldsymbol{M}_{12} = -m\boldsymbol{S}(\boldsymbol{r}_{g}^{b})$, $\boldsymbol{M}_{21} = -m\boldsymbol{S}(\boldsymbol{r}_{g}^{b})$ and $\boldsymbol{M}_{22} = \boldsymbol{I}_{g}$. Now, (2.43) may be explicitly expressed as:

$$T = \frac{1}{2}\Big(\boldsymbol{v}_1^T\boldsymbol{M}_{11}\boldsymbol{v}_1 + \boldsymbol{v}_2^T\boldsymbol{M}_{12}\boldsymbol{v}_2 + \boldsymbol{v}_1^T\boldsymbol{M}_{21}\boldsymbol{v}_1 + \boldsymbol{v}_2^T\boldsymbol{M}_{22}\boldsymbol{v}_2\Big) \tag{2.45}$$

Kirchoff's equations are formulated by:

$$\frac{\partial}{\partial t}\Big(\frac{\partial T}{\partial \boldsymbol{v}_1}\Big) + \boldsymbol{v}_2 \times \frac{\partial T}{\partial \boldsymbol{v}_1} = \boldsymbol{\tau}_1 \tag{2.46}$$

$$\frac{\partial}{\partial t}\Big(\frac{\partial T}{\partial \boldsymbol{v}_2}\Big) + \boldsymbol{v}_2 \times \frac{\partial T}{\partial \boldsymbol{v}_2} + \boldsymbol{v}_1 \times \frac{\partial T}{\partial \boldsymbol{v}_1} = \boldsymbol{\tau}_2 \tag{2.47}$$

Time differentiations of the kinetic energy with respect to $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$:

$$\frac{\partial T}{\partial \boldsymbol{v}_1} = \boldsymbol{M}_{11}\boldsymbol{v}_1 + \boldsymbol{M}_{12}\boldsymbol{v}_2 \tag{2.48}$$

$$\frac{\partial T}{\partial \boldsymbol{v}_2} = \boldsymbol{M}_{21}\boldsymbol{v}_1 + \boldsymbol{M}_{22}\boldsymbol{v}_2 \tag{2.49}$$

The equation of motion is written according to (2.46) and (2.47) :

$$\begin{bmatrix} \boldsymbol{M}_{11} & \boldsymbol{M}_{12} \\ \boldsymbol{M}_{21} & \boldsymbol{M}_{22} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}}_1 \\ \dot{\boldsymbol{v}}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{3\times3} & -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{v}_1 + \boldsymbol{M}_{12}\boldsymbol{v}_2) \\ -\boldsymbol{S}(\boldsymbol{M}_{11}\boldsymbol{v}_1 + \boldsymbol{M}_{12}\boldsymbol{v}_2) & -\boldsymbol{S}(\boldsymbol{M}_{21}\boldsymbol{v}_1 + \boldsymbol{M}_{22}\boldsymbol{v}_2) \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \end{bmatrix} \tag{2.50}$$

So, alternative representation of $\boldsymbol{C}_{RB}^{CO}$ is obtained as:

$$\boldsymbol{C}_{RB}^{CO} = \begin{bmatrix} \boldsymbol{0}_{3\times3} & -m\boldsymbol{S}(\boldsymbol{v}_1) - m\boldsymbol{S}(\boldsymbol{v}_2)\boldsymbol{S}(\boldsymbol{r}_{g}^{b}) \\ -m\boldsymbol{S}(\boldsymbol{v}_1) + m\boldsymbol{S}(\boldsymbol{r}_{g}^{b})\boldsymbol{S}(\boldsymbol{v}_2) & -\boldsymbol{S}(\boldsymbol{I}_{b}\boldsymbol{v}_2) \end{bmatrix} \tag{2.51}$$

where $\boldsymbol{I}_b = \boldsymbol{I}_g - m\boldsymbol{S}^2(\boldsymbol{r}_g^b)$.

$$\boldsymbol{I}_b = \begin{bmatrix} I_x + m(y_g^2 + z_g^2) & -I_{xy} - mx_g y_g & -I_{xz} - mx_g z_g \\ -I_{yx} - mx_g y_g & I_y + m(x_g^2 + z_g^2) & -I_{yz} - my_g z_g \\ -I_{zx} - mx_g z_g & I_{zy} - my_g z_g & I_z + m(x_g^2 + y_g^2) \end{bmatrix} = \begin{bmatrix} I_x^b & I_{xy}^b & I_{xz}^b \\ I_{yx}^b & I_y^b & I_{yz}^b \\ I_{zx}^b & I_{zy}^b & I_z^b \end{bmatrix} \tag{2.52}$$

$\boldsymbol{M}_{RB}^{CO}$ is explicitly written with its indices form in the following matrix:

$$\boldsymbol{M}_{RB}^{CO} = \begin{bmatrix} m & 0 & 0 & 0 & mz_g & -my_g \\ 0 & m & 0 & -mz_g & 0 & mx_g \\ 0 & 0 & m & my_g & -mx_g & 0 \\ 0 & -mz_g & my_g & I_x^b & I_{xy}^b & I_{xz}^b \\ mz_g & 0 & -mx_g & I_{yx}^b & I_y^b & I_{yz}^b \\ -my_g & mx_g & 0 & I_{zx}^b & I_{zy}^b & I_z^b \end{bmatrix} \tag{2.53}$$

Following matrix indicates indices of the Coriolis and centripetal matrix, $\boldsymbol{C}_{RB}^{CO}$.

$$\boldsymbol{C}_{RB}^{CO} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -m(y_g q + z_g r) & m(y_g p + w) & m(z_g p - v) \\ m(x_g q - w) & -m(z_g r + x_g p) & m(z_g q + u) \\ m(x_g r + v) & m(y_g r - u) & m(x_g p + y_g q) \end{bmatrix}$$
$$\begin{bmatrix} m(y_g q + z_g r) & -m(x_g q - w) & -m(x_g r + v) \\ -m(y_g p + w) & m(z_g r + x_g p) & -m(y_g r - u) \\ -m(z_g p - v) & -m(z_g q + u) & m(x_g p + y_g q) \\ 0 & -I_{yz}^b q - I_{xz}^b p + I_z^b r & I_{yz}^b r + I_{xy}^b p - I_y^b q \\ I_{yz}^b q + I_{xz}^b p - I_z^b r & 0 & -I_{xz}^b r - I_{xy}^b q + I_x^b p \\ -I_{yz}^b r - I_{xy}^b p + I_y^b q & I_{xz}^b r + I_{xy}^b q - I_x^b p & 0 \end{bmatrix} \tag{2.54}$$

When CO corresponds exactly CG, $\boldsymbol{r}_g^b = [0 \quad 0 \quad 0]^T$ is satisfied. Thus, $\boldsymbol{M}_{RB}^{CO}$ and $\boldsymbol{C}_{RB}^{CO}$ matrices become more simple. Entries whose values are zero of $\boldsymbol{M}_{RB}^{CO}$ matrix

increase. $\boldsymbol{M}_{RB}^{CO}$ turns out to be a diagonal matrix like:

$$\boldsymbol{M}_{RB}^{CO} = \begin{bmatrix} m\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & diag(I_x, I_y, I_z) \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_x & 0 & 0 \\ 0 & 0 & 0 & 0 & I_y & 0 \\ 0 & 0 & 0 & 0 & 0 & I_z \end{bmatrix} \tag{2.55}$$

$\boldsymbol{C}_{RB}^{CO}$ matrix becomes:

$$\boldsymbol{C}_{RB}^{CO} = \begin{bmatrix} 0 & 0 & 0 & 0 & mw & -mv \\ 0 & 0 & 0 & -mw & 0 & mu \\ 0 & 0 & 0 & mv & -mu & 0 \\ 0 & mw & -mv & 0 & I_z r & -I_y q \\ -mw & 0 & mu & -I_z r & 0 & I_x p \\ mv & -mu & 0 & I_y q & -I_x p & 0 \end{bmatrix} \tag{2.56}$$

2.5 Added Mass Dynamics

Surface vehicles move along partially submerged in fluid. When a rigid-body goes around the water surface, it has to move out fluid of its headway. This phenomenon is indicated as 'added mass', sometimes is called 'virtual mass', or 'ascension to mass' [11]. In order to find appropriate equations of motion, dynamics of the ambient fluid must be added to (2.38) or (2.50). These dynamics include inertia, centripetal and Coriolis matrices which are derived from the kinetic energy approach of Kirchoff [27]. The vehicle needs kinetic energy produced by thrusters to move. This energy must be greater than the kinetic energy of the surrounding fluid. Otherwise, vehicle's motion is steady. Dynamic equations of the added mass are completely expressed with state variables and partial differentiation of forces and moments with respect to acceleration [4], [11], [27]:

$$\begin{aligned} X_A = {} & X_{\dot{u}}\dot{u} + X_{\dot{v}}\dot{v} + X_{\dot{w}}\dot{w} + X_{\dot{p}}\dot{p} + X_{\dot{q}}\dot{q} + X_{\dot{r}}\dot{r} \\ & + X_{\dot{w}}uq + Y_{\dot{w}}vq + Z_{\dot{w}}wq + Z_{\dot{p}}pq + Z_{\dot{q}}qq + Z_{\dot{r}}rq \\ & - X_{\dot{v}}ur - Y_{\dot{v}}vr - Y_{\dot{w}}wr - Y_{\dot{p}}pr - Y_{\dot{q}}qr - Y_{\dot{r}}rr \end{aligned} \tag{2.57}$$

$$
\begin{aligned}
Y_A &= Y_{\dot{u}}\dot{u} + Y_{\dot{v}}\dot{v} + Y_{\dot{w}}\dot{w} + Y_{\dot{p}}\dot{p} + Y_{\dot{q}}\dot{q} + Y_{\dot{r}}\dot{r} \\
&\quad - X_{\dot{w}}up - Y_{\dot{w}}vp - Z_{\dot{w}}wp - Z_{\dot{p}}pp - Z_{\dot{q}}qp - Z_{\dot{r}}rp \\
&\quad + X_{\dot{u}}ur + X_{\dot{v}}vr + X_{\dot{w}}wr + X_{\dot{p}}pr + X_{\dot{q}}qr + X_{\dot{r}}rr \qquad (2.58) \\
Z_A &= Z_{\dot{u}}\dot{u} + Z_{\dot{v}}\dot{v} + Z_{\dot{w}}\dot{w} + Z_{\dot{p}}\dot{p} + Z_{\dot{q}}\dot{q} + Z_{\dot{r}}\dot{r} \\
&\quad + X_{\dot{v}}up + Y_{\dot{v}}vp - Y_{\dot{w}}wp + Y_{\dot{p}}pp + Y_{\dot{q}}qp + Y_{\dot{r}}rp \\
&\quad - X_{\dot{u}}uq - X_{\dot{v}}vq - X_{\dot{w}}wq - X_{\dot{p}}pq - X_{\dot{q}}qq - X_{\dot{r}}rq \qquad (2.59) \\
K_A &= K_{\dot{u}}\dot{u} + K_{\dot{v}}\dot{v} + K_{\dot{w}}\dot{w} + K_{\dot{p}}\dot{p} + K_{\dot{q}}\dot{q} + K_{\dot{r}}\dot{r} \\
&\quad + X_{\dot{w}}uv + Y_{\dot{w}}vv + Z_{\dot{w}}wv + Z_{\dot{p}}pv + Z_{\dot{q}}qv + Z_{\dot{r}}rv \\
&\quad - X_{\dot{v}}uw - Y_{\dot{v}}vq - Y_{\dot{w}}ww - Y_{\dot{p}}pw - Y_{\dot{q}}qw - Y_{\dot{r}}rw \\
&\quad + X_{\dot{r}}uq + Y_{\dot{r}}vq + Z_{\dot{r}}wq + K_{\dot{r}}pq + M_{\dot{r}}qq + N_{\dot{r}}rq \\
&\quad - X_{\dot{q}}ur - Y_{\dot{q}}vr - Z_{\dot{q}}wr - K_{\dot{q}}pr - M_{\dot{q}}qr - M_{\dot{r}}rr \qquad (2.60) \\
M_A &= M_{\dot{u}}\dot{u} + M_{\dot{v}}\dot{v} + M_{\dot{w}}\dot{w} + M_{\dot{p}}\dot{p} + M_{\dot{q}}\dot{q} + M_{\dot{r}}\dot{r} \\
&\quad - X_{\dot{w}}uu - Y_{\dot{w}}vu - Z_{\dot{w}}wu - Z_{\dot{p}}pu - Z_{\dot{q}}qu - Z_{\dot{r}}ru \\
&\quad + X_{\dot{u}}uw + x_{\dot{v}}vw + X_{\dot{w}}ww + X_{\dot{p}}pw + X_{\dot{q}}qw + X_{\dot{r}}rw \\
&\quad - X_{\dot{r}}up - Y_{\dot{r}}vp - Z_{\dot{r}}wp - K_{\dot{r}}pp - M_{\dot{r}}qp - N_{\dot{r}}rp \\
&\quad + X_{\dot{p}}ur + Y_{\dot{p}}vr + Z_{\dot{p}}wr + K_{\dot{p}}pr + K_{\dot{q}}qr + K_{\dot{r}}rr \qquad (2.61) \\
N_A &= N_{\dot{u}}\dot{u} + N_{\dot{v}}\dot{v} + N_{\dot{w}}\dot{w} + N_{\dot{p}}\dot{p} + N_{\dot{q}}\dot{q} + N_{\dot{r}}\dot{r} \\
&\quad + X_{\dot{v}}uu + Y_{\dot{v}}vu + Y_{\dot{w}}wu + Y_{\dot{p}}pu + Y_{\dot{q}}qu + Y_{\dot{r}}ru \\
&\quad - X_{\dot{u}}uv - X_{\dot{v}}vv - X_{\dot{w}}wv - X_{\dot{p}}pv + X_{\dot{q}}qv - X_{\dot{r}}rv \\
&\quad + X_{\dot{q}}up + Y_{\dot{q}}vp + Z_{\dot{q}}wp + K_{\dot{q}}pp + M_{\dot{q}}qp + N_{\dot{r}}rp \\
&\quad - X_{\dot{p}}ur - Y_{\dot{p}}vr - Z_{\dot{p}}wr - K_{\dot{p}}pr - K_{\dot{q}}qr - K_{\dot{r}}rr \qquad (2.62)
\end{aligned}
$$

X_A, Y_A, Z_A are the added forces and K_A, M_A, N_A stand for the moments which come from added mass dynamics. $X_{\dot{u}} = \dfrac{\partial X}{\partial \dot{u}}$ is the abbreviation of the partial derivative of force in surge direction with respect to acceleration in surge direction. Other partial derivative terms can be interpreted similarly. Some terms from (2.57) to (2.62) may be simply written as follows [4]:

$$
\begin{aligned}
a1 &= X_{\dot{u}}u + X_{\dot{v}}v + X_{\dot{w}}w + X_{\dot{p}}p + X_{\dot{q}}q + X_{\dot{r}}r \\
a2 &= X_{\dot{v}}u + Y_{\dot{v}}v + Y_{\dot{w}}w + Y_{\dot{p}}p + Y_{\dot{q}}q + Y_{\dot{r}}r
\end{aligned}
$$

$$
\begin{aligned}
a3 &= X_{\dot{w}}u + Y_{\dot{w}}v + Z_{\dot{w}}w + Z_{\dot{p}}p + Z_{\dot{q}}q + Z_{\dot{r}}r \\
a4 &= X_{\dot{p}}u + Y_{\dot{p}}v + Z_{\dot{p}}w + K_{\dot{p}}p + K_{\dot{q}}q + K_{\dot{r}}r \\
a5 &= X_{\dot{q}}u + Y_{\dot{q}}v + Y_{\dot{q}}w + K_{\dot{q}}p + M_{\dot{q}}q + M_{\dot{r}}r \\
a6 &= X_{\dot{r}}u + Y_{\dot{r}}v + Z_{\dot{r}}w + K_{\dot{r}}p + M_{\dot{r}}q + N_{\dot{r}}r
\end{aligned} \tag{2.63}
$$

Using from (2.57) to (2.63), the simple equation is written as follows:

$$
\underbrace{\begin{bmatrix}
X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} \\
Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} \\
Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} \\
K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} \\
M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} \\
N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}}
\end{bmatrix}}_{-\boldsymbol{M}_A}
\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} +
\underbrace{\begin{bmatrix}
0 & 0 & 0 & 0 & a_3 & -a_2 \\
0 & 0 & 0 & -a_3 & 0 & a_1 \\
0 & 0 & 0 & -a_2 & a_1 & 0 \\
0 & a_3 & -a_2 & 0 & a_6 & -a_5 \\
-a_3 & 0 & a_1 & -a_6 & 0 & a_4 \\
a_2 & -a_1 & 0 & a_5 & -a_4 & 0
\end{bmatrix}}_{-\boldsymbol{C}_A(\nu)}
\begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix} =
\begin{bmatrix} X_A \\ Y_A \\ Z_A \\ K_A \\ M_A \\ N_A \end{bmatrix}
\tag{2.64}
$$

(2.64) is added the left-hand side of to (2.38) or (2.50) by multiplying it -1. Added mass inertia matrix, $\boldsymbol{M}_A$, is positive definite, i.e. $\boldsymbol{M}_A > \boldsymbol{0}$. Added Coriolis and centripetal matrix, $\boldsymbol{C}_A(\boldsymbol{\nu})$, is a positive semi-definite matrix which satisfies $\boldsymbol{C}_A(\boldsymbol{\nu}) = -\boldsymbol{C}_A(\boldsymbol{\nu})^T \geq \boldsymbol{0}$. It makes that $\boldsymbol{C}_A(\boldsymbol{\nu})$ is skew-symmetric and it conserves the feature of overall Coriolis and the centripetal matrix is skew-symmetric. For surface vehicles, the heave, roll, pitch modes are disregarded due to the fact that these variables are small according to surge speed. $\boldsymbol{M}_A$ and $\boldsymbol{C}_A(\boldsymbol{\nu})$ are simplified when the magnitude

of the surge speed, $|u|$, becomes considerably larger than zero [21]:

$$\boldsymbol{M}_A = -\begin{bmatrix} X_{\dot{u}} & 0 & 0 & 0 & 0 & 0 \\ 0 & Y_{\dot{v}} & 0 & 0 & 0 & Y_{\dot{r}} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & N_{\dot{v}} & 0 & 0 & 0 & N_{\dot{r}} \end{bmatrix} \tag{2.65}$$

and

$$\boldsymbol{C}_A(\boldsymbol{\nu}) = -\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -Y_{\dot{v}}v - \dfrac{Y_{\dot{r}} + N_{\dot{v}}}{2}r \\ 0 & 0 & 0 & 0 & 0 & X_{\dot{u}}u \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ Y_{\dot{v}}v + \dfrac{Y_{\dot{r}} + N_{\dot{v}}}{2}r & -X_{\dot{u}}u & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.66}$$

When forward speed becomes zero ($u \approx 0$), $N_{\dot{v}}$ can be replaced with $Y_{\dot{r}}$ and the (2.64) can be rewritten as follows:

$$\begin{bmatrix} X_{\dot{u}} & 0 & 0 & 0 & 0 & 0 \\ 0 & Y_{\dot{v}} & 0 & 0 & 0 & Y_{\dot{r}} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & N_{\dot{v}} & 0 & 0 & 0 & N_{\dot{r}} \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} +$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -Y_{\dot{v}}v - Y_{\dot{r}}r \\ 0 & 0 & 0 & 0 & 0 & X_{\dot{u}}u \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ Y_{\dot{v}}v + Y_{\dot{r}}r & -X_{\dot{u}}u & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix} = \begin{bmatrix} X_A \\ Y_A \\ 0 \\ 0 \\ 0 \\ N_A \end{bmatrix} \tag{2.67}$$

Since $\boldsymbol{M}_A = \boldsymbol{M}_A^T$ and $\boldsymbol{M}_A > \boldsymbol{0}$. This provides that inertia matrix is symmetric. When it is added to the body-fixed inertia matrix, symmetric property of the overall

system is preserved. As a conclusion overall inertia, $\boldsymbol{M}$, and Coriolis and centripetal, $\boldsymbol{C}(\boldsymbol{\nu})$, matrices of the surface vehicle is defined by the following equations:

$$\boldsymbol{M} = \boldsymbol{M}_{RB}^{CO} + \boldsymbol{M}_A \tag{2.68}$$

$$\boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C}_{RB}^{CO}(\boldsymbol{\nu}) + \boldsymbol{C}_A(\boldsymbol{\nu}) \tag{2.69}$$

2.6 Hydrodynamic Damping Forces

The damping matrix is situated on the left-hand side of the (2.5). To put damping force in matrix form, it is generally written as a summation of the four kinds of damping components [21]:

$$\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_P(\boldsymbol{\nu}) + \boldsymbol{D}_S(\boldsymbol{\nu}) + \boldsymbol{D}_W(\boldsymbol{\nu}) + \boldsymbol{D}_M(\boldsymbol{\nu}) \tag{2.70}$$

Hydrodynamic damping matrix, $\boldsymbol{D}(\boldsymbol{\nu})$, is a real, strictly positive and non-symmetric matrix that is convenient to be written as, $\boldsymbol{D}(\boldsymbol{\nu}) > \boldsymbol{0}$ for all $\boldsymbol{\nu} \in \mathbb{R}^6$. Radiation-induced potential damping is represented by $\boldsymbol{D}_P(\boldsymbol{\nu})$. Linear skin and quadratic skin friction are called $\boldsymbol{D}_S(\boldsymbol{\nu})$. In the presence of wave in the environment, wave drift damping $\boldsymbol{D}_W(\boldsymbol{\nu})$ occurs. $\boldsymbol{D}_M(\boldsymbol{\nu})$ stands for the damping due to vortex shedding. Damping forces and moments are found by multiplying damping matrix $\boldsymbol{D}(\boldsymbol{\nu})$ with velocity vector, $\boldsymbol{\nu}$:

$$\boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_d. \tag{2.71}$$

Radiation-induced potential damping: It is also called linear frequency-dependent potential damping. Potential damping occurs in a body that oscillates with the wave excitation frequency in the absence of incident waves. At low frequencies, potential damping terms can be negligible against viscous damping.

Friction: When surface vehicles have motion in low frequency, linear skin friction due to the laminar boundary layer is regarded as a friction damping factor. At the high frequencies, quadratic skin friction is taken into account because of the turbulent boundary layer.

Wave Drift Damping: This kind of damping that is derived from second-order wave theory in the presence of the wave. Wavelength, wavelength encounter angle, vehicle's forward speed (for higher sea state) and wave encounter frequency are a func-

tion of the wave drifting damping [11]. This damping appears like added resistance against the motion of the vehicle.

Damping due to Vortex Shedding: While the vehicle moves in a viscous fluid, frictional forces occur due to the fact that the system is not conservative with respect to energy. This phenomenon is also called 'interference drag' and happens at the sharp edges of the vehicle because of the shedding of vortex sheets. Force equation of the viscous damping due to the vortex shedding can be written as follows:

$$f(U) = \frac{1}{2}\rho C_D(R_n) A|u|u \tag{2.72}$$

Force is a function of the forward speed of the vehicle, u, and $|u|$ is the absolute value of the forward speed. The submerged cross-sectional area of the vehicle is represented by A. Water density is indicated by ρ. $C_D(R_n)$ is the drag coefficient function of Reynold's number which is formulated as:

$$R_n = \frac{UD}{k} \tag{2.73}$$

where D stands for the characteristic length of the body, and k is used to describe the kinematic viscosity coefficient. By considering abovementioned damping elements, the damping matrix is separated into two matrices which are linear damping matrix and nonlinear damping matrix:

$$\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_l(\boldsymbol{\nu}) + \boldsymbol{D}_n(\boldsymbol{\nu}) \tag{2.74}$$

Linear and nonlinear damping matrices are represented by $\boldsymbol{D}_l(\boldsymbol{\nu})$ and $\boldsymbol{D}_n(\boldsymbol{\nu})$, respectively. The terms which come from linear and nonlinear damping components may not be physically separated from each other. However, linear damping matrix for surface vehicles is expressed by 'Damping Model for Dynamic Positioning of Ships' whereas nonlinear damping matrix is explained via 'Nonlinear Damping Model for High-Speed Maneuvers' in [19]. By neglecting heave, roll, pitch modes, the surge mode may be decoupled from steering modes (sway and yaw) with $x - z$ symmetry for low-speed ships. Linearized damping matrix in (2.74) can be written in compo-

nent form as follows:

$$\boldsymbol{D}_l = -\begin{bmatrix} X_u & 0 & 0 & 0 & 0 & 0 \\ 0 & Y_v & 0 & 0 & 0 & Y_r \\ 0 & 0 & Z_w & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & N_v & 0 & 0 & 0 & N_r \end{bmatrix} \tag{2.75}$$

At low speeds, $N_v = Y_r$ is taken to obtain symmetric linear damping matrix, i.e., $\boldsymbol{D}_l = \boldsymbol{D}_l^T$. Assuming that surge is decoupled, nonlinear damping matrix can be written by adding $Z_{|w|w}|w|$ term according to Norrbin's nonlinear model [39]:

$$\boldsymbol{D}_n(\boldsymbol{\nu}) = -\begin{bmatrix} X_{|u|u}|u| & 0 & 0 & 0 & 0 & 0 \\ 0 & Y_{|v|v}|v| + Y_{|r|v}|v| & 0 & 0 & 0 & Y_{|v|r}|v| + Y_{|r|r}|r| \\ 0 & 0 & Z_{|w|w}|w| & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & N_{|v|v}|v| + N_{|r|v}|r| & 0 & 0 & 0 & N_{|v|r}|v| + N_{|r|r}|r| \end{bmatrix} \tag{2.76}$$

where,

$$X_{|u|u} = \frac{1}{2}\rho A_x C_x \tag{2.77}$$

$C_x > 0$ is the current coefficient which is determined from experiments when the vehicle is in up to 1.0 m/s currents and A_x is the frontal project area. Other terms can be easily found by arranging (2.77). Matrix $\boldsymbol{D}_n(\boldsymbol{\nu})$ can be simplified by neglecting $|r|r$ and $|r|v$ components which are considered small against surge speed, u. So nonlinear damping matrix becomes:

$$\boldsymbol{D}_n = -\begin{bmatrix} X_{|u|u}|u| & 0 & 0 & 0 & 0 & 0 \\ 0 & Y_{|v|v}|v| & 0 & 0 & 0 & Y_{|v|r}|v| \\ 0 & 0 & Z_{|w|w}|w| & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & N_{|v|v}|v| & 0 & 0 & 0 & N_{|v|r}|v| \end{bmatrix} \tag{2.78}$$

Derivation of the damping matrix is done under some assumptions. The body of the vehicle is assumed symmetrical, and it has a box shape. Reynold number is accepted

to be higher than 10^4, i.e., $R_n > 10^4$. Components $Z_{|w|w}|w|$ and Z_w which are obtained from z-direction are taken into account for damping force calculations.

2.7 Restoring Forces

Combination of the buoyancy and gravitational forces is simply called restoring forces. For surface vehicles, the bouncy force acts through the centre of the submerged part, CB, and gravitational force acts through the centre of gravity, CG. Although forces act on these points, they must be transformed to the body-fixed frame by knowing $\boldsymbol{r}_g^b = [x_g \quad y_g \quad z_g]^T$ to calculate equations of motion correctly. Restoring forces, $\boldsymbol{\tau}_g$, appear in right hand-side of (2.5).

A hypothetical point, M_T, is defined to find out static stability. It is called meta-centre that is obtained from an intersection of a vertical line through the CB and the new CB when the vehicle is moved, or tilted in the fluid. Restoring forces are obtained from meta-centric heights, the position of CG and CB, and water planer area of the craft. Meta-centric heights include two quantities that one of them is transverse meta-centric height (in meter), $\overline{GM}_T$, and other is longitudinal meta-centric height (in meter), $\overline{GM}_L$. Meta-center, CB, and new CB are shown in Fig. 2.4. The phenomenon of a craft floating on a fluid is explained by Archimedes equation that is based on

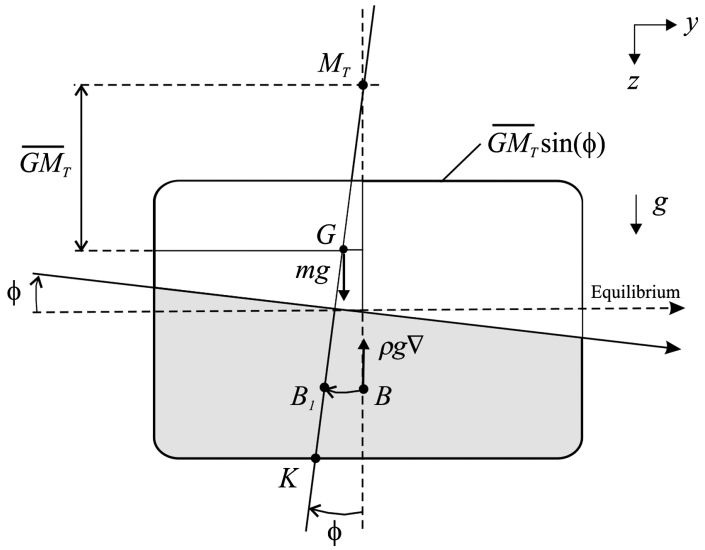

Figure 2.4: Representation of transverse meta-centric stability [22].

a balance between the weight of vehicle and weight of water displacement of the submerged part. This relation is simply formulated as follows:

$$mg = \rho g \nabla \tag{2.79}$$

where ∇ is the displaced water volume. $\rho_w = \rho_w(T^o)$ is the water density which depends on the temperature and salt ratio of the water. When a body is not in motion in z-direction, i.e. $z = 0$, (2.79) is satisfied. Any change in z-direction changes buoyancy forces, so restoring forces (also called hydrostatic forces) acting on the body is calculated:

$$\begin{aligned} Z &= mg - \rho_w g[\nabla + \delta\nabla(z)] \\ &= -\rho_w g \delta\nabla(z) \end{aligned} \tag{2.80}$$

To simplify (2.80) some assumptions are made. Firstly, the rigid body of the surface vehicle is assumed box-shaped. Water planer area which is the function of z position, $A_{wp}(z)$, is accepted to be constant, i.e. $A_{wp}(z) = A_{wp}(0)$. The density of the water is also assumed to be constant. Restoring forces is rewritten in compact and components forms:

$$Z \approx -\rho_w g A_{wp} z = z_0 z \tag{2.81}$$

$$\delta \boldsymbol{f}_r^b = \begin{bmatrix} 0 \\ 0 \\ Z \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ z_0 z \end{bmatrix} \tag{2.82}$$

where $z_0 = -\rho_w g A_{wp}$ is a constant. The final form of the restoring forces are explained in the body-fixed frame using (2.6) and (2.82):

$$\begin{aligned} \delta \boldsymbol{f}_r^b &= \boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb})^{-1} \delta \boldsymbol{f}_r^n = \boldsymbol{R}_n^b(\boldsymbol{\Theta}_{bn})^{-1} \delta \boldsymbol{f}_r^n \\ &= z_0 z \begin{bmatrix} -\sin(\theta) \\ \cos(\theta)\sin(\phi) \\ \cos(\theta)\cos(\phi) \end{bmatrix} \end{aligned} \tag{2.83}$$

Furthermore, restoring force in the z-direction can be written:

$$\delta \boldsymbol{f}_r^b = \boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb})^{-1} \delta \boldsymbol{f}_r^n = \boldsymbol{R}_n^b(\boldsymbol{\Theta}_{bn}) \begin{bmatrix} 0 \\ 0 \\ -\rho_w g \nabla \end{bmatrix} = -\rho_w g \nabla \begin{bmatrix} -\sin(\theta) \\ \cos(\theta)\sin(\phi) \\ \cos(\theta)\cos(\phi) \end{bmatrix} \tag{2.84}$$

The contribution of restoring forces defined in (2.78) to moments is negligibly small because buoyancy force is considerably larger. Cross product operation is done between moment arms (in roll and pitch) and buoyancy force to find moments which arise from buoyancy forces. Moments arm is obtained from 2.4 as follows:

$$\boldsymbol{r}_r^b = \begin{bmatrix} -\overline{GM}_T \sin(\theta) \\ \overline{GM}_L \sin(\phi) \\ 0 \end{bmatrix} \tag{2.85}$$

Cross product between (2.85) and (2.84) gives:

$$\boldsymbol{m}_r^b = \boldsymbol{r}_r^b \times \delta \boldsymbol{f}_r^b = -\rho_w g \nabla \begin{bmatrix} \overline{GM}_T \sin(\phi)\cos(\theta)\cos(\phi) \\ \overline{GM}_L \sin(\theta)\cos(\theta)\cos(\phi) \\ (-\overline{GM}_L \cos(\theta) + \overline{GM}_T)\sin(\phi)\sin(\theta) \end{bmatrix} \tag{2.86}$$

Restoring forces and moments can be written in vector form as

$$\boldsymbol{\tau}_g = \begin{bmatrix} \delta \boldsymbol{f}_r^b \\ \boldsymbol{m}_r^b \end{bmatrix} = \begin{bmatrix} -z_0 z \sin(\theta) \\ z_0 z \cos(\theta)\sin(\phi) \\ z_0 z \cos(\theta)\cos(\phi) \\ -\rho_w g \nabla \overline{GM}_T \sin(\phi)\cos(\theta)\cos(\phi) \\ -\rho_w g \nabla \overline{GM}_L \sin(\theta)\cos(\theta)\cos(\phi) \\ -\rho_w g \nabla (-\overline{GM}_L \cos(\theta) + \overline{GM}_T)\sin(\phi)\sin(\theta) \end{bmatrix} \tag{2.87}$$

When the roll and pitch angles are near zero or too small, and displacement in the z-direction is quite small. Small angle assumption can be implemented to reduce calculation. Trigonometric functions can be arranged such that $sin(\theta) \approx \theta$, $sin(\phi) \approx \phi$, $\cos(\theta) \approx 1$, $\cos(\phi) \approx 1$. By considering this assumption, (2.87) becomes:

$$\boldsymbol{\tau}_g = \begin{bmatrix} \delta \boldsymbol{f}_r^b \\ \boldsymbol{m}_r^b \end{bmatrix} = \begin{bmatrix} -z_0 z \theta \\ z_0 z \phi \\ z_0 z \\ -\rho_w g \nabla \overline{GM}_T \phi \\ -\rho_w g \nabla \overline{GM}_L \theta \\ -\rho_w g \nabla (-\overline{GM}_L + \overline{GM}_T)\phi\theta \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ z_0 z \\ -\rho_w g \nabla \overline{GM}_T \phi \\ -\rho_w g \nabla \overline{GM}_L \theta \\ 0 \end{bmatrix} \tag{2.88}$$

Meta-centric heights can be determined by basic hydrostatic [22]. Transverse, $\overline{BM}_T$, and longitudinal, $\overline{GM}_L$, radii of curvature may be estimated:

$$\overline{GM}_T = \frac{I_T}{\nabla}, \quad \overline{GM}_L = \frac{I_L}{\nabla} \tag{2.89}$$

where ∇ is the displaced water volume, I_T and I_L which are the moments of area about water plane are calculated as follows:

$$I_T = \int \int_{wp}^{A} y^2 dA, \quad I_L = \int \int_{wp}^{A} x^2 dA \tag{2.90}$$

If water planer area is taken as rectangular, i.e. $A_{wp} = BL$ where B and L are the beam and length of the craft, (2.90) can be approximated as:

$$I_T \approx \frac{1}{12} B^3 L, \quad I_L \approx \frac{1}{12} L^3 B \tag{2.91}$$

Traverse and longitudinal meta-centric are defined as, respectively

$$\begin{aligned} \overline{GM}_T &= \overline{BM}_T - \overline{BG} \\ \overline{GM}_L &= \overline{BM}_L - \overline{BG} \end{aligned} \tag{2.92}$$

2.8 Thruster Forces

As it was previously mentioned vehicle includes two thrusters that are located at the aft. One of them is on the left side, and other is on the right side of the craft. Assuming that effect of the rotation is negligible, moments produced by the left and right thrusters are formulated as follows:

$$\boldsymbol{m}_R = \boldsymbol{r}_{t_R} \times \boldsymbol{f}_{t_R} \tag{2.93}$$

$$\boldsymbol{m}_L = \boldsymbol{r}_{t_L} \times \boldsymbol{f}_{t_L} \tag{2.94}$$

where $\boldsymbol{r}_{t_R}$ and $\boldsymbol{r}_{t_L}$ represent position vectors (lever arm) from the centre of gravity to the right and left thrusters, respectively. $\boldsymbol{f}_{t_R}$ indicates the force produced by right thruster, and $\boldsymbol{f}_{t_L}$ is used for the force which is generated by the left thruster. $\boldsymbol{m}_R$, $\boldsymbol{m}_L$ are the moments induced by the right and left thrusters, respectively. Combination of these forces and moments is written in vector form as follows:

$$\boldsymbol{\tau}_t = \begin{bmatrix} \boldsymbol{f}_{t_R} \\ \boldsymbol{m}_R \end{bmatrix} + \begin{bmatrix} \boldsymbol{f}_{t_L} \\ \boldsymbol{m}_L \end{bmatrix} \tag{2.95}$$

$\boldsymbol{\tau}_t$ represents the forces and moments produced by thrusters in which it is defined in the right side of the (2.5).

2.9 Air Drag Forces

A substantial part of the vehicle moves in the air. Consequently, wind and air drag give rise to forces and moments on the vehicle. These are indicated by a vector form and denoted by $\boldsymbol{\tau}_a$ in (2.5). They are proportional to the velocity of the vehicle with respect to air or wind velocity. This relative velocity, $\boldsymbol{v}_r$, is found by subtracting the velocity, $\boldsymbol{v}_a$, of the air from the linear velocity of the body $\boldsymbol{v}_b$ in (2.96) [4].

$$\boldsymbol{v}_r = \boldsymbol{v}_b - \boldsymbol{v}_a \tag{2.96}$$

Now air drag forces are indicated by:

$$\boldsymbol{F}_a = A_a \boldsymbol{P}_a C_{d,a} = [F_{ax} \;\; F_{ay} \;\; F_{az}]^T. \tag{2.97}$$

In the left-hand side, $\boldsymbol{F}_a$ stands for forces of air drag which act on the rigid-body. A_a is the buoyant part of the vehicle in the air. $C_{d,a}$ represents the air drag coefficient. Air pressure is denoted by $\boldsymbol{P}_a$, and it is approximated by [21]:

$$\boldsymbol{P}_a \approx 2.56 \boldsymbol{v}_r^2 \tag{2.98}$$

(2.98) states that air pressure is proportional with the square of relative velocity. Some underlying assumptions are made to find torques which result from the cross product between the moment arm and the air drag forces acting on the body. The first assumption is that the body is symmetrical, small and box-shaped contrary to a large vessel. Secondly, the air drag forces are equally distributed along the buoyant part of the body, and the centre of gravity is taken as a midpoint of the craft. From this assumptions, the moment arms and torques produced by air drag force are written:

$$\boldsymbol{\lambda} \approx \begin{bmatrix} \frac{l}{2} & \frac{wi}{2} & \frac{h}{2} \end{bmatrix}^T \tag{2.99}$$

$$\boldsymbol{T}_a = \boldsymbol{\lambda} \times \boldsymbol{F}_a \tag{2.100}$$

where l, wi and h represent the length, width, height of the body, respectively. $\boldsymbol{T}_a$ indicates moments of the air drag forces. It is assumed that aerodynamic centre of the vehicle coincides with the gravitational centre. So, vector form of the air drag moments and torques are explained as [17]:

$$\boldsymbol{T}_a = -[F_{ax}\,\mathrm{sgn}(u) \;\; F_{ay}\,\mathrm{sgn}(v) \;\; F_{az}\,\mathrm{sgn}(w)]^T \tag{2.101}$$

$$\boldsymbol{\tau}_a = [\boldsymbol{F}_a^T \;\; \boldsymbol{T}_a^T]^T \tag{2.102}$$

u, v and w are linear velocity components. The sign function $\mathrm{sgn}(\bullet)$ in (2.101) is expressed as with $x \in \mathbb{R}$:

$$\mathrm{sgn}(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases} \tag{2.103}$$

2.10 Environmental (Disturbance) Forces

Environmental forces and moments that are produced from different kinds of disturbance sources may be classified into three main environmental categories as follows:

- Waves which are induced by wind ($\boldsymbol{\tau}_{wave}$)
- Wind effects ($\boldsymbol{\tau}_{wind}$)
- Ocean currents ($\boldsymbol{\tau}_{current}$)

All the disturbance forces are taken as only one force, $\boldsymbol{\tau}_{dis}$, in the left-hand side of the (2.5). From these three forces, since wind-generated forces which are the modeled forces for surface vehicles are taken into account in the mathematical model.

Wind-generated waves are composed of a summation of a large number of components. For the sake of simplicity, wave spectra are considered with one peak frequency. A_i denotes wave amplitude, and i is the wave component which is related to the density function of the spectrum. Square of the wave amplitude has relation with spectrum as follows:

$$A_i^2 = 2S(w_i)\triangle w \tag{2.104}$$

where w_i is the circular random frequency of the $i-th$ component. It takes value in the frequency interval $\triangle w$ which is described as a constant difference between two successive frequencies. Wave number of the single wave component, k_i, can be found that 2π is divided by the wavelength of the $i-th$ wave component, λ_i.

$$k_i = \frac{2\pi}{\lambda_i} \tag{2.105}$$

Using (2.104) and (2.105) wave elevation, $\zeta(x,t)$, which is the function of x-(forward) position and time is written with its first order and second order components:

$$\zeta(x,t) = \sum_{i=1}^{N} A_i \cos(w_i t - k_i x + \phi_i) + \sum_{i=1}^{N} \frac{1}{2} A_i^2 \cos 2(w_i t - k_i x + \phi_i) \quad (2.106)$$

where ϕ_i is the random phase angle which is uniformly distributed in $[0, 2\pi)$ and is independent of time. For example, the density function of the Pierson-Moskowitz spectrum is used to obtain first-order wind-generated wave forces [21]. Spectrum is the function of gravitational constant, g, and significant wave height, H_s.

$$S(w) = Aw^{-5} e^{-Bw^{-4}} \quad (2.107)$$

where $A = 8.1 \times 10^{-3} g^2$ and $B = \dfrac{3.11}{H_s^2}$. When the values of A and B are substituted in (2.107), the final form of the spectral density function obtained as follows:

$$S(w) = 8.1 \times 10^{-3} g^2 w^{-5} e^{\frac{3.11}{H_s^2} w^{-4}} \quad (2.108)$$

First order wind-generated wave forces which consist of the surge, sway forces and yaw torque is based on block shape ship assumption. To be able to find these forces, wave slope must be found from the derivation of the first term of (2.106) with respect to x-position. Wave slope of $i-th$ wave component, s_i, calculated as:

$$s_i(x,t) = \frac{\partial \zeta(x,t)}{\partial x} = A_i k_i \sin(w_i t - k_i x + \phi_i) \quad (2.109)$$

For a moving vehicle, w_i is replaced with w_e and above equation is simply written by assuming $x = 0$. (2.109) becomes:

$$s_i(t) = s_{0,i}(t) = A_i k_i \sin(w_e t + \phi_i) \quad (2.110)$$

As a result, first order disturbance forces in surge and sway and moment in yaw are formulated:

$$X_{wave}(t) = \sum_{i=1}^{N} \rho_w g B L T \cos(\beta - \Psi) s_i(t) \quad (2.111)$$

$$Y_{wave}(t) = \sum_{i=1}^{N} -\rho_w g B L T \sin(\beta - \Psi) s_i(t) \quad (2.112)$$

$$N_{wave}(t) = \sum_{i=1}^{N} \frac{1}{24} \rho_w g B L (L^2 - B^2) \cos(\beta - \Psi) s_i(t) \quad (2.113)$$

where B and L indicate beam and length of the surface vehicle, respectively. Draft of the submerged part of the vehicle is represented by T. The angle between the heading of ship and wave direction is mathematically explained by $\beta - \Psi$. Disturbance forces and torques vector is written in the following equation by using from (2.111) to (2.113):

$$\boldsymbol{\tau}_{dis} = [X_{wave} \;\; Y_{wave} \;\; 0 \;\; 0 \;\; 0 \;\; N_{wave}]^T. \tag{2.114}$$

2.11 Implementation

The mathematical model is derived in the continuous-time domain using Newtonian or Lagrangian dynamics. In order to implement the mathematical model which is defined earlier in this chapter, some discretization techniques must be applied in a computer environment. To able to discretize model, forward and backward Euler integration which is comprehensively expressed in [22] is used. This method is accepted as a stable method for the under-damped second-order system and utilized for nonlinear models to discretize them. If this method applied to (2.11) and (2.13), two discrete equations are obtained. Forward and backward Euler equations are represented respectively,

$$\boldsymbol{\nu}(k+1) = \boldsymbol{\nu}(k) + T_s \left\{ \boldsymbol{M}^{-1}(\boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\nu}(k))\boldsymbol{\nu}(k) \right\} \tag{2.115}$$

$$\boldsymbol{\eta}(k+1) = \boldsymbol{\eta}(k) + \boldsymbol{J}_{\Theta}(\boldsymbol{\eta}(k))\boldsymbol{\nu}(k+1) \tag{2.116}$$

In (2.115), T_s represent the time interval, i.e., sampling time, which is taken as 0.05 second.

2.12 Simulation Results

A MATLAB GUI which calls the function of the discretized mathematical model is used for simulation purposes. This GUI can be seen at the following in Fig. 2.5. In this part, behaviour of the vehicle are investigated without giving an initial torque and with providing a torque input to the system.

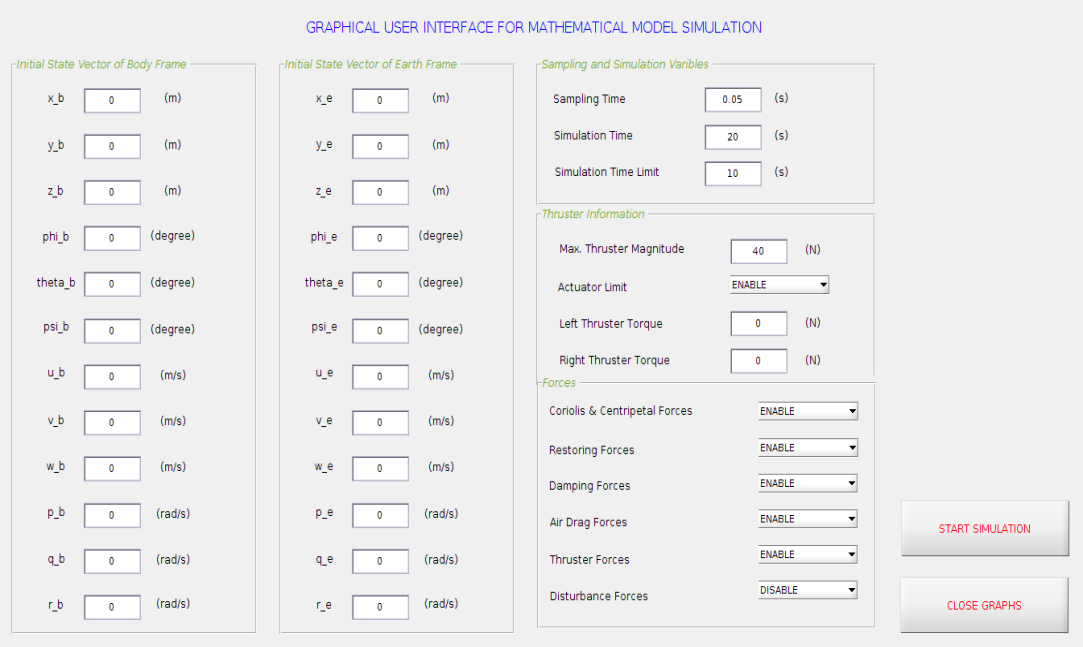

Figure 2.5: Graphical user interface for the mathematical model.

2.12.1 Case study: Zero Input – Zero Initial State

In this state, it is expected that the boot is stationary. Acting forces on the vehicle do not change with time. Absolute value buoyancy force is equal to the absolute value of the gravitational force. So, the rate of change in the linear velocity and the angular velocity is zero. Damping forces and moments are not available in this scenario. Simulation time is taken five seconds. It is understood from Figs. 2.6 and 2.7 that initial states of the system $\boldsymbol{\eta}(k) = \mathbf{0}$, $\boldsymbol{\nu}(k) = \mathbf{0}$ will remain zero for all k.

2.12.2 Case study: Zero Input – Nonzero Initial State

This case is investigated in two parts which are composed of roll and pitch rotation. In both cases, the boat is rotated about +10 degrees. In this scenario, it is understood that boot becomes stable and stationary after a while. During this simulation, restoring and damping force dominantly act on the vehicle to make it stable in a fluid. Simulation time is taken as 20 seconds for both cases.

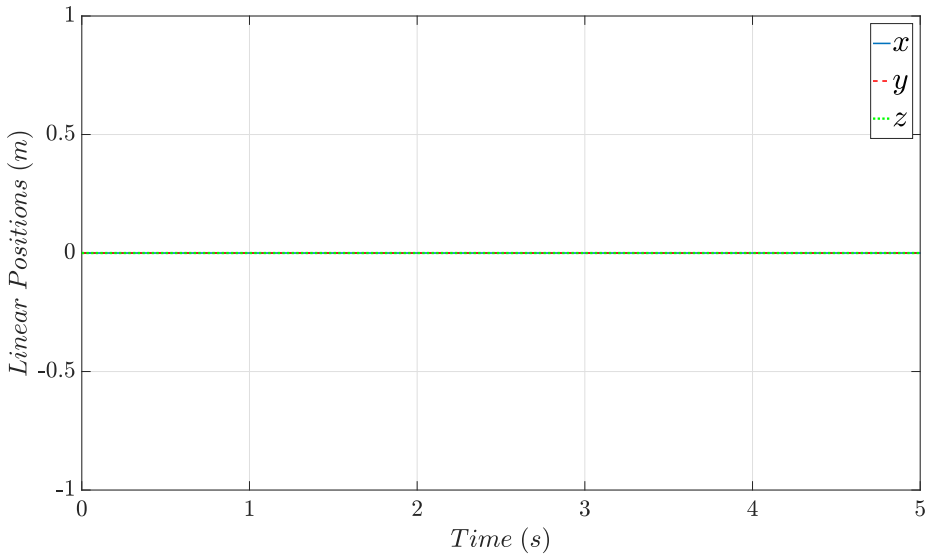

Figure 2.6: Case study: zero input zero state - linear position of the system.

2.12.2.1 +10 Degrees Roll Rotation

Giving an initial roll rotation to the system makes a small change in the y-direction. Position in x, y and z directions becomes zero after some time, as can be seen from Fig. 2.8. Angular position in roll direction has an exponentially decreasing oscillation

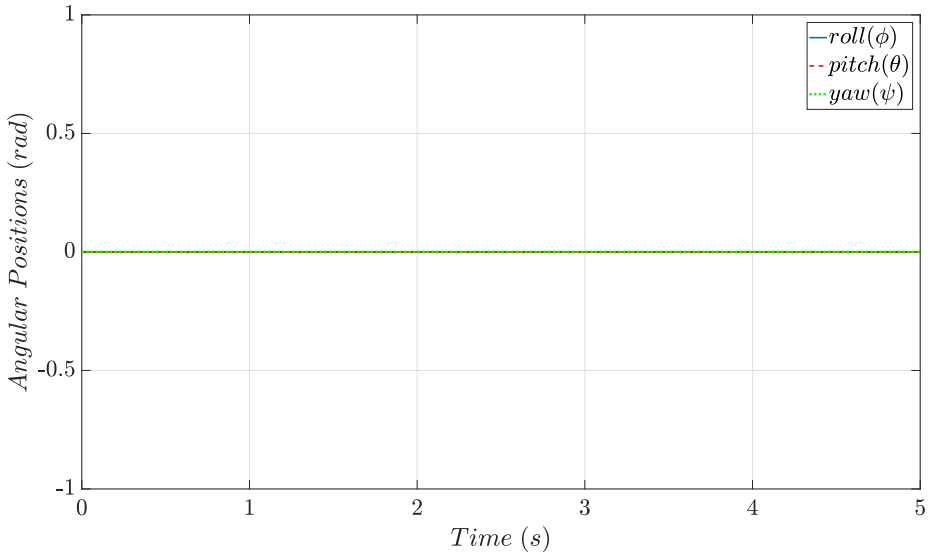

Figure 2.7: Case study: zero input zero state - angular position of the system.

due to the dominant damping force. This oscillation can be observed in Fig. 2.9.

2.12.2.2 +10 Degrees Pitch Rotation

When the ship is stable and stationary in standing fluid, it is tough to perform a test about pitch rotation. So simulation results give useful information about pitch rotation. The scenario is made by rotating the vehicle in the pitch rotation about +10 degrees. It can be seen from Fig. 2.10 that there is no displacement in 3-D. Due to the fact that the vehicle has an oscillation in a fluid, the dominant force acting on the vehicle is naturally restoring force. Exponentially deceased oscillation arising from restoring force can be seen in Fig. 2.11.

2.12.2.3 -0.1 Meter Submersion of the Vehicle in Fluid

For this test, the vehicle is pushed from its centre of gravity into the water about 0.1 meters in the z-direction. Oscillation is observed in the z-direction in Fig. 2.12, until gravitational and buoyancy forces balance each other. In the end, it is found out that restoring force becomes dominant in this scenario. Because of the fact that there is no rotating forces and torques, it can be seen in Fig. 2.13 that rotation in roll, pitch

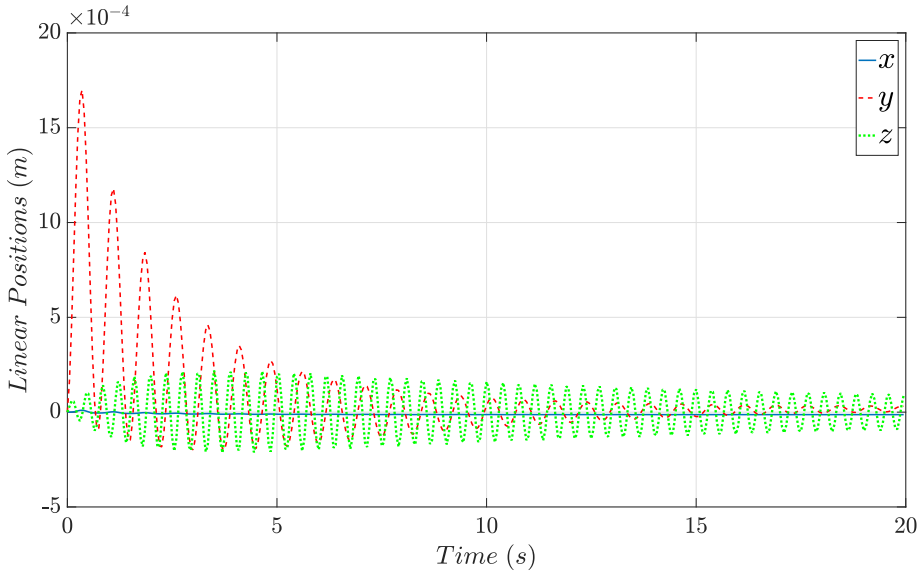

Figure 2.8: Case study: +10 degrees roll rotation - linear position of the system.

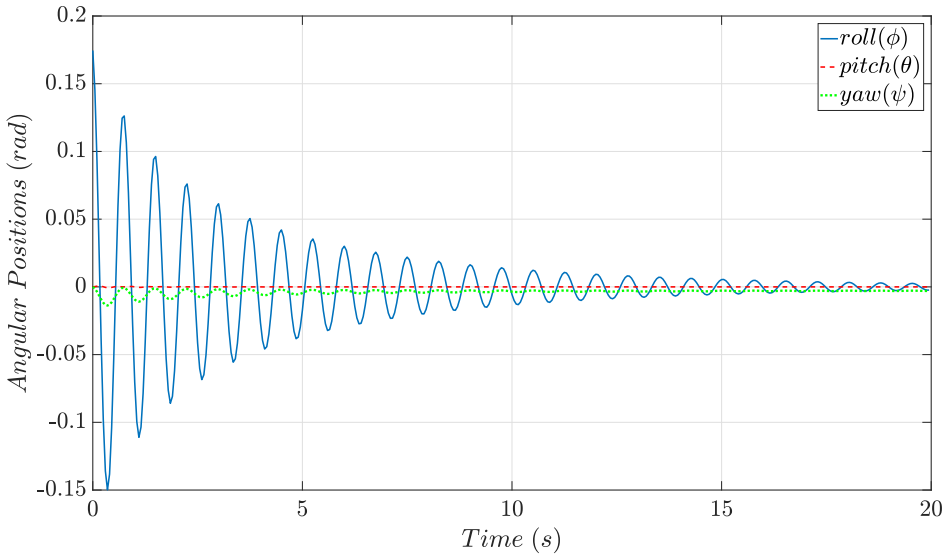

Figure 2.9: Case study: +10 degrees roll rotation - angular position of the system.

and yaw direction preserve initial values.

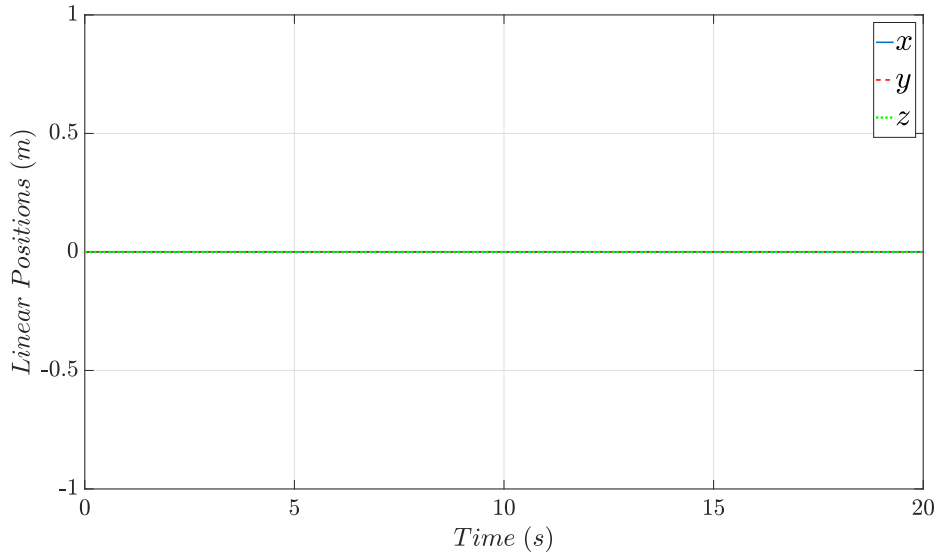

Figure 2.10: Case study: +10 degrees pitch rotation - linear position of the system.

2.12.3 Case study: Nonzero Input – Zero Initial State

In this scenario, it is given initial torque to the system in order to observe the behaviour of the vehicle. Firstly, the same amount of torques applied to the left and right thrusters. When the same amount of torques applied both thrusters, it is expected that the vehicles start moving or preserve in the forward direction (x-direction). Another test is related to the differently valued torques to the thrusters. When a different amount of torques applied to the thrusters, vehicle heads for the side of larger applied torque. Simulation time is determined as 5 seconds for two tests.

2.12.3.1 Equal Thruster Inputs – Zero Initial States

Applied torques to the left and right thruster is given, $\boldsymbol{\tau}_t^L = [5 \quad 0 \quad 0]^T$ and $\boldsymbol{\tau}_t^R = [5 \quad 0 \quad 0]^T$, respectively. Linear position and angular rate can be seen Fig. 2.14 and 2.15, respectively.

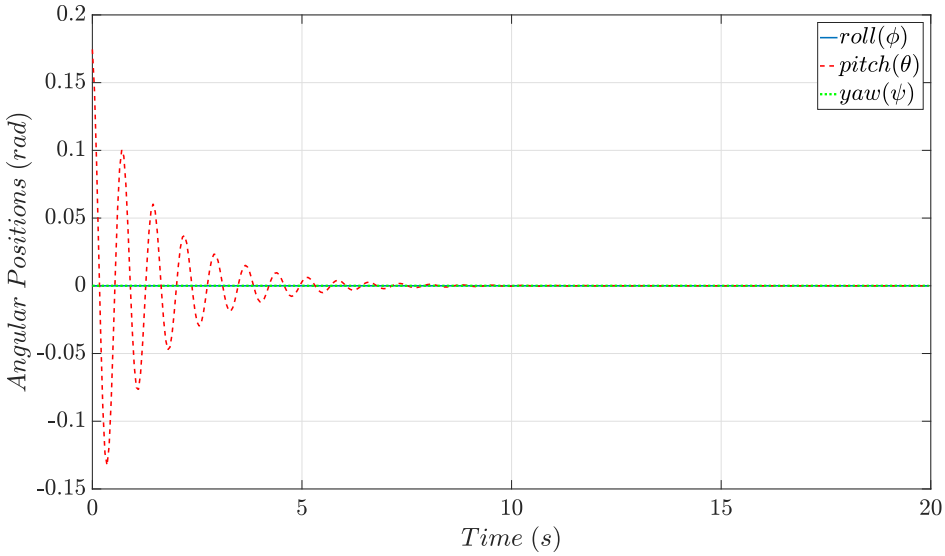

Figure 2.11: Case study: +10 degrees pitch rotation - angular position of the system.

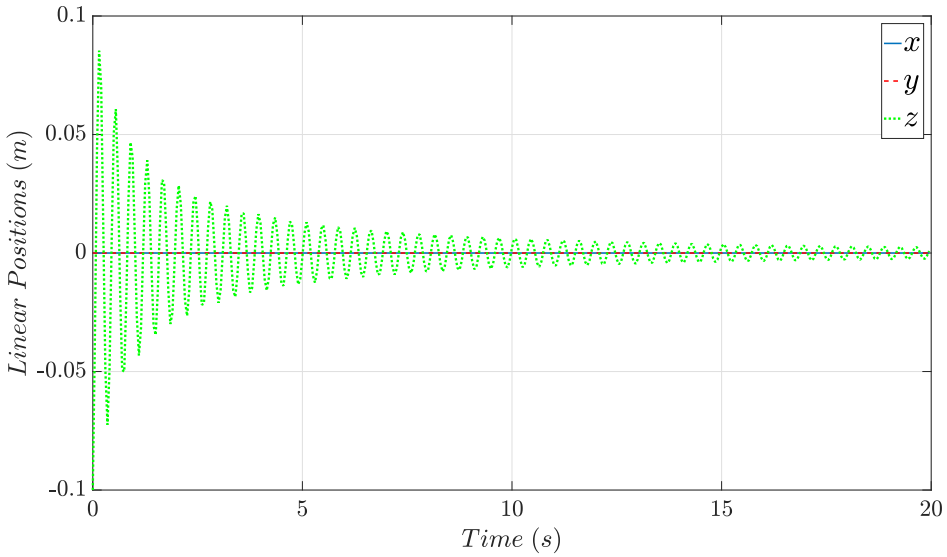

Figure 2.12: Case study: -0.1 meter submersion - linear position of the system.

2.12.3.2 Non equal Thruster Inputs – Zero Initial States

Applied torques to the left and right thruster is given, $\boldsymbol{\tau}_t^L = [2 \quad 0 \quad 0]^T$ and $\boldsymbol{\tau}_t^R = [5 \quad 0 \quad 0]^T$, respectively. Linear position and angular rate can be seen Fig. 2.16 and 2.17, respectively.

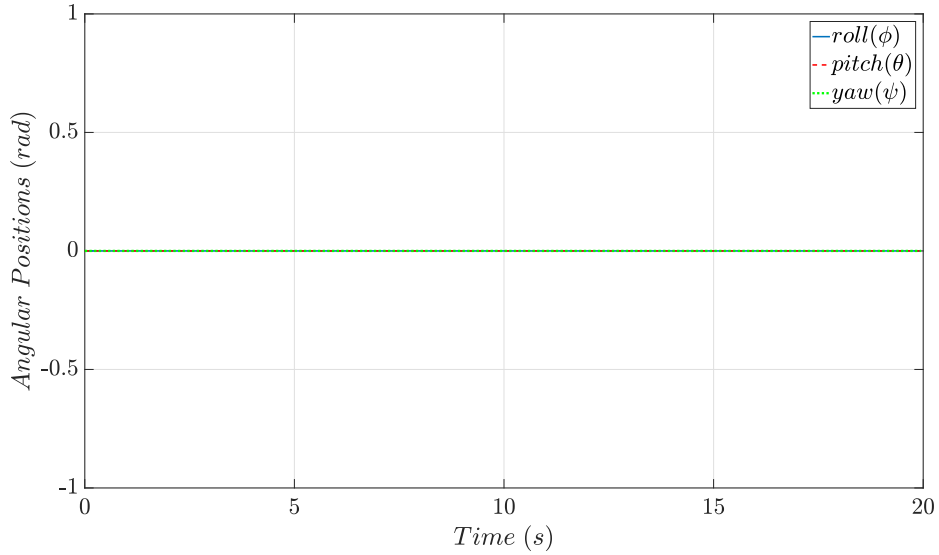

Figure 2.13: Case study: -0.1 meter submersion - angular position of the system.

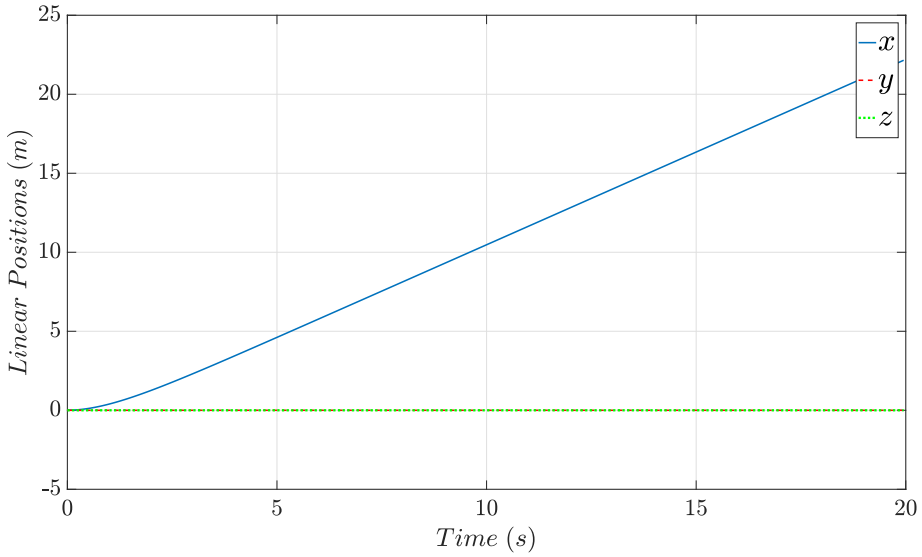

Figure 2.14: Case study: equal thruster inputs and zero initial states - linear position of the system.

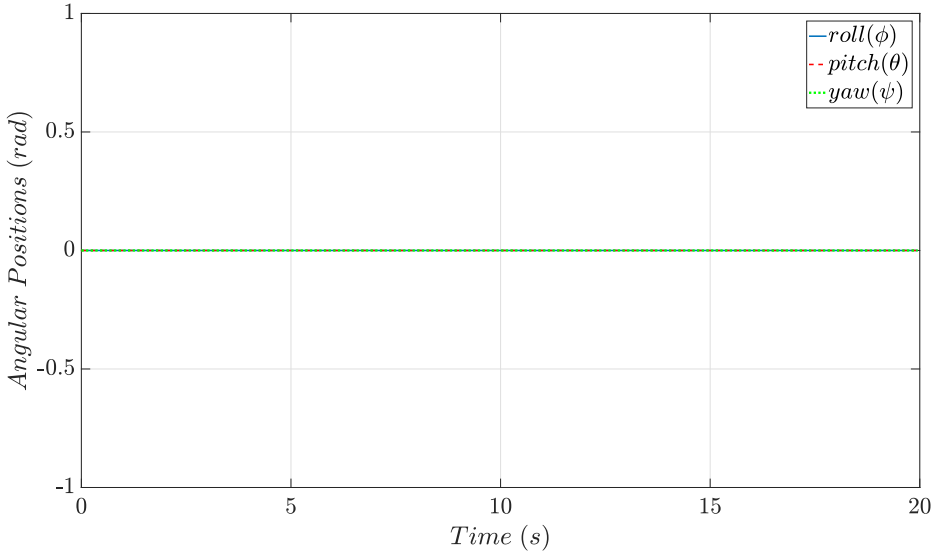

Figure 2.15: Case study: equal thruster inputs and zero initial states - angular position of the system.

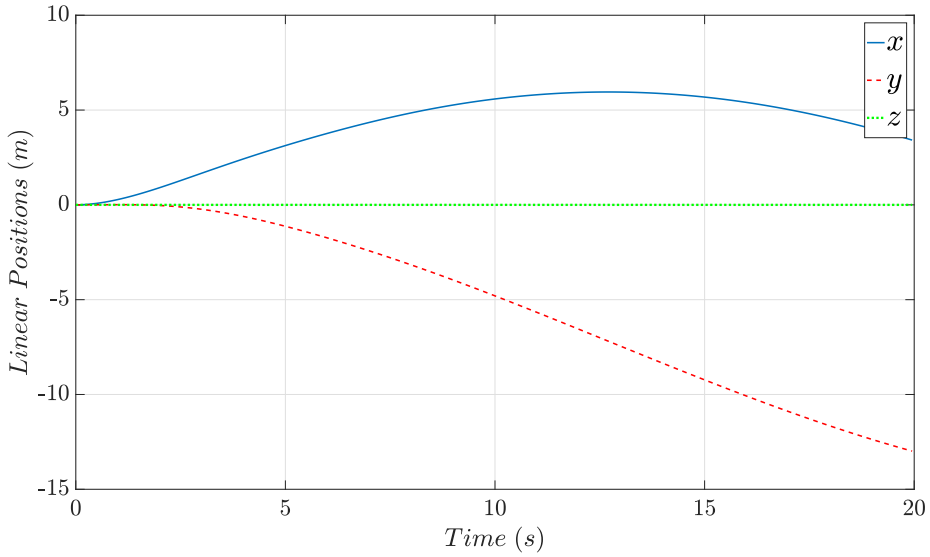

Figure 2.16: Case study: non-equal thruster inputs and zero initial states - linear position of the system.

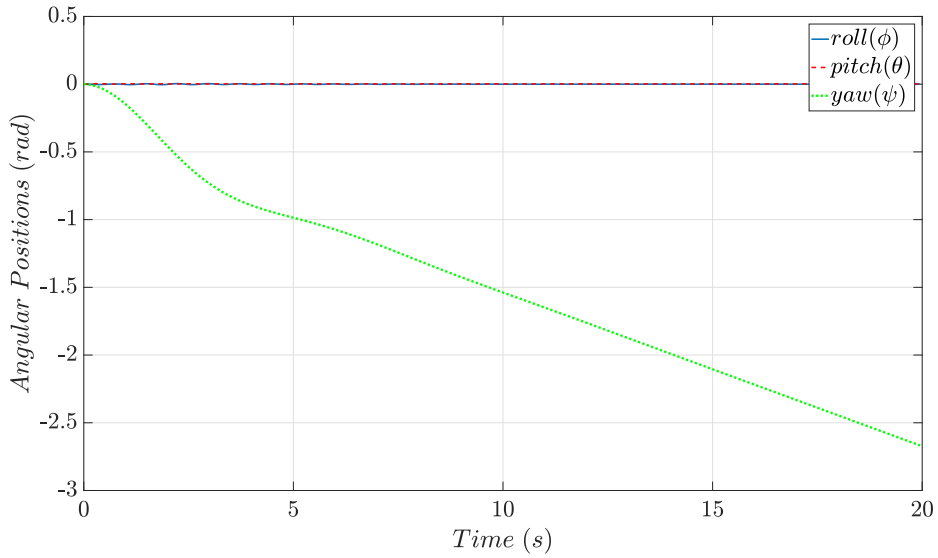

Figure 2.17: Case study: non-equal thruster inputs and zero initial states - angular position of the system.

CHAPTER 3

GUIDANCE

3.1 Introduction

Guidance in terms of engineering perspective is defined as the process for guiding the predefined or subsequently identified a path of an object with respect to the given point which may be stationary or moving [37]. This given point is generally called way-point, but it is named as a target in missile literature. A vehicle or an autonomously controlled system (such as a surface, a marine or an aerial vehicle) may be a guided object. At the beginning of the 20^{th} century, guidance was firstly implemented for unmanned boats which are remotely controlled for military purposes. [37]. Then it is used for complicated control systems in a wide range of applications, e.g., from ground vehicles to ballistic missiles.

The motion of a USV is investigated in some categories: path following, trajectory tracking and point stabilization according to traditional guidance literature [10]. Path manoeuvring problem fills the gap between trajectory tracking and path following problems [46]. In target tracking, a stationary or moving target whose instantaneous position and velocity are known is tracked to realize guidance mission. Path following problem is described as following a predefined path which is composed of way-points. In contrast to path following; trajectory tracking is used to follow a path of the target by a guided object along a calculated trajectory that is predefined. The objective of the path manoeuvring is to obtain vehicle manoeuvres by considering constraints on path and vehicles on the predefined path.

Control of the motion of the unmanned surface vehicle whose mathematical model is obtained in the previous chapter is investigated in three levels as it is seen in Fig. 3.1.

These levels are called strategic, tactical and execution control levels, [10], [51]. The first level is called as strategic, or organization level includes kinematic calculations which come from a human operator and/or guidance system. Strategic level sometimes is named as kinematic control which can directly correspond guidance laws. In this level, reference information, e.g., vehicle surge speed and yaw angle are obtained via guidance rules from the vehicle position and way-point or target position using geometric calculation. After that, applied torque and moment for actuators are determined by a kinetic controller to be able to generate desired motion. These controllers should overcome the uncertainties of the system parameters and environmental effects. Control allocation assists kinetic control block to distribute input command or signal to actuators of the vehicle. At final, individual actuator controllers produce fixed sample rate signals to motors in execution level. So, it can be understood that whether vehicle achieves to move the desired position or not, that is determined by guidance law.

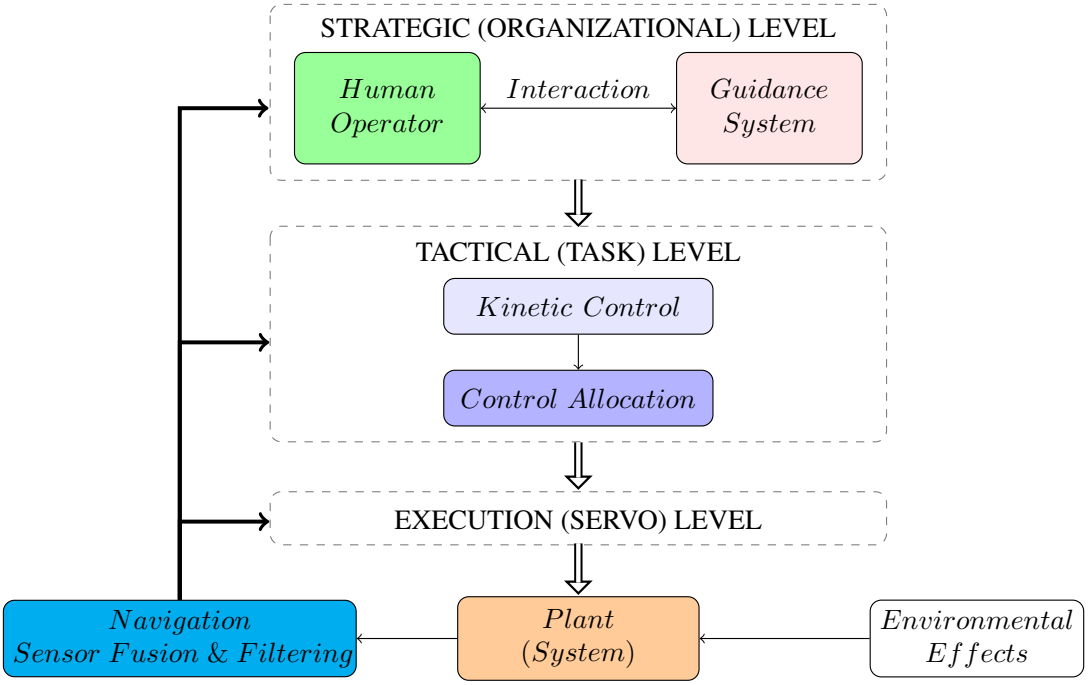

Figure 3.1: Levels of the motion control.

3.2 Guidance Laws

For an under-actuated USV with two thrusters, guidance law generally produces surge speed and yaw angle references for autopilots [5]. Due to the fact that vehicle moves on the water surface, path following or target tracking purposes is achieved in 2-D via some guidance techniques. As can be seen in Fig. 3.2, only kinematic information of the vehicle and predetermined way-points are taken as inputs. In a path following problem, reference commands for autopilots may be obtained by reducing the distance between the planar way-point, $\boldsymbol{p}_w[k] \triangleq [x_w[k] \quad y_w[k]]^T \in \mathbb{R}^2$, [10] and the position of the vehicle, $\boldsymbol{p}[k] \triangleq [x[k] \;\; y[k]]^T \in \mathbb{R}^2$:

$$\lim_{k\to\infty} (\boldsymbol{p}_w - \boldsymbol{p}[k]) = \boldsymbol{0} \tag{3.1}$$

where $\boldsymbol{p}_w[k]$ is taken as $\boldsymbol{p}_w$ since way-points are assigned as fixed points on a path throughout the motion of the vehicle in the scope of this study. Once, the USV enters or hits to the neighbourhood of a way-point, which is called circle of acceptance, guidance algorithms are calculated for the subsequent way-point to generate reference surge speed and yaw angle. A circle of acceptance (CoA), which can be seen in Fig. 3.3, around each way-point, is expressed as follows:

$$\sqrt{(x_w - x[k])^2 + (y_w - y[k])^2} \leq R_w \tag{3.2}$$

where R_w is the radius of CoA for way-point $\boldsymbol{p}_w$. Radius information is stored in the way-points table because there are different CoA values for each way-point. The

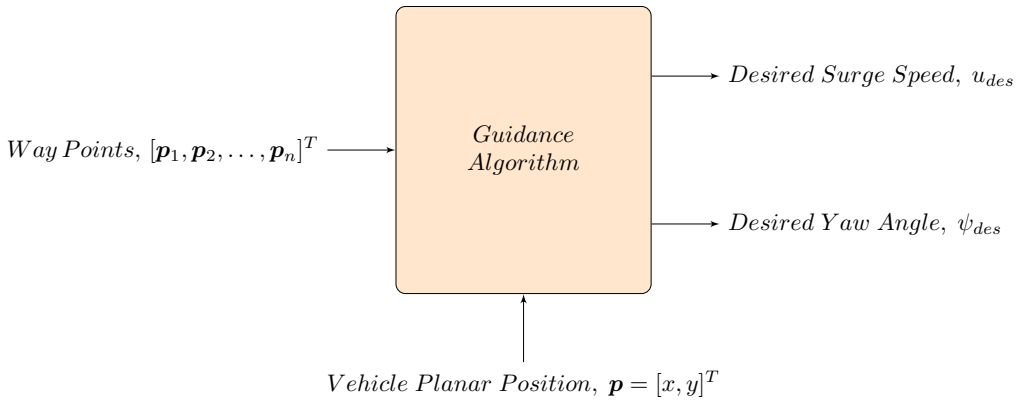

Figure 3.2: Representation of a basic guidance system for a USV by using block diagram.

value of a CoA is practically taken as one or two-fold of the USV length for path tracking. The radius of CoA may be arranged according to manoeuvre requirements for parallel docking (parking) case. Each way-point includes some information that composes of x and y position of way-point, motion direction which can be backward and forward, radius circle of acceptance (CoA), and surge speed. The yaw angle is calculated according to the position information. Then, path tracking is achieved with the help of reference yaw angle and other way-point information. Table 3.1 which can be considered an array whose rows are composed of each way-point information is represented below.

Table 3.1: Way-point information for motion control hierarchy.

way-point index	x position	y position	Motion direction (Forward or Backward)	CoA	Reference surge speed
1	x_1	y_1	1 or 2	R_1	u_1
2	x_2	y_2	1 or 2	R_2	u_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n-1$	x_{n-1}	y_{n-1}	1 or 2	R_{n-1}	u_{n-1}
n	x_n	y_n	1 or 2	R_n	u_n

n way-points are described in Table 3.1. For embedded programming, this table can be written in the program or sent to program as an array in real-time. Planar position of way-point i is represented by $\boldsymbol{p_i} = [x_i \ y_i]^T \ \forall \ i \in \mathbb{Z}_{[1,n]}$. Forward and backward motions are represented with integers as 1 and 2, respectively. Controllers produce forces and torques according to motion in order to move forward and backward direction. This information is also important for forward or backward desired yaw angle calculations which are explained later in this section. Circle of acceptance and reference surge speed data are expressed as R_i and $u_i \ \forall \ i \in \mathbb{Z}_{[1,n]}$, respectively.

There are lots of suitable guidance techniques to achieve way-point tracking and path following performances. Whereas some techniques need three points which are the

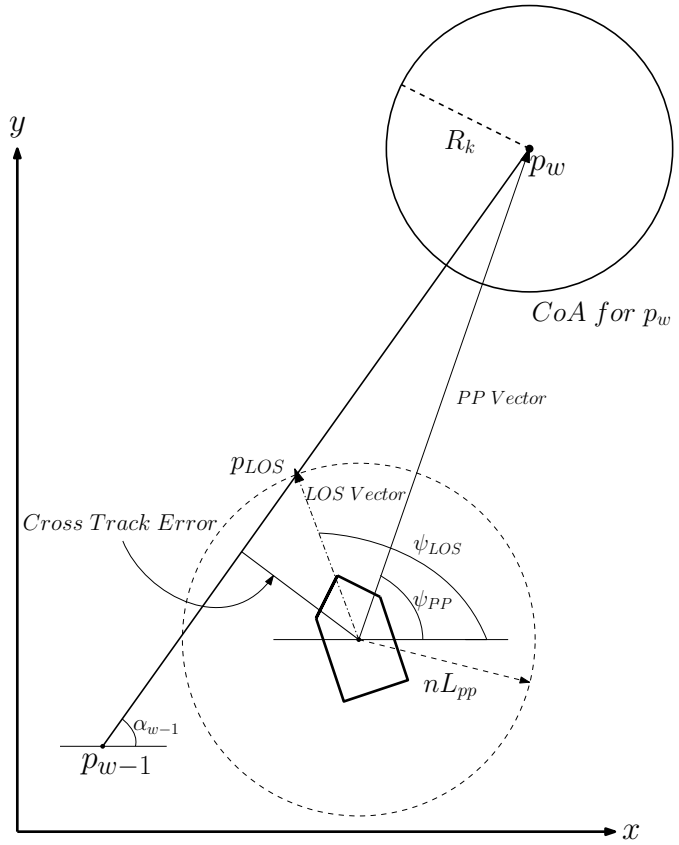

Figure 3.3: Representation of pure-pursuit (PP) and line-of-sight (LOS) guidance techniques in 2D.

previous way-point, $\boldsymbol{p}_{w-1}$, current way-point, $\boldsymbol{p}_w$, and vehicle surface position, $\boldsymbol{p}[k]$, some only use $\boldsymbol{p}_w$ and $\boldsymbol{p}[k]$ to give reference autopilot inputs. Line-of-sight (LOS), is categorised as three-point scheme, and pure-pursuit (PP), is classified as a two-point guidance method. Strategies are presented and implemented to be able to carry out path following and parking capabilities of the vehicle.

3.2.1 Line-of-Sight (LOS) Guidance

LOS method directs the orientation of the craft into the direction along the line between two successive way-points. As shown in Fig. 3.3, the geometry of the LOS approach may be described with the current way-point $\boldsymbol{p}_w$, the previous way-point $\boldsymbol{p}_{w-1}$, and the instantaneous vehicle position $\boldsymbol{p}[k]$. $\boldsymbol{p}[k]$ is situated at the centre of a circle with the radius of n-fold of the ship length: nL_{pp} [23]. The line-of-sight point,

$\boldsymbol{p}_{LOS}$, is the point which is on the line segment between way-point $\boldsymbol{p}_{w-1}$ and $\boldsymbol{p}_w$. The solution of $\boldsymbol{p}_{LOS}[k]$ is obtained from the below equations (3.3) and (3.4):

$$(x_{los}[k] - x[k])^2 + (y_{los}[k] - y[k])^2 = (nL_{pp})^2 \tag{3.3}$$

$$\frac{y_{los}[k] - y_{w-1}}{x_{los}[k] - x_{w-1}} = \frac{y_w - y_{w-1}}{x_w - x_{w-1}} = \tan(\alpha_{w-1}) \tag{3.4}$$

The radius of the circle, nL_{pp}, in ((3.3)) should be sufficiently larger than the radius of CoA, R_w which is defined in ((3.2)), i.e., $nL_{pp} \geq R_w$. This relation provides the existence of a solution. $\boldsymbol{p}_{LOS}$ is selected as the closest point to $\boldsymbol{p}_w$ for forward motion. On the other hand, $\boldsymbol{p}_{LOS}$ is chosen as the closest point to $\boldsymbol{p}_{w-1}$ as the USV is driven backward [14]. Calculations to obtain $\boldsymbol{p}_{LOS}$ is comprehensively explained in Appendix A.1. Desired yaw angles, $\psi_{LOS,forward}[k]$ and $\psi_{LOS,backward}[k]$ at each sampling time are formulated as (3.5), respectively.

$$\begin{aligned} \psi_{LOS,forward}[k] &= \text{atan2}(y_{los}[k] - y[k], x_{los}[k] - x[k]) \\ \psi_{LOS,backward}[k] &= \text{atan2}(y[k] - y_{los}[k], x[k] - x_{los}[k]) \end{aligned} \tag{3.5}$$

Return of $\text{atan2}(\ ,\ )$ function is a single value in $[-\pi, \pi]$, [54]. The main reason to use this function is that $\arctan(\frac{y}{x})$ could not give correct result when $x < 0$. Sign of variable x and y in $\text{atan2}(y, x)$ are separately known to determine appropriate angle without losing information. However, $\text{atan2}(\ ,\ )$ has an important drawback due to discontinuity at the $-\pi$ or $+\pi$ junction. Rapid change in the desired yaw angle, $\psi_d[k]$, cause unintended behaviours during the motion. A reference model before tactical level implementation can be proposed to prevent abnormal heading calculation [23]. Discontinuity issue can be handled by mapping $\psi_d[k]$ angle from $[-\pi, \pi]$ to $(-\infty, \infty)$. Then the reference model gives a proper angle value. Continuous desired angle is eventually found by making re-mapping from $(-\infty, \infty)$ to $[-\pi, \pi]$. Mapping and re-mapping process are inclusively explained in Appendix A.2. Desired surge speed commands are taken as constant to fed a controller for each line segment:

$$\lim_{k \to \infty} (u_w - u[k]) = 0 \tag{3.6}$$

where u_w indicates the desired surge speed between two way-points $\boldsymbol{p}_w$ and $\boldsymbol{p}_{w-1}$. When the vehicle is in a CoA, velocity reference is calculated for next way-point.

3.2.2 Pure-Pursuit (PP) Guidance

Pure pursuit guidance algorithm, also known as two-point way-point guidance method, is applied for tracking a way-point regardless of the path as illustrated in Fig. 3.3. PP method is comprised of the instantaneous vehicle position $\boldsymbol{p}[k]$ and the interested way-point, $\boldsymbol{p}_w$. Previous way-point, $\boldsymbol{p}_{w-1}$, is allocated to memory for the cross-track error calculations. PP approach may be an appropriate choice for path following when the distance between two consecutive way-points is relatively small. Heading of the vehicle is tried to align with the closest distance between the instantaneous position of the vehicle and the interested way-point. Reference yaw angles are calculated in (3.7) for forward and backward motions, respectively.

$$\begin{aligned} \psi_{d,forward}[k] &= \mathrm{atan2}(y_w - y[k], x_w - x[k]) \\ \psi_{d,backward}[k] &= \mathrm{atan2}(y[k] - y_w, x[k] - x_w) \end{aligned} \tag{3.7}$$

Only two points, the position of the vehicle and current interested way-point position are enough to calculate the desired yaw angle for backward and forward motion. The problem that is encountered due to discontinuity in $\mathrm{atan2}(\ ,\ )$ function also exists for PP method. As in the case with LOS guidance, the same methodology is applied to obtain continuous yaw angle information for motion in Appendix A.2.

For current way-point, reference surge speed of the vehicle is given to the tactical (task) level controller to achieve velocity objective of the guidance method given in (3.6). It can be considered that the surge speed of the vehicle may be decreased in proportion to current velocity due to obtaining reasonable manoeuvre.

CHAPTER 4

CONTROLLER (AUTOPILOT) DESIGNS OF THE UNMANNED SURFACE VEHICLE

4.1 Introduction

In this chapter, model predictive control (MPC) and cascaded proportional-derivative-integral (PID) methods are implemented for the tactical level control of an unmanned surface vehicle whose mathematical model and its parameters have been obtained earlier using Fossen's well-known vectorial model and kinematic equations that are suitable for designing controllers. The vehicle has two propellers as control actuators which are driven by two motors at the aft. Applied forces to the propellers are taken as control inputs. The objective of the controllers is to reach the desired yaw angle and surge velocity references as much as possible. Since rudder is not available in boat model, yaw position is controlled by the generating imbalance between left and right thruster inputs.

Disturbance forces such as wind, current etc. are not taken into account in the design of the controllers. Left and right thrusters take their commands via control allocation which is explained in 3.1.

First of all, linearization of the state equations and appropriate sampling time selection are investigated. Then, linear MPC, (LMPC), and cascaded PID controllers are designed. Parameters of the controllers are tuned on an "S" shape curve using line-of-sight guidance algorithm which was explained in Chapter 3.2.1. Simulation results and path tracking performance of LMPC and cascaded PID are stated and compared

at the end of this chapter.

4.2 Preliminary Work Before Autopilot Design

4.2.1 Linearization of the Nonlinear System

To be able to make easier or effortless controller design, using a linear model can be one of the useful ways. Linearization of the system at an operating point gives some information about the nonlinear systems in the neighbourhood of that operating point. By obtaining state space representation, it can be understood whether the designed controller provides an adequate response or not. Following the nonlinear vectorial model equations will be linearized to design an MPC controller.

$$\dot{\boldsymbol{\eta}} = J_{\Theta}(\boldsymbol{\eta})\boldsymbol{\nu} \tag{4.1}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_t + \boldsymbol{\tau}_g + \boldsymbol{\tau}_d + \boldsymbol{\tau}_a \tag{4.2}$$

As can be understood from (4.2), disturbance forces and torques which are coming from environmental effects are not added to the system dynamics in contrary to general vectorial model. States of the linear and nonlinear model which consist of 6-DoF: positions and Euler angles, linear and angular velocities. Applied forces to left and right thrusters are taken as control inputs. States and input vector are rewritten in different forms as follows:

$$\boldsymbol{x} = [\boldsymbol{\eta}^T \;\; \boldsymbol{\nu}^T]^T \tag{4.3}$$

$$\boldsymbol{u} = [\boldsymbol{u}_L \;\; \boldsymbol{u}_R]^T \tag{4.4}$$

The closed form of the equations (4.1) and (4.2) might be represented by the vectors above. The nonlinear system equations are a function of the state and input vector. This function is written as:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \tag{4.5}$$

where $f \in \mathbb{R}^{12}$, $\boldsymbol{x} \in \mathbb{R}^{12}$ and $\boldsymbol{u} \in \mathbb{R}^2$. The following relation is obtained when (4.5) is linearized via Taylor series expansion at the operating point, $(\boldsymbol{x}_o, \boldsymbol{u}_o)$:

$$\dot{\Delta\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{x}_0) + \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x}_0, \boldsymbol{u}_0)} \Delta\boldsymbol{x} + \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{(\boldsymbol{x}_0, \boldsymbol{u}_0)} \Delta\boldsymbol{u} + H.O.T. \tag{4.6}$$

simply,

$$\dot{\Delta \boldsymbol{x}} \approx \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{x}_0) + \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x}_0, \boldsymbol{u}_0)} \Delta \boldsymbol{x} + \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{(\boldsymbol{x}_0, \boldsymbol{u}_0)} \Delta \boldsymbol{u}. \tag{4.7}$$

since $\dot{\boldsymbol{x}}_o = \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{u}_0)$, $\Delta \boldsymbol{x} = \boldsymbol{x} - \boldsymbol{x}_o$ and $\Delta \boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}_o$. $\dfrac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}} = \boldsymbol{A}'(t) \in \mathbb{R}^{12x12}$ and $\dfrac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}} = \boldsymbol{B}(t) \in \mathbb{R}^{12x2}$ are the Jacobian matrices of the nonlinear function with respect to state and input vectors, respectively. The contribution of the high order terms in the equation can be negligible, so they may be disregarded in the linearized state space representation. After substituting Jacobians at the operating point, $(\boldsymbol{x}_o, \boldsymbol{u}_o)$, and time, t, (4.5) can be represented in the state space representation as follows:

$$\begin{bmatrix} \Delta \dot{\boldsymbol{x}}(t) \\ 0 \end{bmatrix} = \underbrace{\begin{bmatrix} \boldsymbol{A}'(t) & \boldsymbol{f}(\boldsymbol{x}_0, \boldsymbol{u}_0) \\ \boldsymbol{0} & 0 \end{bmatrix}}_{\boldsymbol{A}(t)} \begin{bmatrix} \Delta \boldsymbol{x}(t) \\ 1 \end{bmatrix} + \underbrace{\begin{bmatrix} \boldsymbol{B}'(t) \\ 1 \end{bmatrix}}_{\boldsymbol{B}(t)} \Delta \boldsymbol{u}(t) \tag{4.8}$$

where $\boldsymbol{A}(t) \in \mathbb{R}^{13x13}$, $\boldsymbol{B}(t) \in \mathbb{R}^{13x2}$. In controller design, surge speed, u, and yaw angle, ψ, are taken as controlled states whose reference signals are produced via guidance algorithms. Output equations can be written in vector form as follows:

$$\boldsymbol{y}(t) = \begin{bmatrix} u \\ \psi \end{bmatrix} = \underbrace{[zeros(2,5) \;\; eye(2) \;\; zeros(2,6)]}_{\boldsymbol{C}(t)} \boldsymbol{x}(t) + \boldsymbol{D}(t)\, \boldsymbol{u}(t) \tag{4.9}$$

where $\boldsymbol{C}(t) \in \mathbb{R}^{2x13}$ matrix with zero indices except $\boldsymbol{C}_{1,6} = 1$ and $\boldsymbol{C}_{2,12} = 1$, $\boldsymbol{D}(t) = \boldsymbol{0} \in \boldsymbol{0}^{2x2}$.

4.2.2 Determination of the Appropriate Sample Time

Sample time plays a critical role in order to make reasonable and approximate systems for discrete controller design and simulate the nonlinear system in a computer environment. Zero-order hold (ZOH) is the one of the most used technique to discretize system, and it is chosen to make discrete time controllers. In Fig. 4.1, an input signal which is the controller output is represented with the ZOH approach. As can be understood from the figure, the input signal or controlled signal has a time delay amount of $h/2$. h implies sampling time of the signal. If sampling time is not chosen

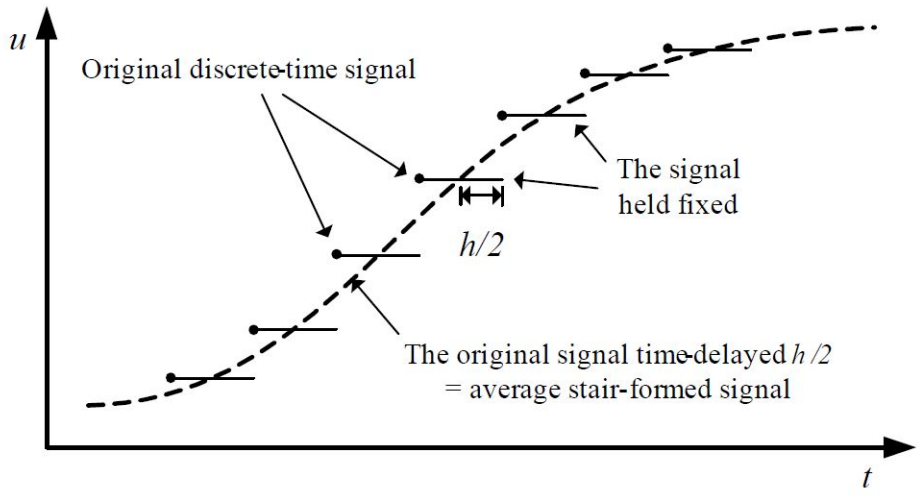

Figure 4.1: Time delay of the input signal [26].

very well, discrete time system might be unstable or less stable compared to its continuous counterpart. In any case, delays introduce instability to the system, but it can be brought into a tolerable level. As a rule of thumb, half of the delay time is chosen to be smaller than one-tenth of the response time [26]. This relation can be written in (4.10) and (4.11) as follows:

$$\frac{h}{2} \leq \frac{T_r}{10} \tag{4.10}$$

To write more simply,

$$h \leq \frac{T_r}{5} \tag{4.11}$$

Response time can be found via step response of the closed-loop system. It is accepted 63% of the rise time which is the required amount of response from 10% to 90%, 5% to 95% or 0% to 100% according to under-damped, over-damped conditions.

4.3 Model Predictive Control

Model predictive control, shortly MPC, which is also called receding horizon control, finds a control sequence along with finite output and control horizons by solving constrained or unconstrained optimization problem. At each time instant, an optimal control input sequence is calculated and only the value at the first time instant of control horizon sequence is applied to the system. This procedure is repeated in each

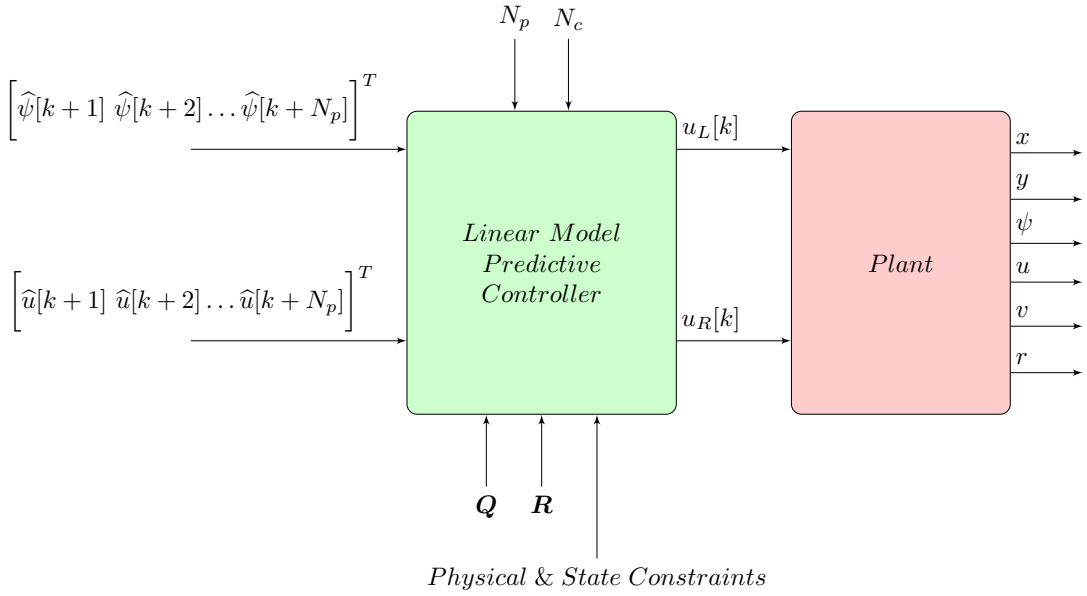

Figure 4.2: Block diagram representation of MPC.

sampling instant within an optimization window to calculate the optimal control input sequence and to minimize errors between references and outputs.

4.3.1 Discrete-time State-Space Model with Embedded Integrator

This study covers the linear model predictive controller, LMPC, which is designed for the discrete-time system with an appropriately chosen sampling time, T_s, whose definition is in 4.2.2. Discrete-time state-space equations are written as follows:

$$\boldsymbol{x}_d[k+1] = \boldsymbol{A}_d\,\boldsymbol{x}_d[k] \;+\; \boldsymbol{B}_d\,\boldsymbol{u}[k] \tag{4.12}$$

$$\boldsymbol{y}[k] = \boldsymbol{C}_d\,\boldsymbol{x}_d[k] \;+\; \boldsymbol{D}_d\,\boldsymbol{u}[k] \tag{4.13}$$

where $\boldsymbol{A}_d$ matrix represents the discrete time state matrix at time k with the same dimension $\boldsymbol{A}(t)$. $\boldsymbol{B}_d$ matrix shows discrete time input matrix at time k with same dimension $\boldsymbol{B}(t)$. Both $\boldsymbol{A}(t)$ and $\boldsymbol{B}(t)$ are previously defined in (4.8). $\boldsymbol{C}_d$ is used for discrete time output matrix which is equal to $\boldsymbol{C}(t)$ defined in (4.9). In this system, input does not directly affect the outputs. Therefore, $\boldsymbol{D}_d = \boldsymbol{0}$ in state space (4.13) will not be written any more. The dimension of the state vector, $\boldsymbol{x}_d[k]$, is same as the number of columns in $\boldsymbol{A}_d$ matrix and dimension of the input vector, $\boldsymbol{u}[k]$, is two due to the two thruster in the system. Proposed discrete-time model predictive controller

finds optimal incremental input movement, $\Delta \boldsymbol{u}[k]$, instead of sequence, $\boldsymbol{u}[k]$. So, integral action on the controller is achieved to eliminate steady-state errors without the necessity of steady-state information about the control, $\boldsymbol{u}[k] = \boldsymbol{u}[k-1] + \Delta \boldsymbol{u}[k]$, and the steady state of the state variable, $\boldsymbol{x}_d[k]$. First of all, the difference of state vectors between $k+1$ and k is calculated using (4.12) as follows:

$$\boldsymbol{x}_d[k+1] - \boldsymbol{x}_d[k] = \boldsymbol{A}_d(\boldsymbol{x}_d[k] - \boldsymbol{x}_d[k-1]) + \boldsymbol{B}_d(\boldsymbol{u}[k] - \boldsymbol{u}[k-1]) \tag{4.14}$$

where the difference of the current and previous states at k+1 is indicated by, $\Delta \boldsymbol{x}_d[k+1] = \boldsymbol{x}_d[k+1] - \boldsymbol{x}_d[k]$, and at time k, $\Delta \boldsymbol{x}_d[k] = \boldsymbol{x}_d[k] - \boldsymbol{x}_d[k-1]$, respectively. Difference of the input vectors are obtained in the same way as $\Delta \boldsymbol{u}[k] = \boldsymbol{u}[k] - \boldsymbol{u}[k-1]$. Then, (4.14) is implicitly rewritten by using above informations as follows:

$$\Delta \boldsymbol{x}_d[k+1] = \boldsymbol{A}_d \Delta \boldsymbol{x}_d[k] + \boldsymbol{B}_d \Delta \boldsymbol{u}[k] \tag{4.15}$$

To be able to write an augmented state space model with embedded integrators, $\Delta \boldsymbol{x}_d[k]$ and $\boldsymbol{y}[k]$ is gathered in one vector that is called augmented state vector as $\boldsymbol{x}[k] = [\Delta \boldsymbol{x}_d[k]^T \;\; \boldsymbol{y}(k)]^T$. Difference between $\boldsymbol{y}(k+1)$ and $\boldsymbol{y}(k)$ is expressed in the following equations.

$$\begin{aligned} \boldsymbol{y}[k+1] - \boldsymbol{y}[k] &= \boldsymbol{C}_d(\boldsymbol{x}_d[k+1] - \boldsymbol{x}_d[k]) = \boldsymbol{C}_d \Delta \boldsymbol{x}_d[k+1] \\ &= \boldsymbol{C}_d \boldsymbol{A}_d \Delta \boldsymbol{x}_d[k] + \boldsymbol{C}_d \boldsymbol{B}_d \Delta \boldsymbol{u}_d[k] \end{aligned} \tag{4.16}$$

So, augmented state space equation is written in compact form by using (4.15) and (4.16) as follows:

$$\underbrace{\begin{bmatrix} \Delta \boldsymbol{x}_d[k+1] \\ \boldsymbol{y}[k] \end{bmatrix}}_{\boldsymbol{x}[k+1]} = \underbrace{\begin{bmatrix} \boldsymbol{A}_d & \boldsymbol{0}_d{}^T \\ \boldsymbol{C}_d \boldsymbol{A}_d & \boldsymbol{I}_{2\times 2} \end{bmatrix}}_{\boldsymbol{A}_k} \underbrace{\begin{bmatrix} \Delta \boldsymbol{x}_d[k] \\ \boldsymbol{y}[k] \end{bmatrix}}_{\boldsymbol{x}[k]} + \underbrace{\begin{bmatrix} \boldsymbol{B}_d \\ \boldsymbol{C}_d \boldsymbol{B}_d \end{bmatrix}}_{\boldsymbol{B}_k} \Delta \boldsymbol{u}[k]$$

$$\boldsymbol{y}[k] = \underbrace{[\boldsymbol{0}_d \;\; \boldsymbol{I}_{2\times 2}]}_{\boldsymbol{C}_k} \begin{bmatrix} \Delta \boldsymbol{x}_d[k] \\ \boldsymbol{y}[k] \end{bmatrix} \tag{4.17}$$

where $\boldsymbol{0}_d \in \mathbb{R}^{2x12}$ is zero matrix. Augmented state space model which used for LMPC design is finally denoted by the triplet $(\boldsymbol{A}_k, \boldsymbol{B}_k, \boldsymbol{C}_k)$. The characteristic equation of $\boldsymbol{A}_k$ matrix is represented as follows:

$$\rho(\lambda) = \det(\lambda \boldsymbol{I} - \boldsymbol{A}_k) = \begin{bmatrix} \lambda \boldsymbol{I} - \boldsymbol{A}_d & \boldsymbol{0}_d{}^T \\ -\boldsymbol{C}_d \boldsymbol{A}_d & (\lambda - 1)\boldsymbol{I}_{2\times 2} \end{bmatrix} \tag{4.18}$$

where $\rho(\lambda) = (\lambda - 1)^2 \det(\lambda \boldsymbol{I} - \boldsymbol{A}_d) = 0$. As can be understood from this relation at least two eigenvalues are on the unit circle. It means that the augmented model includes two embedded integrators which provide an integral action for MPC.

MPC calculates future control signals for finite control horizon, N_c, and uses predicted output signals whose horizon is equal to the length of the optimization window, N_p. These two tuning parameters must be successfully adjusted to be able to attain adequate performance from the proposed model predictive controller. The prediction horizon N_p is chosen to be greater than or equal to control horizon, N_c. The existence of embedded integrator in the controller design may cause unstable system responses when N_p becomes too large. Furthermore, the stability of the system could not be satisfied with small N_p and N_c values [53]. Optimal incremental control horizon, (4.19), is sequentially written from sampling instant k to $k + N_c - 1$ by

$$\{\Delta \boldsymbol{u}[k], \Delta \boldsymbol{u}[k+1], \ldots, \Delta \boldsymbol{u}[k+N_c-1]\} \tag{4.19}$$

Using these control sequence and the triplet $(\boldsymbol{A}_k, \boldsymbol{B}_k, \boldsymbol{C}_k)$, future predicted state and output variables are sequentially written as follows,

$$\begin{aligned}
\boldsymbol{x}[k+1] &= \boldsymbol{A}_k \boldsymbol{x}[k] + \boldsymbol{B}_k \Delta \boldsymbol{u}[k] \\
\boldsymbol{x}[k+2] &= \boldsymbol{A}_k \boldsymbol{x}[k+1] + \boldsymbol{B}_k \Delta \boldsymbol{u}[k+1] \\
&= \boldsymbol{A}_k^2 \boldsymbol{x}[k] + \boldsymbol{A}_k \boldsymbol{B}_k \Delta \boldsymbol{u}[k] + \boldsymbol{B}_k \Delta \boldsymbol{u}[k+1] \\
&\vdots \\
\boldsymbol{x}[k+N_p] &= \boldsymbol{A}_k^{N_p} \boldsymbol{x}[k] + \boldsymbol{A}_k^{N_p-1} \boldsymbol{B}_k \Delta \boldsymbol{u}[k] + \boldsymbol{A}_k^{N_p-2} \boldsymbol{B}_k \Delta \boldsymbol{u}[k+1] \\
&\quad + \cdots + \boldsymbol{A}_k^{N_p-N_c} \boldsymbol{B}_k \Delta \boldsymbol{u}[k+N_c-1]
\end{aligned} \tag{4.20}$$

Above state vectors are multiplied with output matrix $\boldsymbol{C}_k$ to determine sequentially future predicted output vectors which are the function of the current state vector, $\boldsymbol{x}[k]$ and control horizon vectors defined in (4.20).

$$\begin{aligned}
\boldsymbol{y}[k+1] &= \boldsymbol{C}_k \boldsymbol{A}_k \boldsymbol{x}[k] + \boldsymbol{C}_k \boldsymbol{B}_k \Delta \boldsymbol{u}[k] \\
\boldsymbol{y}[k+2] &= \boldsymbol{C}_k \boldsymbol{A}_k \boldsymbol{x}[k+1] + \boldsymbol{C}_k \boldsymbol{B}_k \Delta \boldsymbol{u}[k+1] \\
&= \boldsymbol{C}_k \boldsymbol{A}_k^2 \boldsymbol{x}[k] + \boldsymbol{C}_k \boldsymbol{A}_k \boldsymbol{B}_k \Delta \boldsymbol{u}[k] + \boldsymbol{C}_k \boldsymbol{B}_k \Delta \boldsymbol{u}[k+1] \\
&\vdots \\
\boldsymbol{y}[k+N_p] &= \boldsymbol{C}_k \boldsymbol{A}_k^{N_p} \boldsymbol{x}[k] + \boldsymbol{C}_k \boldsymbol{A}_k^{N_p-1} \boldsymbol{B}_k \Delta \boldsymbol{u}[k] + \boldsymbol{C}_k \boldsymbol{A}_k^{N_p-2} \boldsymbol{B}_k \Delta \boldsymbol{u}[k+1]
\end{aligned}$$

$$+ \cdots + \boldsymbol{C}_k \boldsymbol{A}_k^{N_p - N_c} \boldsymbol{B}_k \Delta \boldsymbol{u}[k + N_c - 1] \tag{4.21}$$

These equations are written in matrix form by defining the following future predicted output vector and a future control vector, respectively.

$$\begin{aligned} \boldsymbol{Y} &= [\boldsymbol{y}^T[k+1] \;\; \boldsymbol{y}^T[k+2] \;\; \ldots \;\; \boldsymbol{y}^T[k+N_p]]^T \\ \Delta \boldsymbol{U} &= [\Delta \boldsymbol{u}^T[k] \;\; \Delta \boldsymbol{u}^T[k+1] \;\; \ldots \;\; \Delta \boldsymbol{u}^T[k+N_c-1]]^T \end{aligned} \tag{4.22}$$

and,

$$\boldsymbol{Y} = \boldsymbol{F}\boldsymbol{x}[k] + \boldsymbol{\Phi} \Delta \boldsymbol{U} \tag{4.23}$$

where,

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{C}_k \boldsymbol{A}_k \\ \boldsymbol{C}_k \boldsymbol{A}_k^2 \\ \vdots \\ \boldsymbol{C}_k \boldsymbol{A}_k^{N_p} \end{bmatrix},$$

$$\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{C}_k \boldsymbol{B}_k & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \boldsymbol{C}_k \boldsymbol{A}_k \boldsymbol{B}_k & \boldsymbol{C}_k \boldsymbol{B}_k & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \vdots & & & & \\ \boldsymbol{C}_k \boldsymbol{A}_k^{N_p-1} \boldsymbol{B}_k & \boldsymbol{C}_k \boldsymbol{A}_k^{N_p-2} \boldsymbol{B}_k & \boldsymbol{C}_k \boldsymbol{A}_k^{N_p-3} \boldsymbol{B}_k & \ldots & \boldsymbol{C}_k \boldsymbol{A}_k^{N_p-N_c} \boldsymbol{B}_k \end{bmatrix} \tag{4.24}$$

4.3.2 Optimization Problem

Constrained quadratic optimization problem is solved along specified prediction (optimization window) and control horizons. Errors between predicted and reference outputs over the optimization window are aimed to be optimized in order to find minimum control effort. The cost function of the MPC is algebraically written as follows:

$$\begin{aligned} \min_{\Delta \boldsymbol{U}} \left\{ J(\Delta \boldsymbol{U}) \right\} = &\sum_{i=1}^{N_p} (\hat{\boldsymbol{y}}_{ref}[k+i] - \boldsymbol{y}[k+i])^T \boldsymbol{Q} \, (\hat{\boldsymbol{y}}_{ref}[k+i] - \boldsymbol{y}[k+i]) \\ &+ \sum_{i=0}^{N_c-1} \Delta \boldsymbol{u}[k+i]^T \boldsymbol{R} \, \Delta \boldsymbol{u}[k+i] \end{aligned} \tag{4.25}$$

The above equation is a function of the current state control input sequence $\Delta \boldsymbol{U}$. First summation term composes of summed quadratic convex functions which reflect error between references and predicted output vectors. Symmetric, diagonal and positive

semi-definite $\boldsymbol{Q}$, $\boldsymbol{Q} \geq \mathbf{0}$, is the weight matrix for track errors minimization and satisfies convexity property of the cost function. The second term which is a function of future incremental control trajectory is added to cost function in order to reduce control effort (applied amount energy) of the system as much as possible. Diagonal, symmetric positive definite $\boldsymbol{R} = r_w \boldsymbol{I}_{2\times 2}$, $\boldsymbol{R} > \mathbf{0}$, is the other weight matrix for minimizing incremental control effort with its diagonal element, r_w. The objective function can be represented in a more compact matrix form using (4.22), (4.24) and (4.25) as follows:

$$J = J(\Delta \boldsymbol{U}) = (\hat{\boldsymbol{Y}} - \boldsymbol{Y})^T \bar{\boldsymbol{Q}} (\hat{\boldsymbol{Y}} - \boldsymbol{Y}) + \Delta \boldsymbol{U}^T \bar{\boldsymbol{R}} \Delta \boldsymbol{U} \tag{4.26}$$

where the reference output vector within its prediction trajectory is defined as

$$\hat{\boldsymbol{Y}}^T = [\ \hat{\boldsymbol{y}}_{ref}^T[k+1]\ \ \hat{\boldsymbol{y}}_{ref}^T[k+2]\ \ \ldots\ \ \hat{\boldsymbol{y}}_{ref}^T[k+N_p]\ ]^T \tag{4.27}$$

Since vector $\hat{\boldsymbol{y}}_{ref}[k+i] \in \mathbb{R}^2$ where $i \in [0, \ldots, N_p]$. $\bar{\boldsymbol{Q}} \in \mathbb{R}_{\geq 0}^{2N_p \times 2N_p}$ and $\bar{\boldsymbol{R}} \in \mathbb{R}_{>0}^{2N_c \times 2N_c}$ is diagonal, positive, semidefinite and definite matrices, $\bar{\boldsymbol{Q}} \geq \mathbf{0}$ and $\bar{\boldsymbol{R}} > \mathbf{0}$, respectively, in following form:

$$\bar{\boldsymbol{Q}} = \begin{bmatrix} \boldsymbol{Q} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{Q} \end{bmatrix},\ \bar{\boldsymbol{R}} = \begin{bmatrix} \boldsymbol{R} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{R} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{R} \end{bmatrix} \tag{4.28}$$

By using (4.23) , explicit form of (4.26) is expressed as

$$\begin{aligned} J(\Delta \boldsymbol{U}) &= (\hat{\boldsymbol{Y}} - (\boldsymbol{F}\boldsymbol{x}[k] + \boldsymbol{\Phi}\Delta \boldsymbol{U}))^T \bar{\boldsymbol{Q}} (\hat{\boldsymbol{Y}} - (\boldsymbol{F}\boldsymbol{x}[k] + \boldsymbol{\Phi}\Delta \boldsymbol{U})) + \Delta \boldsymbol{U}^T \bar{\boldsymbol{R}} \Delta \boldsymbol{U} \\ &= (\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k])^T \bar{\boldsymbol{Q}} (\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k]) - 2\Delta \boldsymbol{U}^T \boldsymbol{\Phi}^T \bar{\boldsymbol{Q}} (\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k]) \\ &\quad + \Delta \boldsymbol{U}^T \boldsymbol{\Phi}^T \bar{\boldsymbol{Q}} \boldsymbol{\Phi} \Delta \boldsymbol{U} + \Delta \boldsymbol{U}^T \bar{\boldsymbol{R}} \Delta \boldsymbol{U} \\ &= \Delta \boldsymbol{U}^T (\boldsymbol{\Phi}^T \bar{\boldsymbol{Q}} \boldsymbol{\Phi} + \bar{\boldsymbol{R}}) \Delta \boldsymbol{U} - 2\Delta \boldsymbol{U}^T \boldsymbol{\Phi}^T \bar{\boldsymbol{Q}} (\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k]) \\ &\quad + (\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k])^T \bar{\boldsymbol{Q}} (\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k]) \end{aligned} \tag{4.29}$$

$\Delta \boldsymbol{U}$ is utilized as a vector which is minimized using quadratic programming techniques in the objective function (4.29). To be able to write this function more compact, following terms simply denoted as $2(\boldsymbol{\Phi}^T \bar{\boldsymbol{Q}} \boldsymbol{\Phi} + \bar{\boldsymbol{R}}) = \boldsymbol{E}$ and $-2\boldsymbol{\Phi}^T \bar{\boldsymbol{Q}} (\hat{\boldsymbol{Y}} -$

$\boldsymbol{F}\boldsymbol{x}[k]) = \boldsymbol{d}$. Also note that $\boldsymbol{E}$ is symmetric and positive semi-definite matrix since $\bar{\boldsymbol{Q}}$, positive semi-definite matrix, and $\bar{\boldsymbol{R}}$, positive definite matrix, are defined as in (4.28). Remaining term, $(\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k])^T \bar{\boldsymbol{Q}}(\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k])$, has not any effect on the optimization problem and calculation of this term is not necessary.

4.3.2.1 Constraints of the Optimization Problem

Constraints play an essential role in model predictive controllers because of physical and operational limits. When convex quadratic cost function (4.29) is solved regardless of constraints, a global optimum solution is found. It may not satisfy physical input constraints which are limited to certain minimum and maximum values. If the result of the unconstrained optimization problem is saturated rather than incorporating constraints into it, the control signal may deteriorate and lead to overshoot in the physical system.

Constraints can be classified into there major types [53]. First two types which are mostly called as hard constraints are related to control variables, $\boldsymbol{u}[k]$, and their incremental variations, $\Delta\boldsymbol{u}[k]$. Another constraint type, soft constraint, which compose of output variable $\boldsymbol{y}[k]$ or state variable $\boldsymbol{x}[k]$. Hard constraints are only added as a subject term to the optimization problem in this study due to the fact that soft constraints take part as minimized terms in the cost function of the optimization problem.

- **Inequality Constraints**

At time k, following two equations represent the rate of change limits and amplitude limits of control signals, respectively. Since control signal is in vectorial form, each component of the control signals is separately represented.

$$\left.\begin{aligned} \Delta u_L^{min} \leq \Delta u_L[k] \leq \Delta u_L^{max} \\ \Delta u_R^{min} \leq \Delta u_R[k] \leq \Delta u_R^{max} \end{aligned}\right\} \Delta\boldsymbol{u}^{min} \leq \Delta\boldsymbol{u}[k] \leq \Delta\boldsymbol{u}^{max} \tag{4.30}$$

and,

$$\left.\begin{aligned} u_L^{min} \leq u_L[k] \leq u_L^{max} \\ u_R^{min} \leq u_R[k] \leq u_R^{max} \end{aligned}\right\} \boldsymbol{u}^{min} \leq \boldsymbol{u}[k] \leq \boldsymbol{u}^{max} \tag{4.31}$$

where $\Delta \boldsymbol{u}^{min} = [\Delta u_L^{min}\ \Delta u_R^{min}]^T$, lower limits are taken as $\Delta u_L^{min} = \Delta u_R^{min}$. $\Delta \boldsymbol{u}^{max} = [\Delta u_L^{max}\ \Delta u_R^{max}]^T$, upper limits are taken to be $\Delta u_L^{max} = \Delta u_R^{max}$. Similarly, $\boldsymbol{u}^{min} = [u_L^{min}\ u_R^{min}]^T$, lower limits for control amplitude are defined as $u_L^{min} = u_R^{min}$, while upper limit of control amplitude is formulated as $\boldsymbol{u}^{max} = [u_L^{max}\ u_R^{max}]^T$, upper limits are taken as $u_L^{max} = u_R^{max}$. When constraints are considered over control horizon, i.e., $k \in [0, \ldots, N_c - 1]$, following inequality forms are written by the help of (4.30) and (4.31).

$$\left.\begin{aligned} \Delta \boldsymbol{u}^{min} \leq \Delta \boldsymbol{u}[k] \leq \Delta \boldsymbol{u}^{max} \\ \Delta \boldsymbol{u}^{min} \leq \Delta \boldsymbol{u}[k+1] \leq \Delta \boldsymbol{u}^{max} \\ \vdots \\ \Delta \boldsymbol{u}^{min} \leq \Delta \boldsymbol{u}[k+N_c-1] \leq \Delta \boldsymbol{u}^{max} \end{aligned}\right\} \Delta \boldsymbol{U}^{min} \leq \Delta \boldsymbol{U} \leq \Delta \boldsymbol{U}^{max} \tag{4.32}$$

and,

$$\left.\begin{aligned} \boldsymbol{u}^{min} \leq \boldsymbol{u}[k] \leq \boldsymbol{u}^{max} \\ \boldsymbol{u}^{min} \leq \boldsymbol{u}[k+1] \leq \boldsymbol{u}^{max} \\ \vdots \\ \boldsymbol{u}^{min} \leq \boldsymbol{u}[k+N_c-1] \leq \boldsymbol{u}^{max} \end{aligned}\right\} \boldsymbol{U}^{min} \leq \boldsymbol{U} \leq \boldsymbol{U}^{max} \tag{4.33}$$

Two-sided inequality constraint will be separated into two inequalities to solve the optimization problem by using matrix property, for example, constraint (4.32), $\Delta \boldsymbol{U}^{min} \leq \Delta \boldsymbol{U} \leq \Delta \boldsymbol{U}^{max}$, is divided into two parts as follows

$$\left.\begin{aligned} -\Delta \boldsymbol{U} \leq -\Delta \boldsymbol{U}^{min} \\ \Delta \boldsymbol{U} \leq \Delta \boldsymbol{U}^{max} \end{aligned}\right\} \Leftrightarrow \begin{bmatrix} -\boldsymbol{I} \\ \boldsymbol{I} \end{bmatrix} \Delta \boldsymbol{U} \leq \begin{bmatrix} -\Delta \boldsymbol{U}^{min} \\ \Delta \boldsymbol{U}^{max} \end{bmatrix} \tag{4.34}$$

Constraints are especially considered for all future times, and all input constraints can be written in the form of $\Delta \boldsymbol{u}[k+i]$, $i \in [0, 1, \ldots, N_c - 1]$. So amplitude of the future control inputs may be written in the following form:

$$\begin{bmatrix} \boldsymbol{u}[k] \\ \boldsymbol{u}[k+1] \\ \vdots \\ \boldsymbol{u}[k+N_c-1] \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{I} \\ \vdots \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{u}[k-1] \ + \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{I} & \boldsymbol{I} & \cdots & \boldsymbol{0} \\ \vdots & & \ddots & \\ \boldsymbol{I} & \boldsymbol{I} & \cdots & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{u}[k] \\ \Delta \boldsymbol{u}[k+1] \\ \vdots \\ \Delta \boldsymbol{u}[k+N_c-1] \end{bmatrix} \tag{4.35}$$

where $\boldsymbol{u}[k-1]$ is $(k-1)^{th}$ sampling time instant of control signal whose value is already known. If k is equal to zero and initially zero input is applied to the system,

then its value becomes zero, i.e., $\boldsymbol{u}[k-1] = 0$. Otherwise, it can be any value between its lower and upper limits. By taking into account the incremental variation on the control signal as the optimization variable, the general form of the hard inequality constraints will be represented by

$$\boldsymbol{M}\Delta\boldsymbol{U} \leq \boldsymbol{\gamma}. \tag{4.36}$$

- **Equality Constraints**

Equality constraints can be written by using state space equations which are already obtained in (4.17) from time k to $k + N_p$ as

$$\begin{aligned} \boldsymbol{x}[k+i+1] &= \boldsymbol{A}_k\boldsymbol{x}[k+i] + \boldsymbol{B}_k\boldsymbol{u}[k+i] \\ \boldsymbol{y}[k+i] &= \boldsymbol{C}_k\boldsymbol{x}[k+i] \end{aligned} \tag{4.37}$$

where $i \in [0, 1, \ldots, N_p]$ and for $k \geq N_c$ input sequences are taken as $\boldsymbol{u}[k] = \boldsymbol{u}[k+1] = \cdots = \boldsymbol{u}[k+N_p]$. These equality equations are already calculated to determine future state variable and future output variables that take part in (4.29). Because of this feature, equality constraints are not repeatedly solved in the optimization problem. The solution of the mentioned quadratic problem with inequality constraint at the global minimum point satisfies the equality constraints.

4.3.3 Numerical Solution of Quadratic Programming for MPC

The constrained optimization problem with its objective function and inequality constraints whose derivation is done in the previous section can be finally expressed as:

$$\begin{aligned} J(\Delta\boldsymbol{U}) &= \frac{1}{2}\Delta\boldsymbol{U}^T\boldsymbol{E}\Delta\boldsymbol{U} + \Delta\boldsymbol{U}^T\boldsymbol{d} \\ \boldsymbol{M}\Delta\boldsymbol{U} &\leq \boldsymbol{\gamma}. \end{aligned} \tag{4.38}$$

where matrices, $\boldsymbol{E}, \boldsymbol{d}, \boldsymbol{M}$ and vector $\boldsymbol{\gamma}$, are previously defined. Without loss of generality, weight or diagonal terms of MPC matrices are taken as positive that makes $\boldsymbol{E}$ is semidefinite due to $\bar{\boldsymbol{Q}}$ and $\bar{\boldsymbol{R}}$. The objective function, $J(\Delta\boldsymbol{U})$, is minimized via Lagrange multipliers subject to inequality constraints. Adding a Lagrange multiplier

vector to the objective function increases the number of design parameters in the minimization function. Only unknown parameters are entities of the $\Delta \boldsymbol{U}$ vector to appear in the objective function. As it stands now,

$$\boldsymbol{J}(\Delta \boldsymbol{U}, \boldsymbol{\lambda}) = \frac{1}{2}\Delta \boldsymbol{U}^T \boldsymbol{E} \Delta \boldsymbol{U} + \Delta \boldsymbol{U}^T \boldsymbol{d} + \boldsymbol{\lambda}^T (\boldsymbol{M} \Delta U - \boldsymbol{\gamma}.) \tag{4.39}$$

where $\boldsymbol{\lambda}$ is the Lagrange multiplier vector whose size m is equal to the number of inequality constraints in the optimization problem. Since the number of inequality constraints can be larger than the number of minimized control variables, active and inactive constraints may be observed in the problem. If $\boldsymbol{M}_i \Delta U = \boldsymbol{\gamma}_i$ is satisfied, constraint $\boldsymbol{M}_i \Delta U \leq \boldsymbol{\gamma}_i$ is said to be active. Otherwise, when $\boldsymbol{M}_i \Delta U < \boldsymbol{\gamma}_i$ becomes constraint is considered that it is inactive. $\boldsymbol{M}_i$ is the $i-th$ row of inequality constraint matrix, $\boldsymbol{M}$, and $\boldsymbol{\gamma}_i$ is the $i-th$ row of the constraint vector, $\boldsymbol{\gamma}$. The first step is to minimize the objective function with a Lagrange multiplier is to take partial derivatives with respect to the vectors $\Delta \boldsymbol{U}$ and $\boldsymbol{\lambda}$.

$$\frac{\partial \boldsymbol{J}}{\partial \Delta \boldsymbol{U}} = \boldsymbol{E} \Delta \boldsymbol{U} + \boldsymbol{d} + \boldsymbol{M}^T \boldsymbol{\lambda} = \boldsymbol{0} \tag{4.40}$$

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{\lambda}} = \boldsymbol{M} \Delta \boldsymbol{U} - \boldsymbol{\gamma}. \tag{4.41}$$

If all the constraints had satisfied their conditions, the minimization process would have been straightforward. The optimal values of $\Delta \boldsymbol{U}$ and $\boldsymbol{\lambda}$ vectors are obtained from (4.40) and (4.41) as follows, respectively,

$$\boldsymbol{\lambda} = -(\boldsymbol{M} \boldsymbol{E}^{-1} \boldsymbol{M}^T)^{-1} (\boldsymbol{\gamma} + \boldsymbol{M} \boldsymbol{E}^{-1} \boldsymbol{d}) \tag{4.42}$$

$$\begin{aligned} \Delta \boldsymbol{U}^{optimal} &= -\boldsymbol{E}^{-1} (\boldsymbol{M}^T \boldsymbol{\lambda} + \boldsymbol{d}) \\ &= -\boldsymbol{E}^{-1} \boldsymbol{d} - \boldsymbol{E}^{-1} \boldsymbol{M}^T \boldsymbol{\lambda} = \Delta \boldsymbol{U}^{global} - \boldsymbol{E}^{-1} \boldsymbol{M}^T \boldsymbol{\lambda} \end{aligned} \tag{4.43}$$

where $\Delta \boldsymbol{U}^{global} = -\boldsymbol{E}^{-1} \boldsymbol{d}$ is the global solution of the (4.29) without any constraint. The correction term, $-\boldsymbol{E}^{-1} \boldsymbol{M}^T \boldsymbol{\lambda}$, comes from inequality relations. Unfortunately, this solution is not valid because of the fact that constraints include two-sided inequality in the proposed optimization problem. That's why only one side of these constraints which are explained in detail at 4.3.2.1 will satisfy the given relation. To handle this problem, Kuhn-Tucker conditions are simply determined with regard to active and inactive constraints in terms of Lagrange multiplier. Four necessary Kuhn-Tucker conditions are formulated by

$$\boldsymbol{E} \Delta \boldsymbol{U} + \boldsymbol{d} + \boldsymbol{M}^T \boldsymbol{\lambda} = 0$$

$$
\begin{aligned}
\boldsymbol{M}\Delta\boldsymbol{U} - \boldsymbol{\gamma} &\leq 0 \\
\boldsymbol{\lambda}^T(\boldsymbol{M}\Delta\boldsymbol{U} - \boldsymbol{\gamma}) &= 0 \\
\boldsymbol{\lambda} &\geq 0,
\end{aligned} \tag{4.44}
$$

When the optimization problem includes inequality constraints, they must be explained in term of a set of active constraints [53]. Their index number is the element of S_{act} and necessary condition are rewritten by using this index number as follows:

$$
\begin{aligned}
\boldsymbol{E}\Delta\boldsymbol{U} + \boldsymbol{d} + \sum_{i \in S_{act}} \boldsymbol{\lambda}_i \boldsymbol{M}_i^T &= 0 \\
\boldsymbol{M}_i\Delta\boldsymbol{U} - \boldsymbol{\gamma}_i &= 0, \quad i \in S_{act} \\
\boldsymbol{M}_i\Delta\boldsymbol{U} - \boldsymbol{\gamma}_i &< 0, \quad i \notin S_{act} \\
\boldsymbol{\lambda}_i &\geq 0, \quad i \in S_{act} \\
\boldsymbol{\lambda}_i &= 0, \quad i \notin S_{act}.
\end{aligned} \tag{4.45}
$$

The optimal solution of the MPC controller is found by using active constraint which means that equation satisfies $\boldsymbol{M}_i\Delta\boldsymbol{U} - \boldsymbol{\gamma}_i = 0$. $\boldsymbol{M}_i$ and $\boldsymbol{\gamma}_i$ are the i-th row of compatible inequality constraint matrix and vector, respectively. When these constraints become equality constraint, corresponding Lagrange multiplier, $\boldsymbol{\lambda}_i$ is taken as a non-negative number, i.e., $\boldsymbol{\lambda}_i \geq 0$. Otherwise, $\boldsymbol{M}_i\Delta\boldsymbol{U} - \boldsymbol{\gamma}_i < 0$ is inactive constraint with its Lagrange multiplier, $\boldsymbol{\lambda}_i = 0$. By taking these relations into consideration, (4.42) and (4.43) are rewritten with active inequality matrix, $\boldsymbol{M}_{act}$, and vector, $\boldsymbol{\gamma}_{act}$,

$$
\boldsymbol{\lambda}_{act} = -(\boldsymbol{M}_{act}\boldsymbol{E}^{-1}\boldsymbol{M}_{act}^T)^{-1}(\boldsymbol{\gamma}_{act} + \boldsymbol{M}_{act}\boldsymbol{E}^{-1}\boldsymbol{d}) \tag{4.46}
$$

$$
\begin{aligned}
\Delta\boldsymbol{U}^{optimal} &= -\boldsymbol{E}^{-1}(\boldsymbol{M}_{act}^T\boldsymbol{\lambda}_{act} + \boldsymbol{d}) \\
&= -\boldsymbol{E}^{-1}\boldsymbol{d} - \boldsymbol{E}^{-1}\boldsymbol{M}_{act}^T\boldsymbol{\lambda}_{act} = \Delta\boldsymbol{U}^{global} - \boldsymbol{E}^{-1}\boldsymbol{M}^T\boldsymbol{\lambda}_{act}
\end{aligned} \tag{4.47}
$$

Active Lagrange multipliers can be obtained from the $Active\ Set\ Method$ or $Primal$-$Dual\ Method$. $Primal$-$Dual\ Method$ is only implemented to find active inequality constraints due to the fact that model predictive control can include larger tuning parameters.

4.3.3.1 Primal-Dual Method

The convex quadratic optimization problem with the symmetric and positive definite matrix, $\boldsymbol{E}$, defined in (4.38) is written in the Lagrangian dual problem is to maximize, $\theta(\boldsymbol{\lambda})$ over $\boldsymbol{\lambda} \geq 0$, where [6]

$$\theta(\boldsymbol{\lambda}) = \inf_{\boldsymbol{\lambda} \geq 0} \{\frac{1}{2}\Delta \boldsymbol{U}^T \boldsymbol{E} \Delta \boldsymbol{U} + \Delta \boldsymbol{U}^T \boldsymbol{d} + \boldsymbol{\lambda}^T (\boldsymbol{M}\Delta U - \boldsymbol{\gamma}) : \Delta \boldsymbol{U} \in \mathbb{R}^{2N_c}\} \quad (4.48)$$

Convexity property of the function $(1/2)\Delta \boldsymbol{U}^T \boldsymbol{E} \Delta \boldsymbol{U} + \Delta \boldsymbol{U}^T \boldsymbol{d} + \boldsymbol{\lambda}^T (\boldsymbol{M}\Delta U - \boldsymbol{\gamma})$ is preserved with a defined $\boldsymbol{\lambda}$ variable, i.e. $\boldsymbol{\lambda} \geq 0$; then necessary and sufficient Kuhn-Tucker condition found by taking the gradient of (4.48) with respect to $\Delta \boldsymbol{U}$ becomes

$$\boldsymbol{E}\Delta \boldsymbol{U} + \boldsymbol{M}^T \boldsymbol{\lambda} + \boldsymbol{d} = 0 \quad (4.49)$$

In the view of the above information, the dual problem is denoted as

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{maximize}} \quad & \frac{1}{2}\Delta \boldsymbol{U}^T \boldsymbol{E} \Delta \boldsymbol{U} + \Delta \boldsymbol{U}^T \boldsymbol{d} + \boldsymbol{\lambda}^T (\boldsymbol{M}\Delta U - \boldsymbol{\gamma}) \\ \text{subject to} \quad & \boldsymbol{E}\Delta \boldsymbol{U} + \boldsymbol{M}^T \boldsymbol{\lambda} = -\boldsymbol{d} \\ & \boldsymbol{\lambda} \geq 0 \end{aligned} \quad (4.50)$$

Positive definitiveness property of $\boldsymbol{E}$, i.e. $\boldsymbol{E}^{-1}$ exists, provides unique solution to 4.49 as follows,

$$\Delta \boldsymbol{U} = -\boldsymbol{E}^{-1}(\boldsymbol{d} + \boldsymbol{M}^T \boldsymbol{\lambda}) \quad (4.51)$$

When a unique solution is substituted in (4.48), maximized optimization problem becomes

$$\begin{aligned} \theta(\boldsymbol{\lambda}) = & -\frac{1}{2}(\boldsymbol{E}^{-1}(\boldsymbol{d} + \boldsymbol{M}^T \boldsymbol{\lambda}))^T \boldsymbol{E}(-\boldsymbol{E}^{-1}(\boldsymbol{d} + \boldsymbol{M}^T \boldsymbol{\lambda})) \\ & - (\boldsymbol{E}^{-1}(\boldsymbol{d} + \boldsymbol{M}^T \boldsymbol{\lambda}))^T \boldsymbol{d} - \boldsymbol{\lambda}^T (\boldsymbol{M}\boldsymbol{E}^{-1}(\boldsymbol{d} + \boldsymbol{M}^T \boldsymbol{\lambda}) - \boldsymbol{\gamma}) \\ = & -\frac{1}{2}\boldsymbol{d}^T \boldsymbol{E}^{-1} \boldsymbol{d} - \frac{1}{2}\boldsymbol{\lambda}^T \boldsymbol{M}\boldsymbol{E}^{-1}\boldsymbol{M}^T \boldsymbol{\lambda} - \boldsymbol{\lambda}^T \boldsymbol{M}\boldsymbol{E}^{-1}\boldsymbol{d} - \boldsymbol{\lambda}\boldsymbol{\gamma} \end{aligned} \quad (4.52)$$

To simply represent the equation, $\boldsymbol{K} \triangleq \boldsymbol{M}\boldsymbol{E}^{-1}\boldsymbol{M}^T$ and $\boldsymbol{v} \triangleq \boldsymbol{\gamma} + \boldsymbol{M}\boldsymbol{E}^{-1}\boldsymbol{d}$ matrices are defined. Now, (4.52) implicitly is written as

$$\theta(\boldsymbol{\lambda}) = -\frac{1}{2}\boldsymbol{\lambda}^T \boldsymbol{K}\boldsymbol{\lambda} - \boldsymbol{\lambda}^T \boldsymbol{v} - \frac{1}{2}\boldsymbol{d}^T \boldsymbol{E}^{-1}\boldsymbol{d} \quad (4.53)$$

Finally, the dual optimization problem which is obtained from the primal problem in simple terms is represented as follows:

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{minimize}} \quad & J(\boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{\lambda}^T \boldsymbol{K} \boldsymbol{\lambda} + \boldsymbol{\lambda}^T \boldsymbol{v} + \frac{1}{2}\boldsymbol{d}^T \boldsymbol{E}^{-1} \boldsymbol{d} \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq 0 \end{aligned} \tag{4.54}$$

$(1/2)\boldsymbol{d}^T \boldsymbol{E}^{-1} \boldsymbol{d}$ term of the above equation has any effect on minimization problem to find $\boldsymbol{\lambda}$ value. So, (4.54) can be simplified by deleting that term as

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{minimize}} \quad & J(\boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{\lambda}^T \boldsymbol{K} \boldsymbol{\lambda} + \boldsymbol{\lambda}^T \boldsymbol{v} \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq 0 \end{aligned} \tag{4.55}$$

The solution of the dual problem can be easily obtained in comparison with the primal problem. Active indexes of $\boldsymbol{\lambda}$ vector are used to find corresponding active inequalities, $\boldsymbol{M}_{act}$ and $\boldsymbol{\gamma}_{act}$ which form $\boldsymbol{M}_{act} \Delta \boldsymbol{U}^{optimal} = \boldsymbol{\gamma}_{act}$.

This problem is solved in two steps. First of all, the global optimal solution is obtained, $\Delta \boldsymbol{U}^{global} = -\boldsymbol{E}^{-1}\boldsymbol{d}$, and then all the constraints are checked whether $\Delta \boldsymbol{U}^{global}$ satisfies inequalities or not. If $\boldsymbol{M} \Delta \boldsymbol{U}^{global} \leq \boldsymbol{\gamma}$, then Hildreth's quadratic algorithm stops. If not, the basis vector $\boldsymbol{e}_i = [0 \quad 0 \quad \dots \quad 1 \quad 0 \quad \dots \quad 0]^T$ is utilized as direction vector to to determine $\boldsymbol{\lambda}_{act}$ value. Activeness of the index Lagrangian multiplier, $\boldsymbol{\lambda}_i$, is understood when it becomes zero or not. When $\boldsymbol{\lambda}_i$ gets zero or negative value, it is said that its corresponding constraint is inactive and $\boldsymbol{\lambda}_{act,i}$ is taken zero. Update procedure of $\boldsymbol{\lambda}_i$ is calculated within predetermined iteration window using $\boldsymbol{K}$ matrix and $\boldsymbol{v}$ vector in (4.55). Update procedure of the Lagrange multiplier is calculated using following two equations.

$$\boldsymbol{\alpha}_i^{k+1} = -\frac{1}{\boldsymbol{K}_{ii}}[\boldsymbol{v}_i + \sum_{j=1}^{i-1} \boldsymbol{K}_{ij} \boldsymbol{\lambda}_j^{k+1} + \sum_{j=i+1}^{n} \boldsymbol{K}_{ij} \boldsymbol{\lambda}_j^{k}] \tag{4.56}$$

$$\boldsymbol{\lambda}_i^{k+1} = max(0, \boldsymbol{\alpha}_i^{k+1}) \tag{4.57}$$

k stands for the iteration index. K_{ij} is the ij^{th} entry of $\boldsymbol{K}$ matrix. j^{th} row of the $\boldsymbol{\lambda}$ vector is denoted by $\boldsymbol{\lambda}_j$ whilst i^{th} row of the $\boldsymbol{v}$ is corresponded $\boldsymbol{v}_i$. n is the row number of the Lagrangian multiplier $\boldsymbol{\lambda}$. The algorithm stops when reach the maximum

Algorithm 1: Hildreth's quadratic programming algorithm

Input: $\boldsymbol{\Phi}$, $\boldsymbol{F}$, $\bar{\boldsymbol{Q}}$, $\bar{\boldsymbol{R}}$, $\boldsymbol{M}$, $\boldsymbol{\gamma}$, N, ϵ

Output: $\Delta \boldsymbol{U}^{optimal}$

Data: $\boldsymbol{E}$, $\boldsymbol{d}$, $\boldsymbol{K}$, $\boldsymbol{v}$, $\boldsymbol{\lambda}$, $\boldsymbol{\lambda}_{prev}$, α, l, n

$Compute\ \boldsymbol{E} \leftarrow 2\boldsymbol{\Phi}^T\bar{\boldsymbol{Q}}\boldsymbol{\Phi} + \bar{\boldsymbol{R}},\ \boldsymbol{d} \leftarrow -2\boldsymbol{\Phi}^T\bar{\boldsymbol{Q}}(\hat{\boldsymbol{Y}} - \boldsymbol{F}\boldsymbol{x}[k])$

$Compute\ \Delta \boldsymbol{U}^{global} \leftarrow -\boldsymbol{E}^{-1}\boldsymbol{d}$

$l = 0$

for $i = 1 \rightarrow n$ **do**

  if $(\boldsymbol{M}_{i,*}\Delta \boldsymbol{U}^{global} > \boldsymbol{\gamma}_i)$ **then**

    $l \leftarrow l + 1$

if $l = 0$ **then**

  $\Delta \boldsymbol{U}^{optimal} \leftarrow \Delta \boldsymbol{U}^{global}$

  return $\Delta \boldsymbol{U}^{optimal}$

$Compute\ \boldsymbol{K} \leftarrow \boldsymbol{M}\boldsymbol{E}^{-1}\boldsymbol{M}^T,\ \boldsymbol{v} \leftarrow \boldsymbol{\gamma} + \boldsymbol{M}\boldsymbol{E}^{-1}\boldsymbol{d}$

$\boldsymbol{\lambda} \leftarrow \boldsymbol{0}$

for $i = 1 \rightarrow N$ **do**

  $\boldsymbol{\lambda}_{prev} \leftarrow \boldsymbol{\lambda}$

  for $i = 1 \rightarrow n$ **do**

    $Compute\ \alpha \leftarrow \dfrac{1}{\boldsymbol{K}_{i,i}}(\boldsymbol{K}_{i,*}\boldsymbol{\lambda} - \boldsymbol{K}_{i,i}\boldsymbol{\lambda}_i + \boldsymbol{v}_i)$

    $Compute\ \boldsymbol{\lambda}_i \leftarrow max(0, \alpha)$

    if $\left((\boldsymbol{\lambda} - \boldsymbol{\lambda}_{prev})^T(\boldsymbol{\lambda} - \boldsymbol{\lambda}_{prev}) < \epsilon\right)$ **then**

      break

$Compute\ \Delta \boldsymbol{U}^{optimal} \leftarrow -\boldsymbol{E}^{-1}\boldsymbol{d} - \boldsymbol{E}^{-1}\boldsymbol{M}\boldsymbol{\lambda}$

return $\Delta \boldsymbol{U}^{optimal}$

/* $\boldsymbol{M}_{i,*}$ and $\boldsymbol{K}_{i,*}$ are the i^{th} row of $\boldsymbol{M}$ and $\boldsymbol{K}$ matricies, respectively */

/* $\boldsymbol{\gamma}_i$ and $\boldsymbol{\lambda}_i$ are the i^{th} element of $\boldsymbol{\gamma}$ and $\boldsymbol{\lambda}$ vectors, respectively */

/* N represents the maximum iteration number */

/* n indicates the size of $\boldsymbol{\lambda}$ vector */

iteration number or change in the Lagrangian multiplier are smaller than termination value, ϵ, e.g. $(\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^{k-1})^T - (\boldsymbol{\lambda}^k - \boldsymbol{\lambda}^{k-1}) < \epsilon$. Thus, it is guaranteed that $\boldsymbol{\lambda}_{act}$ vector includes either zero or positive values. Pseudo code of Hildreth's quadratic programming can be seen in Algorithm 1 with defined matrices and vectors above.

4.3.4 MPC Parameters Tuning

Four tuning parameters, N_p (output horizon), N_c (control horizon), $\boldsymbol{Q}$ and $\boldsymbol{R}$ (weight matrices on errors and control effort, respectively), are adjusted to satisfy robustness and optimality of the controller. Increasing N_p to a reasonable value improves the performance of the MPC. However, the selection of N_p plays a critical role due to the embedded integrators in the state matrix. If it is selected quite small, the robustness of the controller becomes poor. When N_p is large, stability problem on the system can be observed as indicated in 4.3.1. Increasing N_c value too much causes deterioration on the control signals and spreads control effort to a larger horizon. Besides, short horizon values of N_c may enlarge the initial control effort. Positive semi-definite diagonal $\boldsymbol{Q}$ matrix is defined as the weight matrix to adjust the effect of the reference tracking based errors to cost function. Its first diagonal element is corresponding to surge speed error while other is the weight term of the yaw angle error. When manoeuvring without settling a path is performed, the diagonal entity of the $\boldsymbol{Q}$ related to surge speed can be increased. To be able to make collision-free docking, manoeuvring plays an important role at the border or edge of the parking region and slot. The diagonal element of the $\boldsymbol{Q}$ which corresponds to yaw angle should be increased to realize better path following. Positive definite diagonal $\boldsymbol{R}$ matrix adjusts the incremental control effort along the input horizon. Diagonal entities of $\boldsymbol{R}$ are set equal to each other hence thrusters of the vehicle is assumed to be identical.

Parameters of the MPC are tuned along an s-shaped path using LOS guidance. $15 \leq N_p \leq 40$, $2 \leq N_c \leq 15$, $0.001 \leq R_{11} = R_{22} \leq 5$, and $1 \leq Q_{11}, Q_{22} \leq 200$ are specified as intervals to tune parameters of the MPC. Optimized path following performance is achieved by $N_p = 25$, $N_c = 10$, $R_{11} = R_{22} = 0.01$, $Q_{11} = 50$ and $Q_{22} = 100$ values.

Some simulation results of MPC with LOS guidance rule are plotted for the optimized

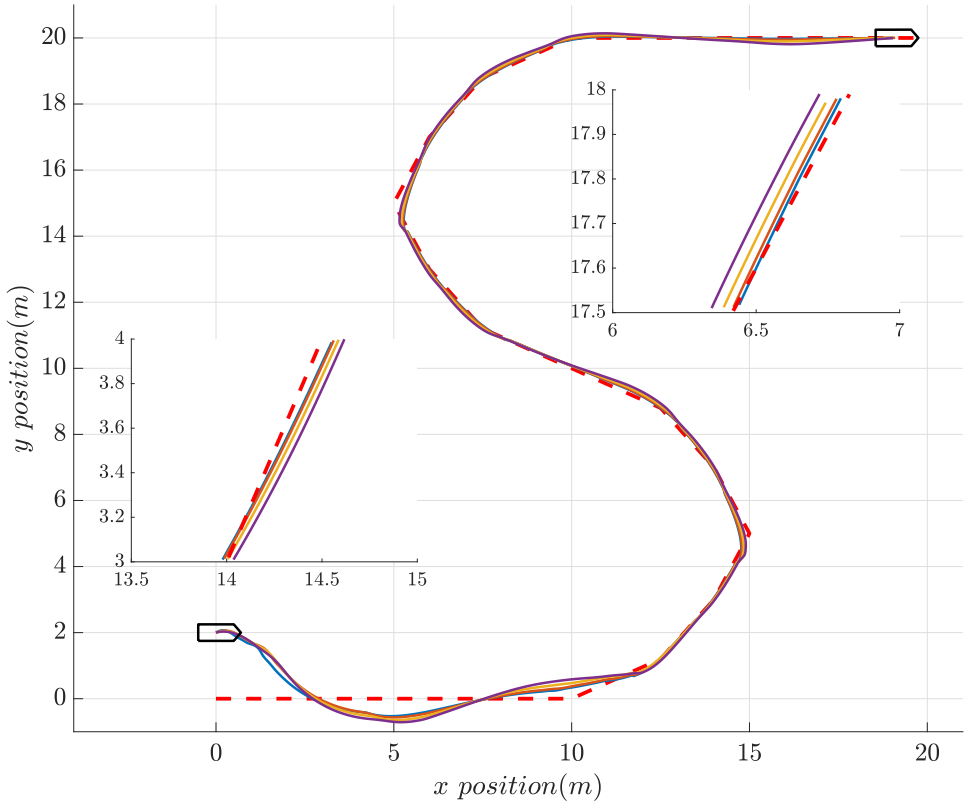

Figure 4.3: Results of the MPC on the s-shape path with different optimized parameter values.

parameters and three parameter sets in Fig. 4.3. Ground truth indicates the shortest path that is represented with a black colour and dash line. Table 4.1 includes simulation parameters and their average cross-track errors. Optimized MPC parameters are in the first row. Other parameters are selected as the variation of the optimized parameters.

Table 4.1: Controller parameters used in Fig. 4.3.

Colour	N_p	N_c	Q_{11}	Q_{22}	$R_{11} = R_{22}$	Avg. Err.(m)
	25	10	50	100	0.01	0.1365
	30	10	50	100	0.01	0.1513
	25	5	50	100	0.01	0.1627
	25	10	50	100	5	0.1868

4.4 Cascaded PID Controller

PID is one of the most used controller technique that can be readily employed in complex control systems. Path tracking performance of the surface craft whose dynamic state equations are coupled and nonlinear can be simply achieved by using PID controllers. PID control is carried out by utilizing and adjusting proportional, K_{p_1}, integral, K_{i_1}, and derivative, K_{d_1}, gains whose notation is represented in Fig. 4.4. These gains take on the task to reduce error or differences between reference signals, $\hat{\boldsymbol{y}}_1[k]$, and output (measured) signal, $\boldsymbol{y}_1[k]$, by considering performance criterion in the controller design. There are various objective functions to minimize track errors at time instant k, $\big\{\boldsymbol{e}_1[k] \triangleq \hat{\boldsymbol{y}}_1[k] - \boldsymbol{y}_1[k]$, $\boldsymbol{e}_2[k] \triangleq \hat{\boldsymbol{y}}_2[k] - \boldsymbol{y}_2[k]$, ..., $\boldsymbol{e}_n[k] \triangleq \hat{\boldsymbol{y}}_n[k] - \boldsymbol{y}_n[k]\big\}$. Integral of absolute errors, **IAE**, is used as a cost function for parameter tuning the PID controller within the scope of this study. **IAE** is formulated as follows:

$$\textbf{IAE} = \big|\boldsymbol{e}_1[k]\big| + \big|\boldsymbol{e}_2[k]\big| + \ ... \ + \big|\boldsymbol{e}_n[k]\big| \tag{4.58}$$

Error minimization is achieved by taking tuning parameters as variables in an optimization window for each PID controller. As mentioned in the introduction section, 4.1, optimization window will be chosen as "S" shape curvature path for PID controllers. After finding proper tuning parameters by using optimization, adjusted values of the PID controller in discrete time is applied to find control effort. The output of a PID controller, $\boldsymbol{U}_1$, which is the control input variable for a system, is formulated as:

$$\boldsymbol{U}_1[k] = K_{p_1} e_1[k] + K_{i_1} T_s \sum_{i=0}^{k} e_1[i] + K_{d_1} \frac{e_1[k] - e_1[k-1]}{T_s} \tag{4.59}$$

where $\boldsymbol{U}_1$ is the output of the first PID controller, and T_s is the sampling time of a discrete system. As it can be understood from (4.59), K_{p_1} is directly multiplied with the error value. In the second term, errors which are from time instant 0 to k are summed and multiplied with $K_{i_1} T_s$ to minimize steady-state errors. It integrates the errors from the beginning until error value reaches to zero. The last term can predict the future behaviour of error by taking the difference between current and previous sampling time instant while other terms are lack of future prediction. It produces K_{d_1}/T_s times rate of change error in sampling time k. The derivative term sometimes

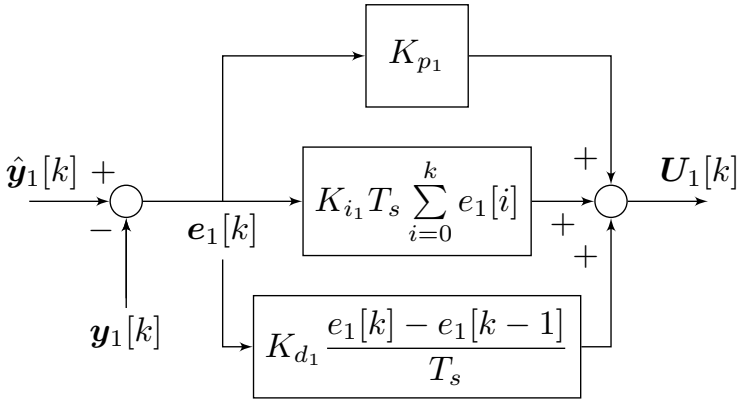

Figure 4.4: PID controller in discrete time.

may increase system response more than expected, so settling time becomes too big and may cause harmful oscillations.

Yaw angle and surge speed errors are minimized by using the linearized vehicle dynamics in the previous controller, MPC in section 4.3. However, minimizing these error by using PID is not sufficient for rapid movements and manoeuvre. When the robust and stable controller is desired, a cascaded PID control loop can be proposed

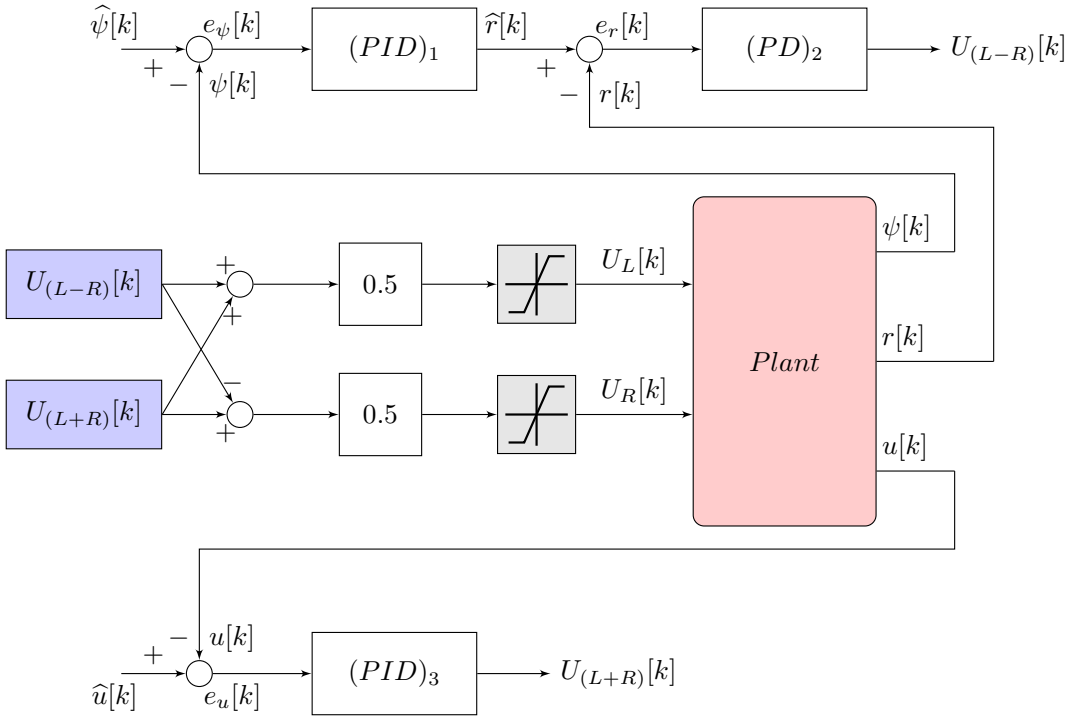

Figure 4.5: Representation of the cascaded PID controller.

for yaw rate error minimization. Cascaded PID becomes useful if there are significant differences in dynamics between the controlled state and the process variables. So control effort is calculated from an intermediate measured signal whose dynamics faster than control signal [49]. Block diagram of proposed cascaded PID controller can be shown in Fig. 4.5. As can be understood from Fig. 4.5, the upper part which is responsible for tracking of yaw angle is composed of two nested control loops. Outer loop controller, sometimes also called primary loop, minimizes yaw angle error, $e_\psi[k]$, between reference yaw angle, $\hat{\psi}[k]$, and yaw angle, $\psi[k]$, at sampling time instant k. The output of the $(PID)_1$ controller is obtained as reference yaw rate, $\widehat{r}[k]$, by using (4.59) and Fig. 4.4 as follows:

$$\widehat{r}[k] = K_{p_1} e_\psi[k] + K_{i_1} T_s \sum_{i=0}^{k} e_\psi[i] + K_{d_1} \frac{e_\psi[k] - e_\psi[k-1]}{T_s} \tag{4.60}$$

(4.60) determines the output of the outer loop controllers as a reference for inner or also called the secondary feedback control loop. It should be taken into consideration that choice of internal measured variable, $r[k]$, must be close relation with the external measured signal, $\psi[k]$ to make tight or fast responsive feedback loop [49]. As it is known from (2.10), $\dot{\boldsymbol{\Theta}}_{nb} = \boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb})\boldsymbol{w}^b_{b|n}$, the derivative of yaw angle is directly related to yaw angle rate. It should also be noted that the secondary loop has a much faster response than the primary loop and environmental disturbances directly act in its loop. Its controller, $(PD)_2$, does not include an integral term because of the fact that it causes overshoot response in the primary loop to eliminate steady-state error. It can be interpreted that secondary loop integrator behaves like a proportional gain in the primary loop. This overshoot can be reduced or even eliminated by using some techniques such as carefully handled integrator windup with anti-windup strategies [52] which is out of scope in this study. The secondary feedback control loop only includes proportional and derivative action in order to find torque difference between left and right thruster , $U_{(L-R)}[k]$, by minimizing $e_r[k] = \widehat{r}[k] - r[k]$.

$$U_{(L-R)}[k] = K_{p_2} e_r[k] + K_{d_2} \frac{e_r[k] - e_r[k-1]}{T_s} \tag{4.61}$$

$(PID)_3$ controller is employed to track reference surge speed, $\widehat{u}[k]$, which is produced by strategic level guidance algorithms. (4.59) is utilized with optimized controller gains K_{p_3}, K_{i_3}, K_{d_3} for proportional, integral and derivative calculations, respectively. Appropriate control effort, produced by $(PID)_3$, which is the summa-

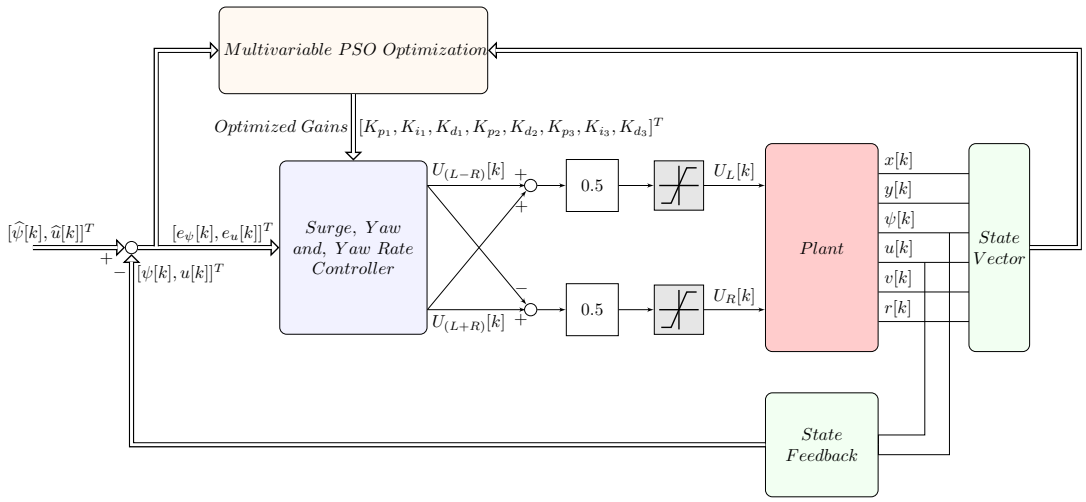

Figure 4.6: Overall cascaded PID controller feedback loop.

tion of left and right thrusters is tried to be found to minimize surge speed error, $e_u[k] = \widehat{u}[k] - u[k]$, as small as possible in.

$$U_{(L+R)}[k] = K_{p_3} e_u[k] + K_{i_3} T_s \sum_{i=0}^{k} e_u[i] + K_{d_3} \frac{e_u[k] - e_u[k-1]}{T_s} \tag{4.62}$$

Thus, the cascaded PID method enhances the stability and robustness of the overall controller design. The applied torques, U_L and U_R, for left and right thrusters may be readily obtained from (4.63), respectively. It should be noted that physical constraints on the thrusters must be taken into consideration by saturating them their maximum and minimum limits before applying these torque values to the system.

$$\begin{aligned} U_L[k] &= \frac{U_{(L+R)}[k] + U_{(L-R)}[k]}{2} \\ U_R[k] &= \frac{U_{(L+R)}[k] - U_{(L-R)}[k]}{2} \end{aligned} \tag{4.63}$$

The cascaded PID control structure finds U_L and U_R torques which are applied to the system in each sampling time. Gains of the controllers, are not be tuned in each time instant. They are found via 'Particle Swarm Optimization, (PSO),' that is one of the evolutionary optimization techniques on the predefined 'S' shaped curvature path. The proposed method to tune cascaded PID variables is explained in the next section.

4.4.1 Particle Swarm Optimization

Gains of the cascaded included PID controllers must be tuned to obtain robust and stable system responses. As it can be seen in Fig. 4.6, gain vector $\boldsymbol{G} = [K_{p_1}\ K_{i_1}\ K_{d_1}\ K_{p_2}\ K_{d_2}\ K_{p_d}\ K_{i_3}\ K_{d_3}]^T$ is taken as design parameters. The cost function of the (4.64) is minimized with variables of $\boldsymbol{G}$ subject to given constraints. Particle swarm optimization technique is employed to find a better $\boldsymbol{G}$ vector within an optimization window which is determined as a curved path with time duration from 0 to k. Gains of the PID controllers are considered as particles of a swarm of the optimization problem. Values that can be taken by particles are restricted as $\boldsymbol{G}_{min} \leq \boldsymbol{G} \leq \boldsymbol{G}_{max}$. Changes of swarm values from one sampling time to another, $\Delta\boldsymbol{G}$ or $\boldsymbol{v}$, is also restricted within a range, i.e. $\Delta\boldsymbol{v}_{min} \leq \Delta\boldsymbol{v} \leq \Delta\boldsymbol{v}_{max}$, to preserve best swarm particles if they are available. At the same time, physical constraints on the left and right thrusters must be kept in their limits, $\boldsymbol{u}_{min} \leq \boldsymbol{u}[k] \leq \boldsymbol{u}_{max}$, during the optimization.

$$\begin{aligned} &CostFunction && J(\boldsymbol{G}) = \left\{ \sum_{i=0}^{k} |e_u[i]| + |e_\psi[i]| + |e_r[i]| \right\} \\ &\text{subject to} && \boldsymbol{G}_{min} \leq \boldsymbol{G} \leq \boldsymbol{G}_{max}, \\ & && \boldsymbol{u}_{min} \leq \boldsymbol{u}[k] \leq \boldsymbol{u}_{max}. \end{aligned} \tag{4.64}$$

Particle swarm optimization, PSO, which is one the most used evolutionary optimization techniques, is applied to multi-variable optimization problems for many years. It does not only imitates human social behaviours but also simulates the social skills of animals by considering them in groups, [45], [16]. PSO is based on an improvement of the performances of individuals and their groups by observing them individually and collectively. Group or swarm can be represented with their individuals from 1 to N for cascaded PID controller as follows:

$$\begin{aligned} \boldsymbol{G}_1 &= [\{K_{p_1}\}_1\ \{K_{i_1}\}_1\ \{K_{d_1}\}_1\ \{K_{p_2}\}_1\ \{K_{d_2}\}_1\ \{K_{p_3}\}_1\ \{K_{i_3}\}_1\ \{K_{d_3}\}_1]^T, \\ \boldsymbol{G}_2 &= [\{K_{p_1}\}_2\ \{K_{i_1}\}_2\ \{K_{d_1}\}_2\ \{K_{p_2}\}_2\ \{K_{d_2}\}_2\ \{K_{p_3}\}_2\ \{K_{i_3}\}_2\ \{K_{d_3}\}_2]^T, \\ \vdots &= \qquad\qquad\qquad \vdots \qquad\qquad\qquad , \\ \boldsymbol{G}_N &= [\{K_{p_1}\}_N\ \{K_{i_1}\}_N\ \{K_{d_1}\}_N\ \{K_{p_2}\}_N\ \{K_{d_2}\}_N\ \{K_{p_3}\}_N\ \{K_{i_3}\}_N\ \{K_{d_3}\}_N]^T. \end{aligned} \tag{4.65}$$

where $\{K_{p_1}\}_1$ represents the first particle of the first individual, $\boldsymbol{G}_1$, which means that the proportional term of the PID_1 controller in the first individual. Other particles

can easily be explained by the help of previously indicated notation. Some basic swarm ideas are fulfilled in the PSO optimization techniques to update particles of the swarms meaningfully. These ideas are categorized into three phenomena as:

- **Inertia**: Each particle in the individual wants to keep its current behaviour based upon its old habits and tendency which are proven to be successful in the past. When the gain parameters are updated, this feature will become a more dominant factor because of the fact that deterioration has to be prevented for good gain values.
- **Influence by society**: People in the community are impressed by one who has more successful than others. So, most of the people try to imitate him/her footsteps and behaviour naturally. One of the updated criteria of the gains of the cascaded PID controller is to approximate the swarms to best swarm, $\boldsymbol{G}_{best}$ in each iteration.
- **Influence by neighbours**: Interaction of the people with their neighbours is more influential than society. One can directly share his/her failures, success with close people. At the beginning of optimization, neighbourhood relation is assigned to each swarm based on some topologies. Ring topology that is explained in [45] is adopted to update particle value term which comes from this influence.

The dimension of the multi-variable optimization problem, n, is equal to the length of the gain vector $length(\boldsymbol{G}_*) = 8$. A population has N candidate solutions which means that one of the vectors in this set $\{\boldsymbol{G}_1,\ \boldsymbol{G}_2,\ \ldots,\ \boldsymbol{G}_N\}$ is candidate solution, i.e. $\boldsymbol{G}_i$. is the best minimization vector over given set $i \in [1, N]$. Each particles in the individual, $\boldsymbol{G}_i = [g_{i,1}\ g_{i,2}\ \ldots g_{i,n}]^T$, is moving with some velocity $\boldsymbol{v}_i = [v_{i,1}\ v_{i,2}\ \ldots v_{i,n}]^T$, $i \in [1, N]$, through the search space. This is the essential property of PSO optimization technique which is a fundamental difference between PSO and other evolutionary optimization algorithms. $\boldsymbol{G}_i$ values are updated dynamically from one generation to next by using inertia, the influence of neighbours and society properties. Ratios of influences are selected randomly to update velocity or approaching vectors. These ratios can be considered as learning rates from others. Maximum learning rates are represented with $\{\phi_1\}_{max}$ and $\{\phi_2\}_{max}$ for the influence

of society and neighbour, respectively. ϕ_1 defines the cognition learning rate which is a random variable uniformly distributed in $[0, \{\phi_1\}_{max}]$, i.e. $\phi_{1,i}[k] \sim U[0, \{\phi_1\}_{max}]$ for $i \in [1, n]$. Likewise, social learning rate is also a uniformly distributed random variable in $[0, \{\phi_2\}_{max}]$, i.e. $\phi_{2,i}[k] \sim U[0, \{\phi_2\}_{max}]$ for $i \in [1, n]$. Inertia term to preserve current velocity of particles also has a weight term, w, which is defined as a constant during the optimization. So, particles want to maintain their velocities while learning from their neighbour and society. By considering the above information, velocity update for the $i-th$ particle of the swarm is formulated as follows:

$$\begin{aligned}
\boldsymbol{v}_{i,1} &= w\boldsymbol{v}_{i,1} + \boldsymbol{\phi}_{1,1}(\boldsymbol{g}_{i,1,best} - \boldsymbol{g}_{i,1}) + \boldsymbol{\phi}_{2,1}(\boldsymbol{g}_{i,neig*,1} - \boldsymbol{g}_{i,1}), \\
\boldsymbol{v}_{i,2} &= w\boldsymbol{v}_{i,2} + \boldsymbol{\phi}_{1,2}(\boldsymbol{g}_{i,2,best} - \boldsymbol{g}_{i,2}) + \boldsymbol{\phi}_{2,2}(\boldsymbol{g}_{i,neig*,2} - \boldsymbol{g}_{i,2}), \\
\vdots \quad &= \qquad\qquad\qquad \vdots \qquad\qquad\qquad , \\
\boldsymbol{v}_{i,n} &= w\boldsymbol{v}_{i,n} + \boldsymbol{\phi}_{i,n}(\boldsymbol{g}_{i,n,best} - \boldsymbol{g}_{i,n}) + \boldsymbol{\phi}_{2,n}(\boldsymbol{g}_{i,neig*,n} - \boldsymbol{g}_{i,n}).
\end{aligned} \tag{4.66}$$

where velocity vector $\boldsymbol{v}_i$, is considered as change of the PID parameters of the particle i. It is written as $\boldsymbol{v}_i = [\{\Delta K_{p_1}\}_i \{\Delta K_{i_1}\}_i \{\Delta K_{d_1}\}_i \{\Delta K_{p_2}\}_i \{\Delta K_{d_2}\}_i \{\Delta K_{p_3}\}_i \{\Delta K_{i_3}\}_i \{\Delta K_{d_3}\}_i]^T$. Best PID parameters for each individual that give the minimum value is indicated by $\{\boldsymbol{G}\}_{i,best} = [\{K_{p_1}\}_{i,best} \{K_{i_1}\}_{i,best} \{K_{d_1}\}_{i,best} \{K_{p_2}\}_{i,best} \{K_{d_2}\}_{i,best} \{K_{p_3}\}_{i,best} \{K_{i_3}\}_{i,best} \{K_{d_3}\}_{i,best}]^T$. Each individual has two neighbours that come from the previously explained ring topology. The closest neighbour of individual i, $\boldsymbol{G}_{i,neig*} = [\{K_{p_1}\}_{i,neig*} \{K_{i_1}\}_{i,neig*} \{K_{d_1}\}_{i,neig*} \{K_{p_2}\}_{i,neig*} \{K_{d_2}\}_{i,neig*} \{K_{p_3}\}_{i,neig*} \{K_{i_3}\}_{i,neig*} \{K_{d_3}\}_{i,neig*}]^T$, whose particles give the minimum result in (4.64) is selected to be employed in (4.66). Likewise, two learning rates are transformed into vector forms as $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ to be able to write (4.66) implicitly as follows:

$$\boldsymbol{v}_i = w\boldsymbol{v}_i + \boldsymbol{\phi}_1^T(\boldsymbol{G}_{i,best} - \boldsymbol{G}_i) + \boldsymbol{\phi}_1^T(\boldsymbol{G}_{i,neig*} - \boldsymbol{G}_i) \tag{4.67}$$

(4.67) must be restricted in the predefined limit, $\boldsymbol{v}_{min} \leq \boldsymbol{v} \leq \boldsymbol{v}_{max}$, which can be seen in (4.64) for preventing particles search space leaving. Update equation of the PID parameters in a particle with related the velocity vector is expressed as

$$\boldsymbol{G}_i = \boldsymbol{G}_i + \boldsymbol{v}_i \tag{4.68}$$

(4.68) can be considered as position update procedure which means that PID parameters are updated by summing them with a rate of changes in them. The final value of

$\boldsymbol{G}_i$ is also limited, $\boldsymbol{G}_{min} \leq \boldsymbol{G} \leq \boldsymbol{G}_{max}$, to keep it within the search domain otherwise particle may move away from its optimum solution point. All the particles which are numbered from 0 to N are updated with (4.67) and (4.68). The pseudo code of PSO can be seen in Algorithm 2.

4.4.2 Tuning of the Cascaded PID Controller Parameters

PID parameters which are values in the range of $\{0 \leq K_{p_1} \leq 100, 0 \leq K_{i_1} \leq 10, 0 \leq K_{d_1} \leq 0.5\} \in PID_1$, $\{0 \leq K_{p_2} \leq 50, 0 \leq K_{d_2} \leq 0.5\} \in PID_2$ and $\{0 \leq K_{p_3} \leq 200, 0 \leq K_{i_3} \leq 10, 0 \leq K_{d_3} \leq 0.5\} \in PID_3$ are tuned by using PSO technique on the s-shape path. Parameters of the PSO which can be shown in Table 4.2 are selected via trial and error method. Since PSO is a kind of evolutionary algorithm, positions and velocities of the individuals are generated randomly. In each evaluation, PSO may give different results. In order to get rid of this problem, the PSO algorithm is repeated 100 times. Later, the best particle in each evaluation is selected as a candidate solution for the path following problem. Mean of all the best candidates are taken to minimize randomness of the algorithm. The final values of the optimized cascaded PID controller parameters are written in the following Table 4.3.

Table 4.2: Parameters of the PSO algorithm.

Generation	End of the path
Number of Particle	500
Swarm Inertia, w	0.5
Cognition Learning Rate, $\{\phi_1\}_{max}$	0.6
Social Learning Rate, $\{\phi_2\}_{max}$	0.6
Maximum Position Vector, $\boldsymbol{G}_{max}^T$	$[100 \;\; 10 \;\; 0.5 \;\; 50 \;\; 0.5 \;\; 200 \;\; 10 \;\; 0.5]^T$
Minimum Position Vector, $\boldsymbol{G}_{min}^T$	$[0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0]^T$
Maximum Velocity Vector, $\boldsymbol{v}_{max}^T$	$[1 \;\; 0.1 \;\; 0.01 \;\; 1 \;\; 0.01 \;\; 1 \;\; 0.1 \;\; 0.01]^T$
Minimum Velocity Vector, $\boldsymbol{v}_{max}^T$	$[\text{-}1 \;\; \text{-}0.1 \;\; \text{-}0.01 \;\; \text{-}1 \;\; \text{-}0.01 \;\; \text{-}1 \;\; \text{-}0.1 \;\; \text{-}0.01]^T$

Algorithm 2: PSO method for tuning of cascaded PID controller parameters.

Input: $w,\ \{\phi_1\}_{max},\ \{\phi_2\}_{max},\ N,\ \boldsymbol{G}_{max},\ \boldsymbol{G}_{min},\ \boldsymbol{v}_{max},\ \boldsymbol{v}_{min},\ \sigma,\ T$

Initialize: $N\ individuals\ \boldsymbol{G}_i\ \in \Big[U\big[0,\{K_{p_1}\}_{max}\big]\ U\big[0,\{K_{i_1}\}_{max}\big]\ U\big[0,\{K_{d_1}\}_{max}\big]\ U\big[0,\{K_{p_2}\}_{max}\big]\ U\big[0,\{K_{d_2}\}_{max}\big]\ U\big[0,\{K_{p_3}\}_{max}\big]\ U\big[0,\{K_{i_3}\}_{max}\big]\ U\big[0,\{K_{d_3}\}_{max}\big]\Big]^T,\ i \in [0,N]$

$Number\ of\ N\ \boldsymbol{v}_i\ \forall\ individuals\ \in \Big[U\big[\{\Delta K_{p_1}\}_{min},\{\Delta K_{p_1}\}_{max}\big]\ U\big[\{\Delta K_{i_1}\}_{min},\{\Delta K_{i_1}\}_{max}\big]\ U\big[\{\Delta K_{d_1}\}_{min},\{\Delta K_{d_1}\}_{max}\big]\ U\big[\{\Delta K_{p_2}\}_{min},\{\Delta K_{p_2}\}_{max}\big]\ U\big[\{\Delta K_{d_2}\}_{min},\{\Delta K_{d_2}\}_{max}\big]\ U\big[\{\Delta K_{p_2}\}_{min},\{\Delta K_{p_3}\}_{max}\big]\ U\big[\{\Delta K_{i_3}\}_{min},\{\Delta K_{i_3}\}_{max}\big]\ U\big[\{\Delta K_{d_3}\}_{min},\{\Delta K_{d_3}\}_{max}\big]\Big]^T,\ i \in [0,N]$

$Initialize\ best{-}so{-}far\ position\ \forall\ individuals:\ \boldsymbol{G}_{i,best} \leftarrow \boldsymbol{G}_i,\ i \in [0,N]$

$iteration \leftarrow 0$

while $iteration \leq T(End\ of\ the\ path)$ **do**

  for $i = 1 \rightarrow N$ **do**

    $Generate\ \boldsymbol{\phi}_1 = [\phi_{1,1}\ \ldots\ \phi_{1,8}]^T\ with\ \phi_{1,k} \sim U[0,\{\phi_1\}_{max}]^T,\ k \in [0,8]$

    $Generate\ \boldsymbol{\phi}_2 = [\phi_{2,1}\ \ldots\ \phi_{2,8}]^T\ with\ \phi_{2,k} \sim U[0,\{\phi_2\}_{max}]^T,\ k \in [0,8]$

    $Neighbours\{\boldsymbol{G}_i\} \leftarrow \{\sigma\ nearest\ neighbour\ of\ \boldsymbol{G}_i\}$

    $\boldsymbol{G}_{i,neig*} \leftarrow \arg\min_{\boldsymbol{G}} \{J(\boldsymbol{G}):\ \boldsymbol{G} \in Neighbours\{\boldsymbol{G}_i\}\},\ look$ (4.64)

    $\boldsymbol{v}_i \leftarrow w\boldsymbol{v}_i + \boldsymbol{\phi}_1^T(\boldsymbol{G}_{i,best} - \boldsymbol{G}_i) + \boldsymbol{\phi}_2^T(\boldsymbol{G}_{i,neig*} - \boldsymbol{G}_i)$

    if $\boldsymbol{v}_i > \boldsymbol{v}_{max}$ **then**

      $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_{max}$

    else if $\boldsymbol{v}_i < \boldsymbol{v}_{min}$ **then**

      $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_{min}$

    $\boldsymbol{G}_i \leftarrow \boldsymbol{G}_i + \boldsymbol{v}_i$

    if $\boldsymbol{G}_i > \boldsymbol{G}_{max}$ **then**

      $\boldsymbol{G}_i \leftarrow \boldsymbol{G}_{max}$

    else if $\boldsymbol{G}_i < \boldsymbol{G}_{min}$ **then**

      $\boldsymbol{G}_i \leftarrow \boldsymbol{G}_{min}$

    $\boldsymbol{G}_{i,best} \leftarrow \arg\min\{J(\boldsymbol{G}_i), J(\boldsymbol{G}_{i,best})\}$

  $iteration \leftarrow iteration + 1$

Table 4.3: Optimized parameters of the cascaded PID controller.

K_{p_1}	K_{i_1}	K_{d_1}	K_{p_2}	K_{d_2}	K_{p_3}	K_{i_3}	K_{d_3}
10.83	2.0	0.025	26.07	0.025	150.73	1.50	0.016

In Fig. 4.7, results of the controller with LOS guidance rule are plotted for optimized values of parameters and different valued parameters. Ground truth that is represented with a red colour and dashed line is the shortest path which is composed of way-points. In Table 4.4, simulation parameters are represented with average cross-track errors. Optimized PID is in the first row. Other parameters are the variation of the optimized parameters.

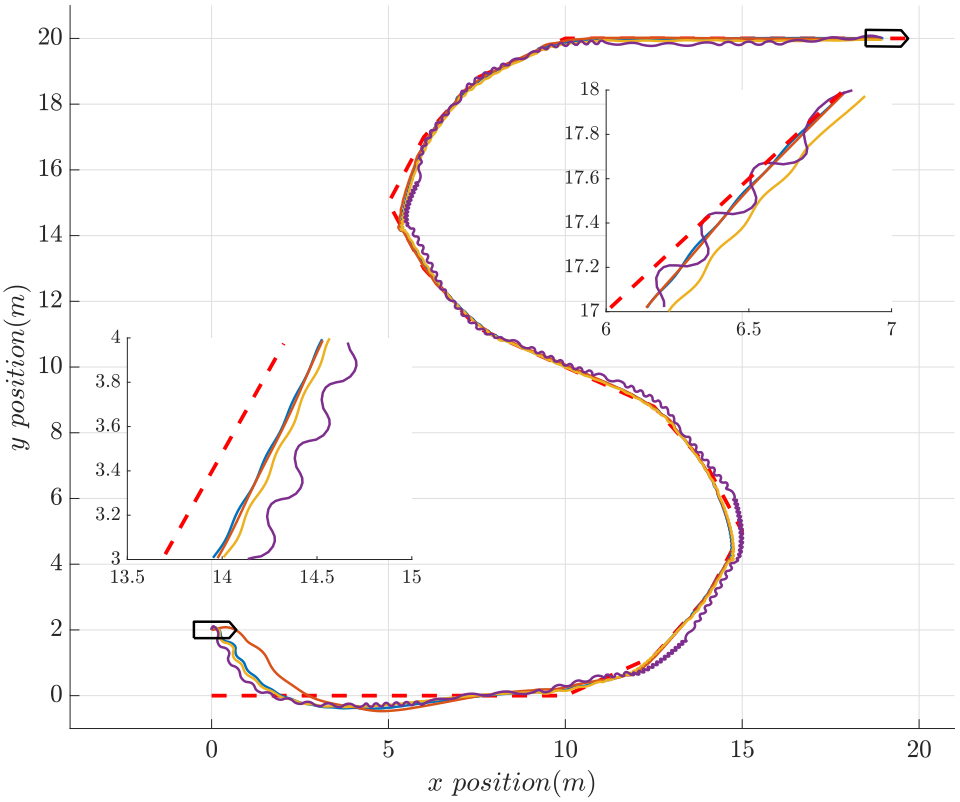

Figure 4.7: Results of the cascaded PID controller on the s-shape path with different optimized parameter values.

Table 4.4: Controller parameters used in Fig. 4.7.

Colour	K_{p_1}	K_{i_1}	K_{d_1}	K_{p_2}	K_{d_2}	K_{p_3}	K_{i_3}	K_{d_3}	Avg. Err.(m)
	10.83	2.0	0.025	26.07	0.025	150.73	1.50	0.016	0.1474
	5.83	2.0	0.025	26.07	0.025	150.73	1.50	0.016	0.1529
	10.83	5	0.025	26.07	0.025	150.73	1.50	0.016	0.1555
	10.83	2.0	0.025	26.07	0.025	150.73	5	0.016	0.2190

4.5 Comparison Between MPC and Cascaded PID Controllers

Both controllers fulfil the path following task on the "S" shaped curved path. MPC directly uses linearized system dynamics and minimizes errors between references and measured signals while it reduces the control effort according to its performance criterion. It also generates a control sequence for the specified input horizon by considering the defined constraints. Cascaded PID controller performs error minimization between references and state feedback signals. If its outputs exceed the physically applicable maximum and minimum torque values, then they are saturated. Saturation process affects the robustness of the controller and responses of the system. Performance criterion for the comparison between MPC and cascaded PID controller is to calculate average cross track errors. As can be seen in Tables 4.1 and 4.4, MPC gives better results compared to cascaded PID controller as it is expected.

CHAPTER 5

PARALLEL DOCKING PROBLEM FOR UNMANNED SURFACE VEHICLES

5.1 Introduction

Autonomously docking a USV to a parking slot can be considered as a subset of two phenomena which are path planning and path following. The path (route) planning part is the determination of the path along which vehicles goes from an initial position to a final point. In the path following, the vehicle autonomously goes by tracking the group of way-points along the optimal and sub-optimal paths and finalizes its motion in the parking slot. First of all, the craft is docked itself into an interim parking region where its orientation parallel is parallel to the parking slot. Then, it starts moving backwardly from this region to its parking point. A path that optimizes the control effort, i.e., an energy-optimal pathway is proposed to find an appropriate path at the first stage. In the second step, it must be determined a path which includes smooth manoeuvres for parallel parking by considering obstacles in the region.

A manoeuvring path generation problem for docking (parking) has been partially solved in many different ways; by solving an optimal control method (time minimum or energy minimum), or by using heuristic approaches for example utilizing fuzzy logic, or by using geometric rules [50]. Combinations of these approaches are adopted to find feasible way-point for docking manoeuvres in the scope of this thesis study.

5.2 Problem Definition

After completing a mission, the vehicle should start following an optimally found path which begins at the $\boldsymbol{p}_s = [x_s \ y_s]^T$ on the quadratic circle and ends at the $\boldsymbol{p}_f = [x_f \ y_f]^T$ planar point as it may be seen in Fig. 5.1 with for two scenarios. Vehicle's orientation should be parallel to the port at the forward point, $\boldsymbol{p}_f$.

An intermediate point, $\boldsymbol{p}_i = [x_i \ y_i]^T$, that is defined by considering park corners is utilized to obtain a curved path for docking to a parking slot. The path between $\boldsymbol{p}_f$ and $\boldsymbol{p}_i$ are followed backward. Then, the vehicle continues to its motion to park a docking slot whose length is about the twice of vehicle's dimensions. From this point, the way-points to accomplish the necessary docking manoeuvre between $\boldsymbol{p}_i$ and $\boldsymbol{p}_p = [x_p \ y_p]^T$ are generated via a proposed geometric rule, and the guidance and controller laws are employed to the vehicle to follow this path. A docking manoeuvre path, which can be shown as blue colour in the above figure, is produced between intermediate point, $\boldsymbol{p}_i$, and parking point, $\boldsymbol{p}_p$ for a parking slot of a predetermined size.

The vehicle may approach the parking region in any direction; however, it should be

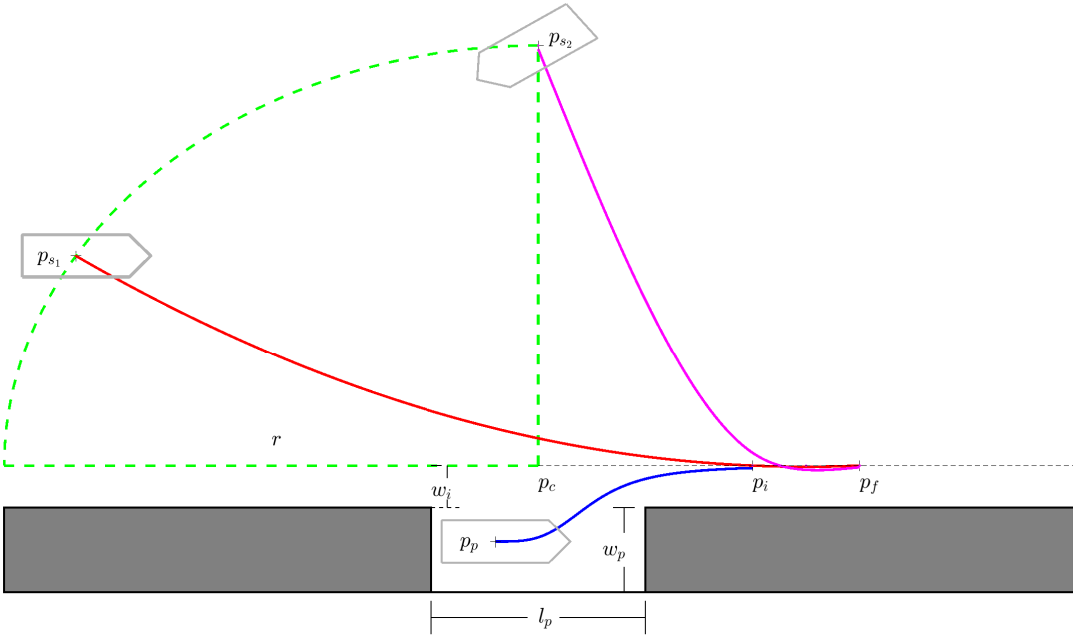

Figure 5.1: Representation of parallel parking with important parking points.

noted that planar symmetry simplifies the solution of the problem. Parallel docking problem is treated as a way-point generation problem that may be dealt with in two steps: while the first stage is entrance to the parking site, the second part is to fulfil the backward docking manoeuvre. Some of these way-points and important points are illustrated in Fig. 5.1.

5.3 Entrance to Parking Site

It is accepted that the vehicle passes through a way-point on a quadratic circle whose centre at $\boldsymbol{p}_c$, and a radius has a reasonable value. This way-point is described as the starting point, $\boldsymbol{p}_s$, which is represented in Fig. 5.1, at the beginning of parking. The vehicle directs its orientation from this point to the intermediate point, $\boldsymbol{p}_i$, as its next way-point. However, the orientation of the craft should be parallel to the parking slot at p_i before beginning backward docking manoeuvre. Therefore, forward point, $\boldsymbol{p}_f$ as an additional point, which is also indicated in Fig. 5.1, should be specified as the next way-point to fix orientation (yaw angle, ψ) of the vehicle parallel to the port. Once the USV arrives at the forward point, the next way-point will is assigned as the intermediate point, which is the point backward docking manoeuvre will start.

The distance between $\boldsymbol{p}_s$ and $\boldsymbol{p}_f$ is a function of initial speed, u, orientation ψ, approaching angle α of the vehicle, and it should be optimized for obtaining an energy-optimal path. Only $\pi/2 \leq \alpha < \pi$ is represented in 5.1, and the intermediate point, $\boldsymbol{p}_i$, is located at the left side of the forward point, $\boldsymbol{p}_f$. A symmetric problem may be solved for $0 \leq \alpha < \pi/2$, and $\boldsymbol{p}_f$ will be at the left side of $\boldsymbol{p}_i$ in this scenario. As a consequence of that the docking problem is described in $\pi/2 \leq \alpha - \psi \leq 3\pi/2$. Here note that the symmetric problem is to be easily solved for $\pi \leq \alpha - \psi \leq 3\pi/2$. The optimization will be performed by addressing the following optimal control problem to determine energy efficient path.

5.3.1 Optimal Path Between $\boldsymbol{p}_s$ and $\boldsymbol{p}_f$ for Forward Docking Maneuver

Optimal control problem, (5.1), is proposed solved to obtain an optimal path between $\boldsymbol{p}_s$ and $\boldsymbol{p}_f$. It is defined open-loop optimal control as two-point boundary value prob-

lem with free time. Initial state $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ at time t_0, final state $\boldsymbol{x}(t_f) = \boldsymbol{x}_f$ at time t_f, constraints on the states and inputs are known. $\widehat{\boldsymbol{x}}(t_f) = \widehat{\boldsymbol{x}}_f$ indicates the desired final states. The objective function and its constraints of the optimal control problem in continuous time are given in the following equation.

$$\begin{aligned}
&CostFunction && J(\boldsymbol{u}(t)) = \int_{t_0}^{t_f} h(\boldsymbol{u}(\tau))d\tau \\
&\text{subject to} && \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \;\; \boldsymbol{x}(t_f) = \boldsymbol{x}_f \\
& && \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \\
& && \boldsymbol{u}_{min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{max}, \;\; t \in [t_0, t_f] \\
& && x_i + L \leq x_f \text{ and } y(t) \geq y_f \\
& && |\psi(t_f)| \leq 5^\circ.
\end{aligned} \tag{5.1}$$

$h(\boldsymbol{u}(t)) = \boldsymbol{u}(t)^T \boldsymbol{R}(t)\boldsymbol{u}(t)$ is expressed as control effort cost which is minimized to find energy-optimal path, and time-varying positive definite $\boldsymbol{R}(t)$ matrix is taken as constant, i.e. $\boldsymbol{R}(t) = \boldsymbol{R}$. $\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t))$ is the state equations of continuous time system which is written as inequality constraints. $\boldsymbol{u}(t)_{min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}(t)_{max}$ indicates constraints on the inputs. To avoid collisions at the parking site and make the orientation of the vehicle parallel in the forward position, $x_i + 2L \leq x_f$ and $y \geq y_f$, in a planar position, are added as inequality constraints (L indicates the length of the vehicle). (5.1) is discretized for numerically calculating optimal input sequences.

$$\begin{aligned}
&CostFunction && J(\boldsymbol{U}) = \sum_{k=k_0}^{k_f-1} \boldsymbol{u}[k]^T \boldsymbol{R}\boldsymbol{u}[k] \\
&\text{subject to} && \boldsymbol{x}[k = k_0] = \boldsymbol{x}_0, \;\; \boldsymbol{x}[k = k_f] = \boldsymbol{x}_f \\
& && \boldsymbol{x}[k+1] = \boldsymbol{f}(\boldsymbol{x}[k], \boldsymbol{u}[k]), \;\; k = k_0, k_0+1 \;\; \ldots, k_f - 1 \quad \text{(look 2.11)} \\
& && \boldsymbol{u}_{min} \leq \boldsymbol{u}[k] \leq \boldsymbol{u}_{max}, \qquad k = k_0, k_0+1 \;\; \ldots, k_f - 1 \\
& && x_i + L \leq x[k_f] \text{ and } y[k] \geq y_f \\
& && |\psi[k_f]| \leq 5^\circ.
\end{aligned} \tag{5.2}$$

where $\boldsymbol{U}$ vector is the input sequence, and its entities are variables of the optimization problem (5.2). $k_f - k_0 = N$ samples are taken, and the number of unknowns in the problem becomes $2 \times N$ because of the two inputs. $x[k_0] = x(t_0)$ and $x[k_f] = x(t_f)$ should be satisfied. By keeping relations the same, other terms are only written in discrete-time. The constrained quadratic optimization problem is solved to find an

optimal control sequence, $\boldsymbol{U}^{optimal} = \left[\boldsymbol{u}[k]^T\ \boldsymbol{u}[k+1]^T\ \ldots\ \boldsymbol{u}[k_f-1]^T\right]^T$, which gives the energy efficient path for the USV.

5.3.2 Solution of the Optimal Control Problem

Large scale optimization problem one of the "Global Optimization Toolbox" of MATLAB, genetic algorithm (GA), is employed to be able to solve above discrete-time system [34]. It is suitable for solving most of the constrained and unconstrained, highly nonlinear, smooth or nonsmooth optimization problems which may not be easily found from other methods. Principles of the GA are based on evolutionary optimization algorithms. At each generation, many of individuals in a population have the ability to reproduce to share their genetic information (crossover). The lifetime of each individual in the population is limited. Small variations in individuals allow the evolutionary process to produce a new generation apart from previous (mutation). Survival in the population is directly and positively correlated with reproducing of individuals (elitism) [45]. All these biological factors are also taken into consideration for the optimization problem, (5.2), to obtain a possible optimal control sequence.

5.3.2.1 Scenario I

The following scenario is selected to demonstrate the performance of the optimization algorithm: initial point state being $\boldsymbol{p}[k_0] = \boldsymbol{p}_s = [x[k_0]\ y[k_0]]^T = [-3.66\ \ 7]^T$, $u[k_0] = 1$ m/sec, and $\psi[k_0] = 0$ rad and one forward point state $\boldsymbol{p}_f = [x_f,\ y_f]^T = [8\ 2]^T$, $u[k_f] = 0$ m/sec, and $\psi[k_f] = 0$ rad at time k_f. The upper limit of the control input is taken as 40 N, i.e. $\boldsymbol{u}_{max} = [40\ 40]^T$, and the lower limit is set as $\boldsymbol{u}_{min} =$ [-40 -40]^T The radius of the quadratic circle is set as $R = 10$ meter. An initial population, i.e. sequence of control inputs represented with black colour in Fig. 5.2, which is obtained via MPC+LOS combination is given to GA as an initial population in order to obtain a faster convergence. Parameters of the GA are indicated in the following Table 5.1.

Energy-optimal path found from GA algorithm is demonstrated in Fig. 5.2 with the blue colour path. It should be understood that first, the orientation of the vehicle is

Table 5.1: Adjusted parameters of the GA.

Generation Limit	10000
Population Size	50
Elitism	10% of population size
Mutation	Adaptive
Crossover Fraction	0.7

directed toward the final forward point $\boldsymbol{p}_f$. When the craft reaches the parking region, it starts manoeuvring to dock itself parallel to the border. When the orientation of the vehicle satisfies the neighbourhood of the desired yaw angle, $-\epsilon \leq \psi[k] \leq \epsilon$, in addition to meeting position requirements, $x_i + L \leq x\big[k \leq k_f\big]$ and $y[k \leq k_f] \geq y_f$, constraints of the optimization problem are satisfied before final time.

Fig. 5.3 shows the change in the cost value during the GA optimization process. At

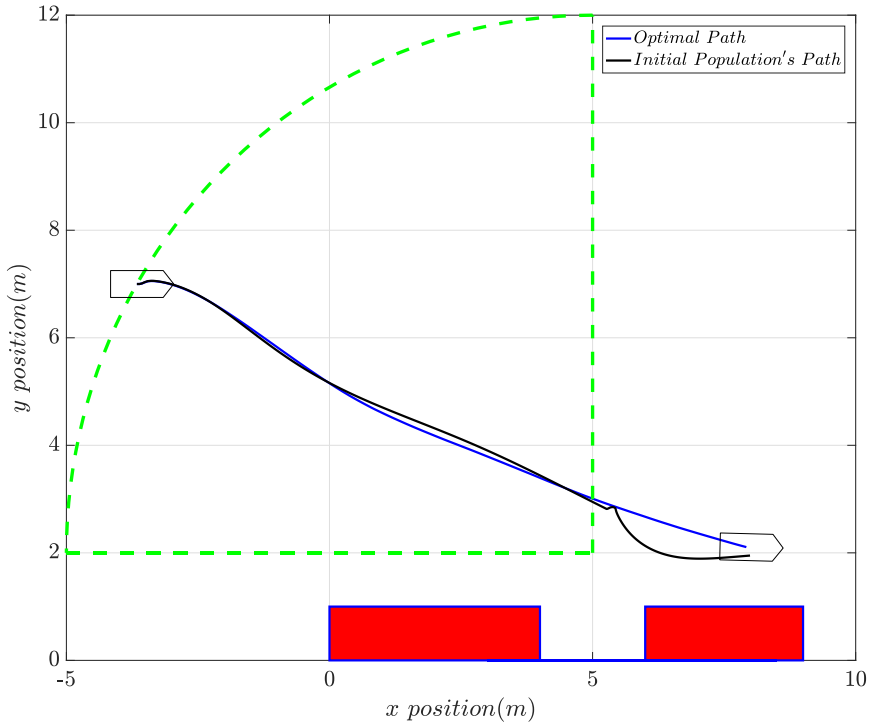

Figure 5.2: Energy optimal path for Scenario I represented in blue line.

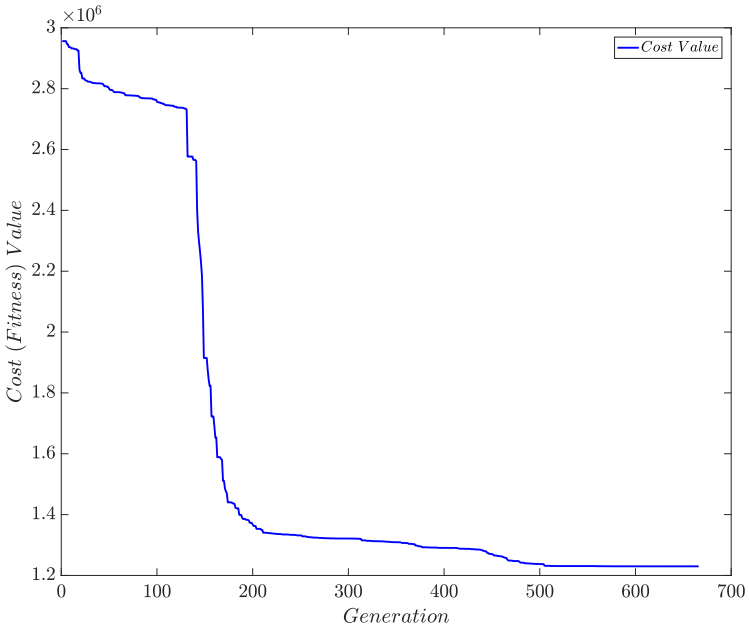

Figure 5.3: Evaluation of the cost function for Scenario I.

the initial, the best individual in the population has 2.96×10^6 cost value. At the termination of the GA, in generation 666, the cost of the best-valued individual is calculated as 1.23×10^6. GA algorithm improves the cost value approximately by 41.6%. It can be seen that GA improves cost function slowly due to the dimension of

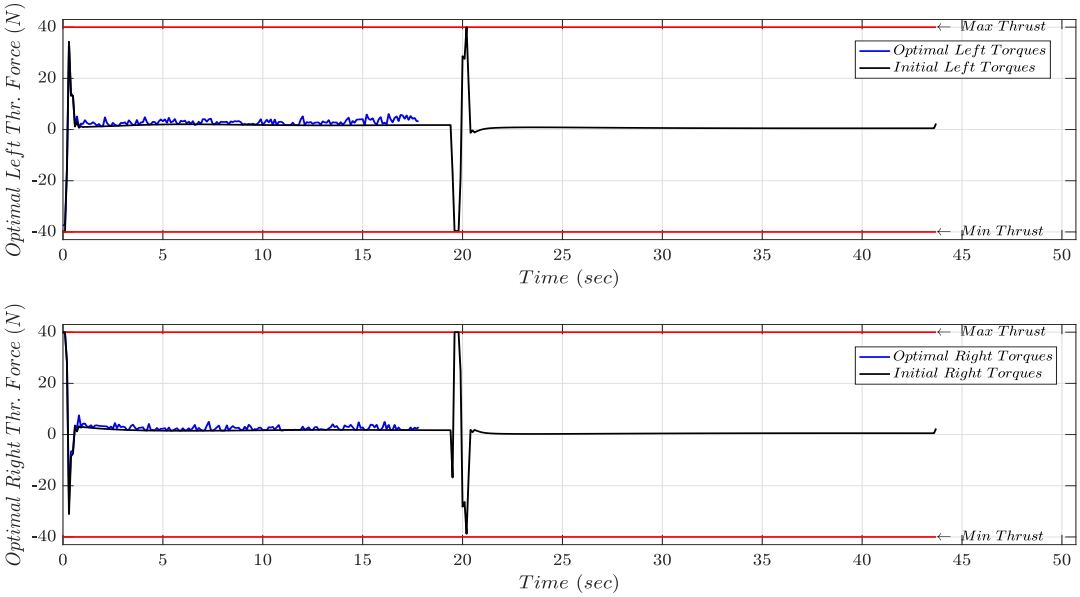

Figure 5.4: Initial and optimal control torques for Scenario I.

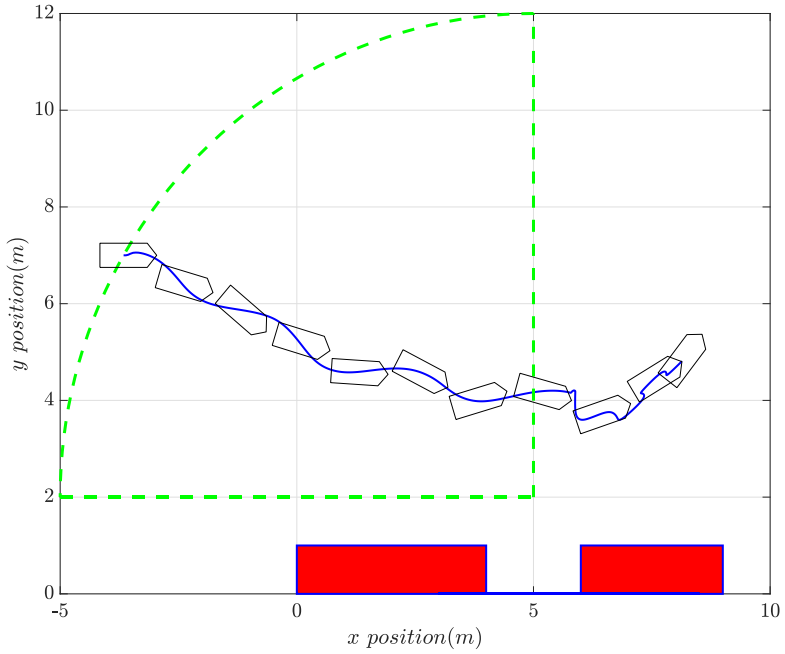

Figure 5.5: Effect of the disturbance on optimal control signals.

the optimization problem. The sequence of the optimal control input for left and right thrusters are represented in Fig. 5.4. GA finds appropriate input signals which satisfy boundary input constraints. If these signal are directly applied to derive the vehicle existence of the wave generated by wind (equations are given in 2.10), the path in Fig. 5.5 is obtained. This figure shows that the optimal path can be used to generate way-points for the vehicle; nevertheless, guidance and autopilot control is necessary for robustness.

5.3.2.2 Scenario II

The second scenario is as follows: $\boldsymbol{p}[k_0] = \boldsymbol{p}_s = [x[k_0]\ y[k_0]]^T = [1.58\ \ 11.4]^T$, $u[k_0] = 1$ m/sec, and $\psi[k_0] = 20^\circ$ and forward point state $\boldsymbol{p}_f = [x_f,\ y_f]^T = [8\ 2]^T$, $u[k_f] = 0$ m/sec, and $\psi[k_f] = 0$ rad. Constraints of the optimal control problem and parameters of the same GA algorithm are same with Scenario 1, 5.3.2.1. An initial population, i.e. a sequence of control inputs resulted in the trajectory with black in Fig. 5.6, which is obtained via MPC+LOS combination. This sequence is also given to GA as an initial population in order to obtain a faster solution.

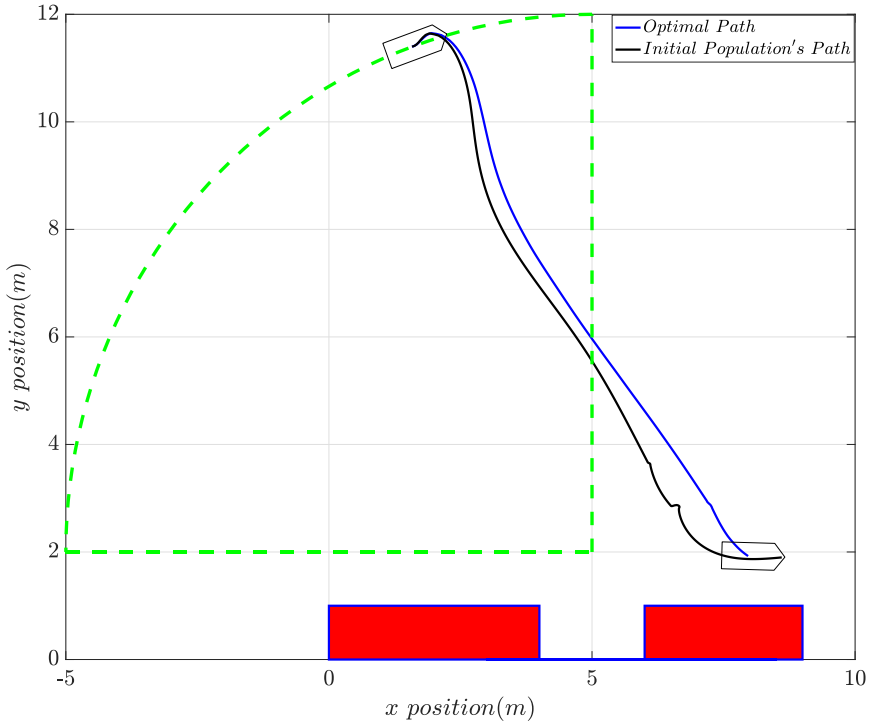

Figure 5.6: Energy optimal path for Scenario II represented with blue line.

Evaluation of the cost function during the optimization process can be seen in Fig. 5.7. At the initial, the best individual in the population gives 5.55×10^6 cost value.

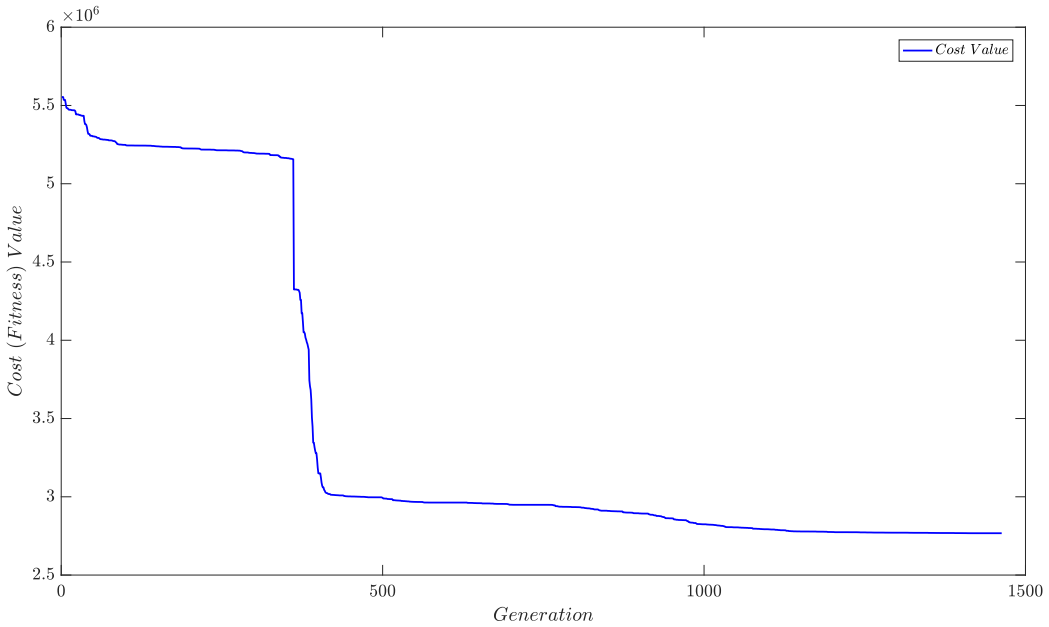

Figure 5.7: Evaluation of the cost function for Scenario II.

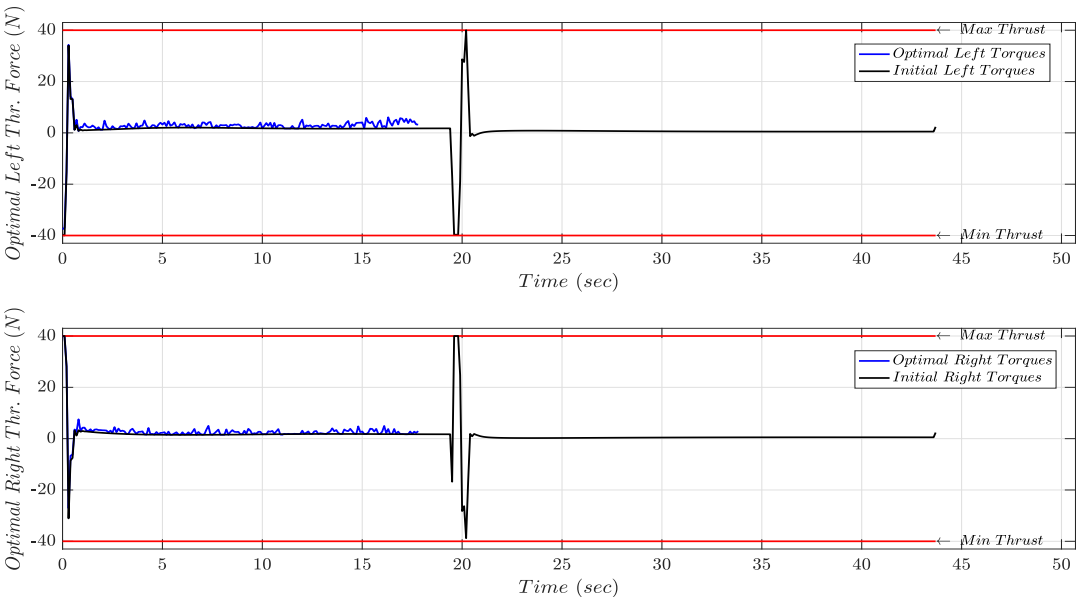

Figure 5.8: Initial and optimal control torques for Scenario II.

At the termination, GA stops in generation 1463, where the cost of the best valued individual is calculated as 2.76×10^6. GA algorithm is improved the initial cost value approximately by 49.7%. Optimal input sequences are demonstrated with blue lines in Fig. 5.8.

5.4 Backward Docking Maneuver

After vehicle reaches to the forward point, it follows a continuous curvature path to approach its parking slots. This curve is defined between the intermediate point, $\boldsymbol{p}_i = [x_i \ y_i]^T$, and the parking points $\boldsymbol{p}_p = [x_p \ y_p]^T$. To be able to realize collision-free parking manoeuvre, $\boldsymbol{p}_i$, $\boldsymbol{p}_p$ and edges of the parking slot should be known [14]. In this study, docking manoeuvre to the parking slot is achieved by following way-points on the 4 parameters logistic continuous curve s-shape. This curve also satisfies collision avoidance at the edge of the slot by considering the geometry of the craft, and yaw rate of the vehicle is restricted to a predefined maximum value during the manoeuvre. This path obtained from the following equation and conditions [50]:

$$\begin{aligned} y &= \frac{y_p - y_i}{1 + (\frac{x - x_p}{C})^B} + y_i \\ x &\geq x_p \geq 0, \ y_i > y_p > 0, \ B > 2, \ C > 0 \end{aligned} \tag{5.3}$$

where variables B and C adjust shape and slope of the curve which is employed for executable path following. Curved path obtained from (5.3) also guarantees zero yaw angle, i.e. $\psi = 0$, at the neighbourhood of $\boldsymbol{p}_i$ and $\boldsymbol{p}_p$ with well-defined B and C parameters. An example of the proposed path is illustrated in Fig. 5.1 with blue colour.

5.5 Results

The parallel docking problem for the USV with different values of initial speed, vehicle pose and approaching angle of the vehicle is solved for all the combinations of autopilot and guidance law designs. Energy consumption (E) and average cross-track error d are calculated in (5.4) along the motion.

$$E = \sum_{k=k_0}^{k_f-1} \left\{ \frac{1}{2}\, m\, (u[k]^2 + v[k]^2) + \frac{1}{2}\, I_z\, r[k]^2 \right\}$$

$$d = \frac{1}{k_f - k_0} \sum_{k=k_0}^{k_f-1} \underbrace{\frac{|(y_w^* - y_{w-1}^*)x[k] - (x_w^* - x_{w-1}^*)y[k] + x_w^* y_{w-1}^* - y_w^* x_{w-1}^*|}{\sqrt{(y_w^* - y_{w-1}^*)^2 + (x_w^* - x_{w-1}^*)^2}}}_{The\ closest\ distance\ at\ time\ k} \tag{5.4}$$

Note that energy consumption, E, is composed of the kinetic and rotational energy components. $u[k]$ and $v[k]$ indicate surge and sway speeds, respectively. $r[k]$ is the yaw rate of the vehicle at time k. m and I_z denote mass and inertia terms, respectively. Average cross-track error, d, is calculated from summed cross-track errors. $\boldsymbol{p}_{w-1}^* = [y_{w-1}^* \; x_{w-1}^*]^T$ and $\boldsymbol{p}_w^* = [y_w^* \; x_w^*]^T$ are the two consecutive way-points at time k. Indexed summation variable is the closest distance of $(x[k], y[k])$ from the line which passes through $\boldsymbol{p}_{w-1}^*$ and $\boldsymbol{p}_w^*$.

In Scenario I, 5.3.2.1, way-points on the optimal path are generated with for speed of 1 m/s, approaching angle of $2\pi/3$ rad, and initial vehicle heading angle of 0 rad, i.e. $\psi = 0$ in Fig. 5.9. Guidance algorithms are compared by using PID controllers for this scenario in Fig. 5.10. Average cross tracking errors are 0.0501 and 0.0832 meters for LOS and PP, respectively. Energy consumption of the PID+LOS combination is 37.80 J while PID+PP combination consumes 36.95 J.

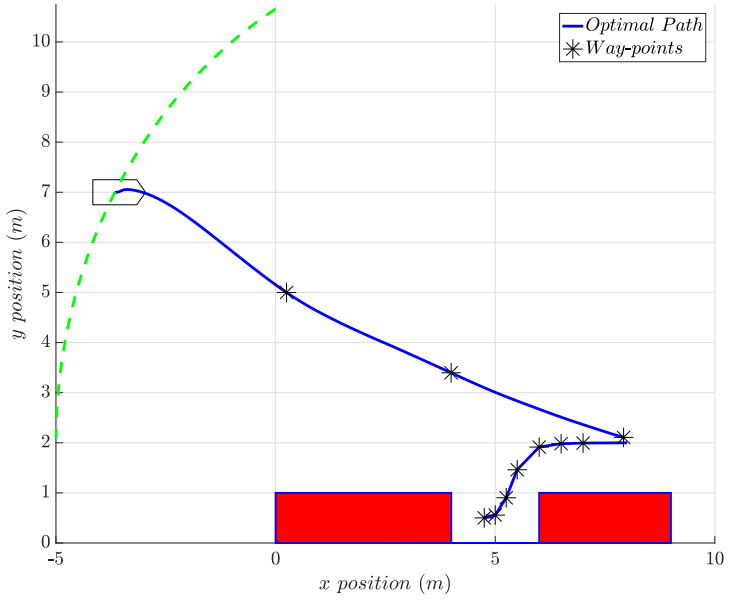

Figure 5.9: Optimal and sub-optimal paths and generated way-points for Scenario I.

The same scenario is applied to the MPC controller and its guidance method combinations. Fig. 5.11 illustrates MPC+LOS and MPC+PP simulations blue and red lines, respectively. Average cross tracking errors are measured as 0.0443 and 0.0812 meters for LOS and PP, respectively. Energy consumption of the MPC+LOS combination is calculated as 33.9 J while the motion of the MPC+PP combination requires 35.60 J.

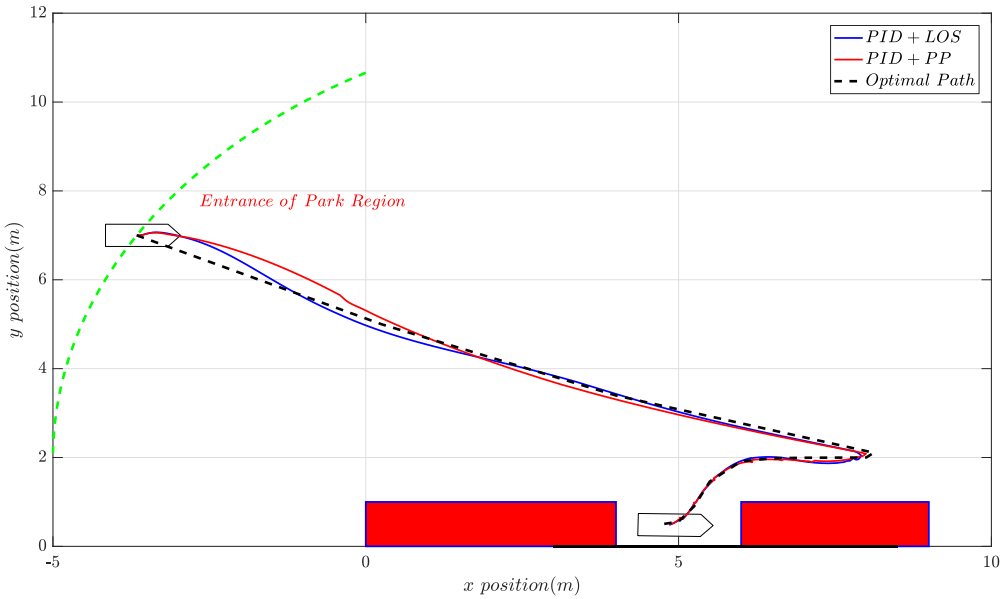

Figure 5.10: Simulation of PID+LOS and PID+PP methods for Scenario I.

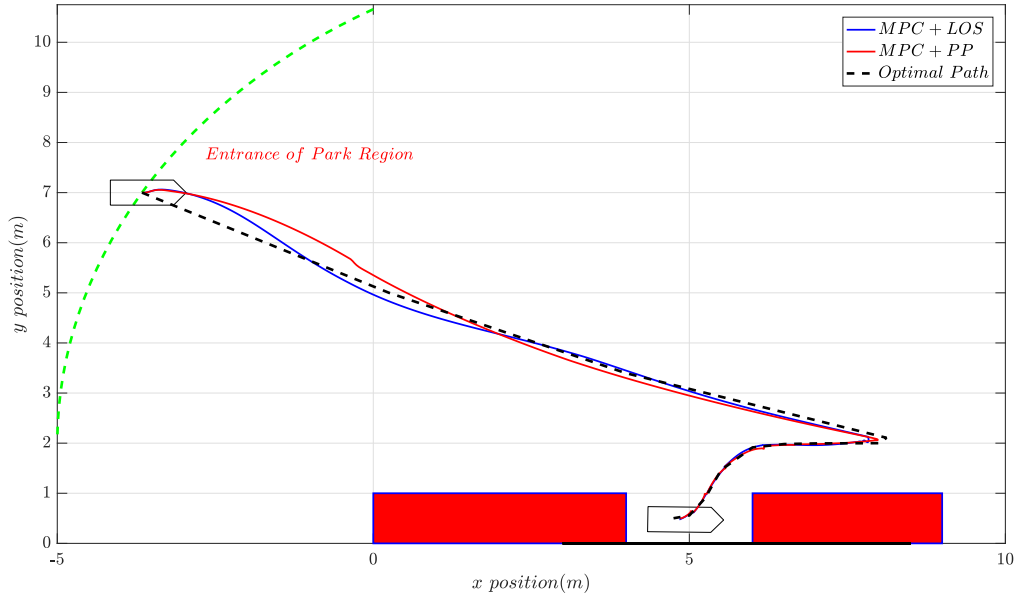

Figure 5.11: Simulation of MPC+LOS and MPC+PP methods for Scenario I.

For the second scenario, 5.3.2.2, way-points on the optimal path are generated with for speed of 1 m/s, approaching angle of 110°, and initial vehicle heading angle of

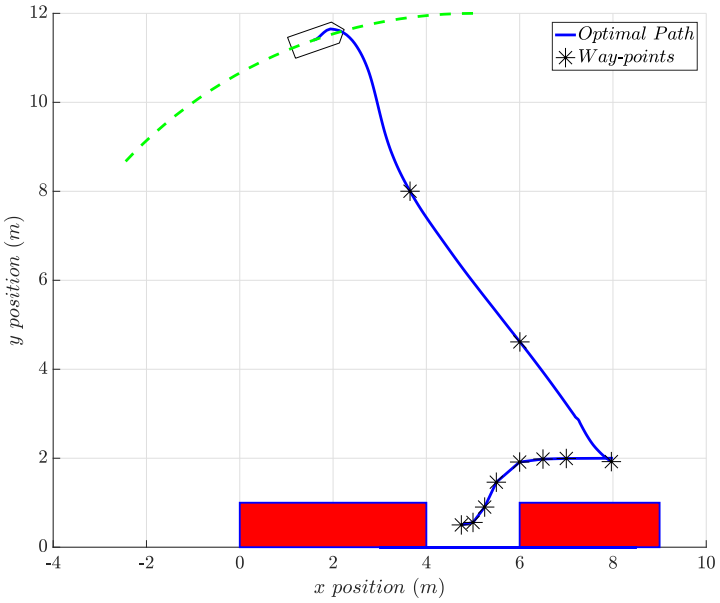

Figure 5.12: Optimal and sub-optimal paths and generated way-points for Scenario II.

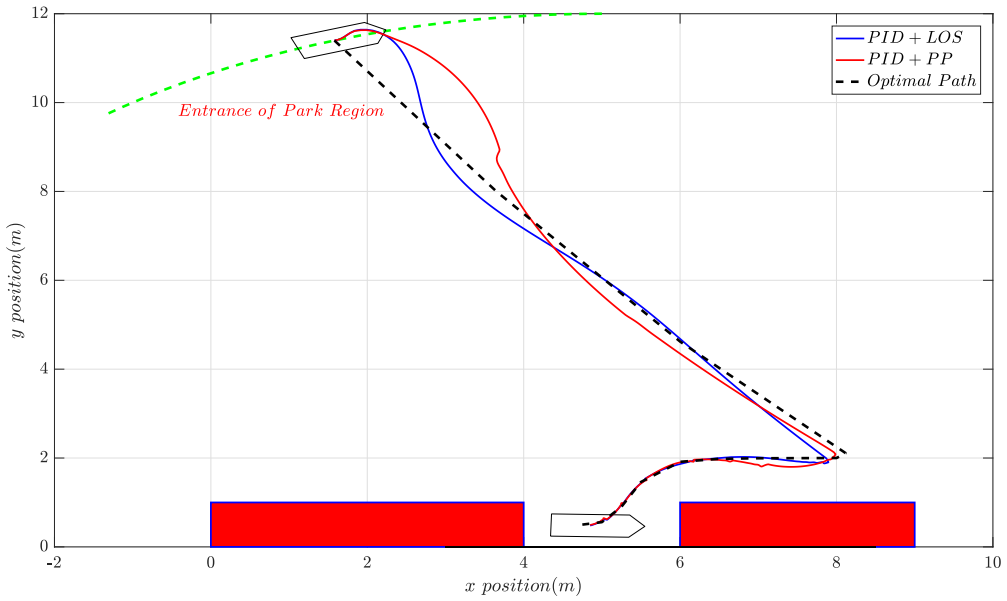

Figure 5.13: Simulation of PID+LOS and PID+PP methods for Scenario II.

20° rad, i.e. $\psi = 0$ in Fig. 5.12. PID+LOS and PID+PP combinations are plotted in Fig. 5.13. Average cross tracking errors are 0.0592 and 0.1386 meters for LOS and PP, respectively. Energy consumption of the PID+LOS combination is 35.01 J while PID+PP combination consumes 36.13 J. MPC controller and its guidance method combinations are demonstrated for Scenario II in Fig. 5.14 Average cross tracking er-

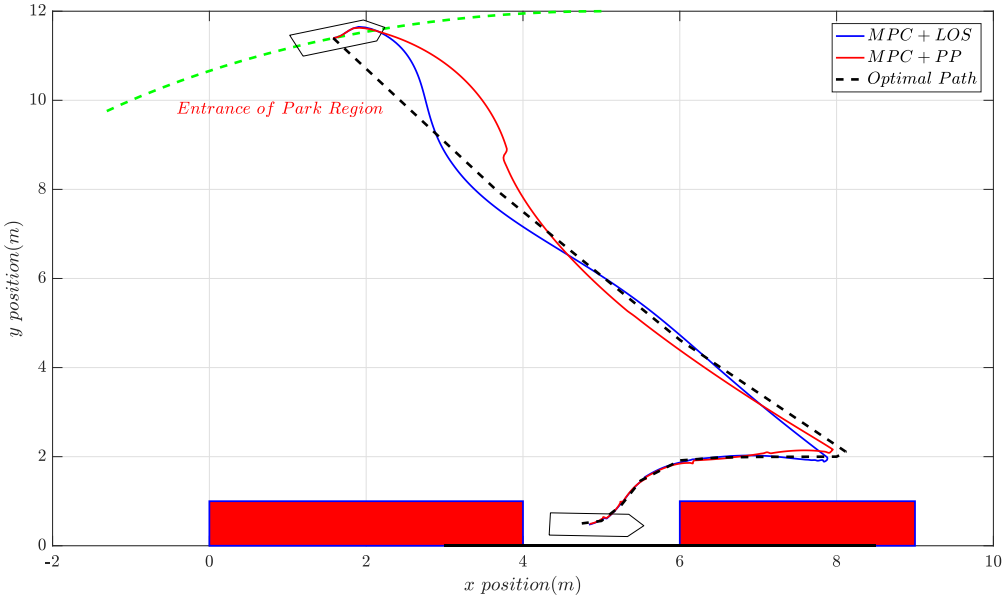

Figure 5.14: Simulation of MPC+LOS and MPC+PP methods for Scenario II.

rors are calculated as 0.0578 and 0.1330 meters for LOS and PP, respectively. Energy consumption of the MPC+LOS combination is estimated as 34.83 J while the motion of the MPC+PP combination requires 35.86 J.

For different values of initial vehicle pose and approaching angle, tracking performances of MPC+LOS, MPC+PP, PID+LOS and PID+PP are presented in Fig. 5.15. It is observed that none of the combinations of the autopilot and guidance designs violate the parking site limitations. This validates our approach of tuning the controller parameters with an objective to optimize the path following performances. It is found that LOS guidance law drives the vehicle closer to the reference trajectory and MPC performs better compared to PID autopilot.

In Fig. 5.16, energy consumptions of the USV performing parallel docking manoeuvres under the rule of different combinations of autopilot and guidance designs are shown as a function of approaching angle and initial yaw position. It is observed that the energy consumption of the vehicle does not change dramatically between different designs. It is observed that LOS guidance law drives the vehicle in a more energy efficient manner, better performing along with MPC.

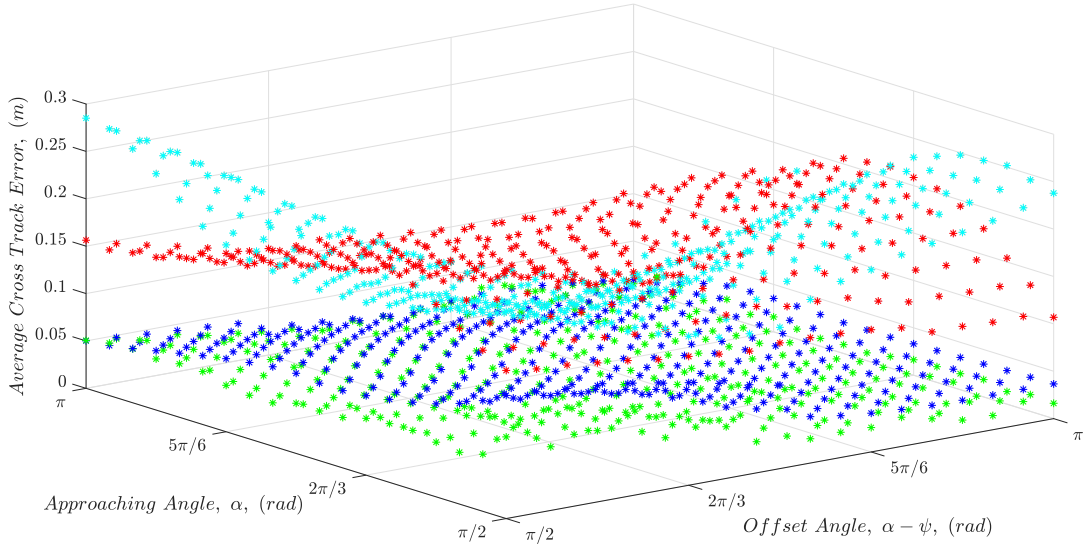

Figure 5.15: Average cross track of the vehicle for four different combination of controller and guidance method: MPC+PP in cyan, MPC+LOS in green, PID+PP in red, PID+LOS in blue.

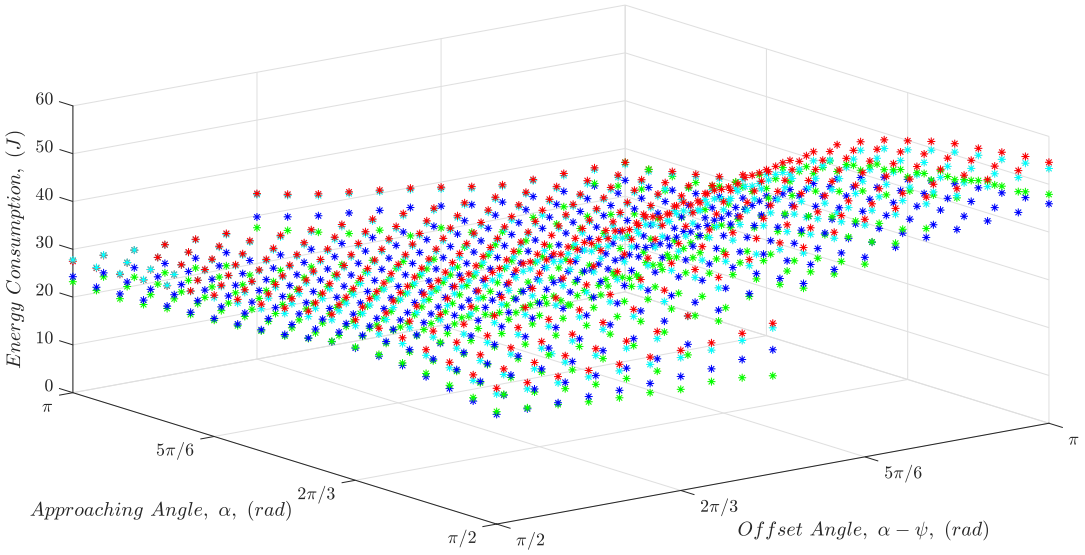

Figure 5.16: Energy consumption of the vehicle for four different combination of controller and guidance method: MPC+PP in cyan, MPC+LOS in green, PID+PP in red, PID+LOS in blue.

CHAPTER 6

EXPERIMENTAL SETUP AND RESULTS

6.1 Introduction

One of the guidance techniques, pure-pursuit (PP), and one the controller methods, model predictive control (MPC) are implemented on board for experimental validation of the thesis. Pacific Islander Tugboat which is represented in Fig. 6.1 is employed to fulfil experimental work. This boat is equipped with the motor driving system, cooling system, regulator, batteries as a rigid body. The mathematical model of the vehicle which is derived in Chapter 2 is used with its previously identified parameters, [17], to be able to design an onboard controller. Entirely open source Pixhawk 1 Flight Controller which is fitted onto the middle front section of the tugboat is employed for experimental validation. Embedded software, also can be called firmware, which is run on NuttX Operating System (OS) runs in real-time for collecting sensor data, driving motors via electronic speed controller (ESC), communication with the ground station, estimating state variables, recording logs during the experiment. Motion control hierarchy defined in Chapter 3 is satisfied fast and reliably via embedding controller and guidance methods in stable firmware. The ground station is only used for sending start and stop commands, and for receiving monitoring information of the boat such as battery and sensor fusion status, the number of used positioning satellites etc.

This chapter is divided into three major parts that are hardware components and software part and experimental results and validation. Part of hardware components includes detailed information about the model boat, autopilot card, ESC, motors, sensors. In software parts, the software architecture for experimental validation is ex-

plained in detail. Last part includes the results of the experimental study.

6.2 Physical Components and Hardware

This section gives information about the technical and physical specification of the modeled USV. First of all model boat is introduced along with its components. Later, autopilot card employed for the experimental result is explained with its sensors and peripherals.

6.2.1 Model Boat and Its Components

A modified 1:40 scale Pacific Islander Tugboat is considered as the vehicle with dimensions of 900x290x260 mm. It is a one-piece fibreglass hull with rubber fenders as tires, and its weight is 11 kg including all the equipment (ESCs, motors, batteries, ballast weights, autopilot card etc.), shown in Figure 6.1. The propellers located at the aft render the vehicle highly manoeuvrable so that it can almost turn around its centre of gravity without altering its linear position and it can travel up to 3 m/s. Two distinct

Figure 6.1: Pacific Islander Tugboat.

four-bladed propellers that are made of brass have clockwise and counter-clockwise turning capacity with appropriate ESCs and motors. Each propeller is surrounded by Kort Nozzle that provides effective control of the stream of water passing through it. The function of the Kort Nozzles is to change the direction of the thrust using a servo motor which is connected to each nozzle. Dynamics of the nozzles are not included and ignored in this study by doing servo motor motionless and fixing them during motion. The vehicle can be viewed from the stern with propellers and Kort nozzles in Fig. 6.2.

Brushless, waterproof motors whose rated values are same and ESCs are used to drive the modelled vehicle for experimental purposes. Two 3360SL 3180 KV motors and Seaking-120 A-V3 ESCs are employed to turn the thrusters in clockwise and counter-clockwise directions. This equipment can be seen in Figs. 6.3a and 6.3b.

Duties of the ESCs are to take pulse-width modulation (PWM) signals which are square pulse waves between 0 and 5 volts. PWM signals are produced based upon duty cycle technique which is expressed as the percentage of 'on' time of rectangular or square wave in a period. ESC interprets duty cycles and produces a constant voltage (negative or positive valued) during that period. The generated voltage value is applied to the electrical motor that is high currents resisting. Then, the motor shaft which is coupled with the propeller shaft starts rotating with a known RPM value because of the applied voltage. Since outputs of the controller or system inputs are

Figure 6.2: View from stern with propellers of Pacific Islander Tugboat.

(a) A 3360SL 3180 KV motor (b) A 120 A-V3 ESC

Figure 6.3: Motor and ESC used in experimental setup, respectively.

torque command, thrust-voltage characteristic given in Fig. 6.4 must be known to control of the boat motion.

Selection of the battery plays a crucial role to feed all the electronic components. ESCs serve as a bridge between battery and motors. It consumes most of the electric power in battery cells. Remain power is used for Pixhawk autopilot card, sensors and small water pump. In order to draw constant and proper current for electronics system, Li-Po battery is selected as a power supply. It has been decided that a three

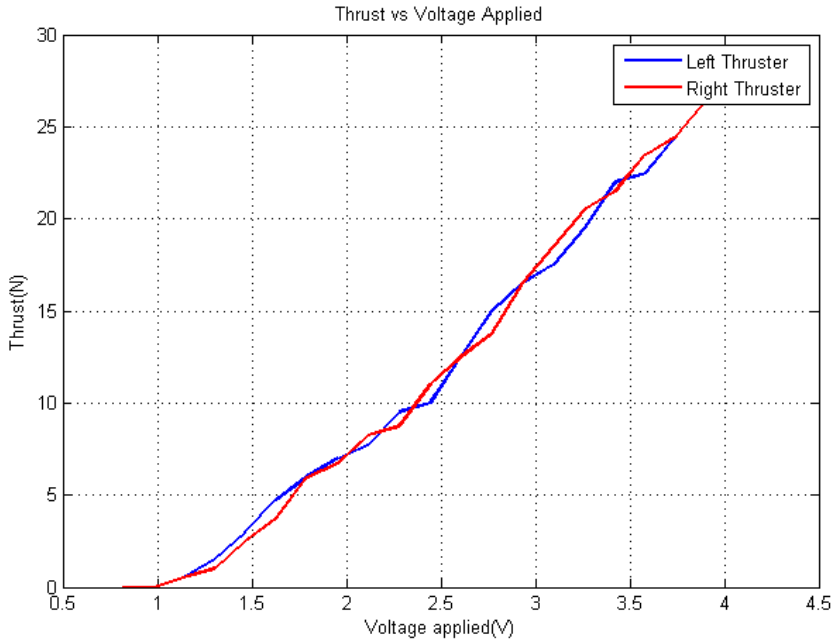

Figure 6.4: Experimentally obtained thrusts (N) vs. applied voltage (V) plot [17].

Figure 6.5: Li-Po battery that provides electric power for all the system.

cells 3300 mAh, 11.1 V lithium-polymer (Li-Po) battery, in Fig. 6.5, is used for experimental work.

6.2.2 Autopilot Board and Its External Components

Pixhawk® 1 autopilot board whose software and hardware are open source has been determined to conduct experiments. It may be used in commercial and scientific unmanned systems due to its flexible and customizable hardware and software. Lightweight, efficient and very stable NuttX, kind of real-time operating system (RTOS), can efficiently perform real-time control system tasks without any need to companion computer. Software development is carried out in C/C++ programming language with or without using an open source integrated development environment (IDE). Pixhawk autopilot card can be seen in Fig. 6.6. Two hardware modules, named as Flight Management Unit (FMU) and Input/Output (IO), are integrated for full system functionality. FMU can be considered that it is the main component of the autopilot

Figure 6.6: A view of Pixhawk autopilot card [2].

card. All the programs, sensor fusion, control algorithms are operated on the FMU. FMU includes an ARM Cortex-M4F micro-controller running at 168 MHz with DSP, 1024 KB (kilobyte) of flash memory and 192 KB of RAM. A 3 axis gyro (a L3GD20 by ST Microelectronics) for orientation, a 3 axis accelerometer and a 3 axis gyro (a MPU-6000 by Invensense) for determining outside influences, a compass (magnetometer) combination of a accelerometer (a LSM303D by ST Microelectronics) for heading and a barometric pressure sensor (a MS5611 by Measurement Specialities Inc.) for determining altitude are the sensors that are located on the FMU. Following physical interfaces are used for connectivity 1x I2C, 1x CAN, 4x UART.

IO module is the carrier board between FMU and other external devices and modules. It contains its own 24 MHz cortex-M3 micro-controller and stacks. It has direct battery input power supply which provides 5V stable DC and limits the current for FMU. 8 high-speed servo PWM signal pins are available to drive motors. 2 solid relays and a variety of PPM-SUM/SBUS input connectors are the other connectivity part of the I/O module.

A group of peripherals is employed to perform communication, to obtain position and altitude information from the GNSS system, to measure wind velocity, to activate and deactivate the PWM outputs. A pair of 433 MHz, plug-and-play, 500 mW HKPilot telemetry radios provides communication between the ground station and autopilot card to send basic commands and MAVLink packets. A Ublox Neo-M8N GPS with compass provides position and minimal acquisition times from GPS, GLONASS and Galileo satellite systems with 10 Hz navigation update rate. External compass or magnetometer gives yaw angle information between autopilot and true North. JST-GH MS4525DO digital airspeed sensor by mRo measures the wind velocity. Necessary safety switch with an internal indicator LED is used for activating and deactivating the PWM outputs. It can be so useful in case of emergency.

6.3 Software Architecture of the Pixhawk Autopilot Card

Pixhawk has a sophisticated software architecture, also known as (PX4), which can be divided into four essential layers. All these layers can be seen in Fig 6.7. Lower

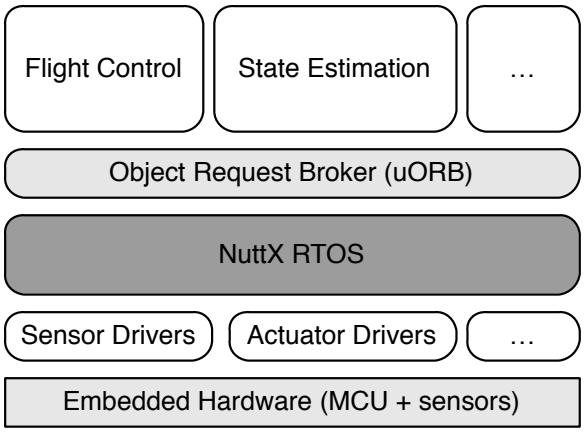

Figure 6.7: Software layers of the PX4 [35].

two fragments constitute the first layer which consists of device drivers and specific software for the micro-controller unit. The second layer is the real-time operating system of the autopilot card which provides interfaces between upper and lower layers. Micro object request broker (uORB) is included in the third layer to make efficient interprocess communication. The uppermost fragment is the application layer for customizable applications such as states estimation, flight control programs etc.

Besides the above explanations, there is a general robotic layer, which known as middleware that includes drivers of the sensors, uORB publish-subscribe message bus, communication with the external peripherals and devices. Software architecture has well adjusted three features. Its all functionality can be split into reusable components. Internal communication between modules is done via asynchronous publish/-subscribe messaging. It may handle an unpredictable workload.

Representation of PX4 software architecture with *Middleware* blocks and component of the *Motion Control* stacks can be seen in Fig. 6.8. The upper part of the figure is the *Middleware*, and the lower part indicates *Motion Control* (flight stack) blocks. Role of the message bus, uORB, is to provide communications between each module with the help of publish/subscribe messages. Parameters can be upload to autopilot via *param* program card and saved to EEPROM/SD Card/ FLASH via *logger* program. Basic commands which are sent from the ground station and information of the preferred message which is sent to the ground station are fulfilled by using the *mavlink* program. *MAVLink* block uses particular communication message frames to transmit data over

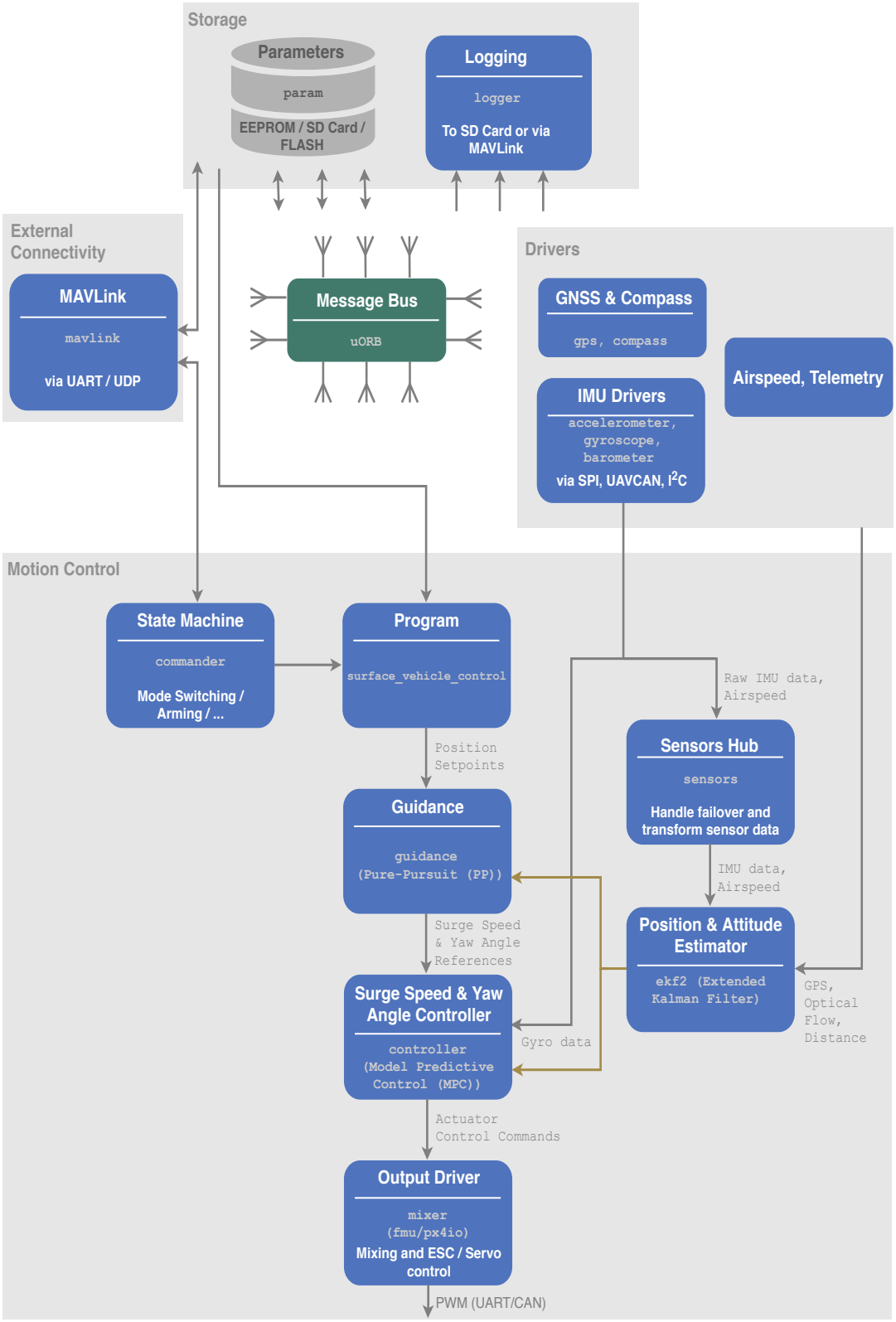

Figure 6.8: High-level software architecture designed in Pixhawk controller.

telemetry radio from/to ground station or autopilot card. Measurements of the sensors are implemented with each sensor driver in the *Drivers* block.

Raw data of sensors from GPS, compass, accelerometer, gyroscope, barometer and airspeed are sent to *Sensor Hub*, *Position & Altitude Estimator* and *Surge Speed & Yaw Angle Controller* blocks which are in the *Motion Control* block for sensor fusion and other purposes. In the *Sensor Hub* block, sensors data is transformed, and failover is handled by the help of *sensors* program. This block gives proper and improved raw data for estimation. Revised extended Kalman filter program, *ekf2*, processes measurements of sensors by using the Estimation and Control Library (ECL) and gives estimated quaternion, attitudes, linear and angular velocities, position and sensor biases in the North-East-Down and Global frame. All the information is shared with related programs via uORB message packets. This program can also detect and report significant and critical sensor failures to a user.

State Machine block is responsible for sending commands from the ground station to Pixhawk board. There are lots of useful commands such as monitoring current RAM and CPU usage of the autopilot card, starting/stopping motors, rebooting systems etc. In this study, the *commander* program is also used for sending start/stop commands to the user-defined *surface_vehicle_control* program. It also provides fast response to stop motion in case of emergency. The *surface_vehicle_control* program starts a constant time loop where *pp_guidance* and *controller* programs are called. Position setpoints (way-points) and estimated local position in NED frame are taken as inputs in the *guidance* program and are converted to reference surge speed and yaw angle. The pure-pursuit method is employed to obtain references.

After, *Surge Speed & Yaw Angle Controller* which includes *controller* program block produces actuator control inputs from reference values and estimated surge speed and yaw angle of the vehicle. Heavy matrix calculations are done for the model predictive controller by the aid of lightweight, dependency-free Matrix Library [1]. Its output is applied torques for left and right thrusters. They are scaled in a range, $[-1,\ 1]$, by considering maximum and minimum torque values is obtained from Fig. 6.4. Propellers turn in the backward direction when outputs in $[-1,\ 0)$. Forward rotations of the propeller are accomplished with actuator commands in $(0,\ 1]$. Propellers are

at the standstill position with 0 actuator commands. At the final stage, *mixer* program converts control commands of actuators to PWM signals to drive motors.

6.4 Experimental Results

Experiments were done with model predictive control (MPC) and pure pursuit (PP) guidance combination for forwarding motion. It can be shown in Fig. 6.9 that the vehicle tries to track a path which is drawn with black line segments. These line segments represent the shortest route. Path illustrated with red line shows the simulation result. The path includes 9 way-points with varying size circle-of-acceptances (CoAs). The experimental result is indicated by blue curvature path. It should be noted that the experimental result and simulation are almost overlapping over most of the motion.

Reference yaw angle, $\widehat{\psi}$, and vehicle's yaw angle, ψ, are plotted in the Fig. 6.10. $\widehat{\psi}$ and ψ are represented by blue and red lines, respectively. The error between these two variables is considerably small. This performance can also redound on the previous Fig. 6.9. It should be noted that the experiment was done by using rope. Although

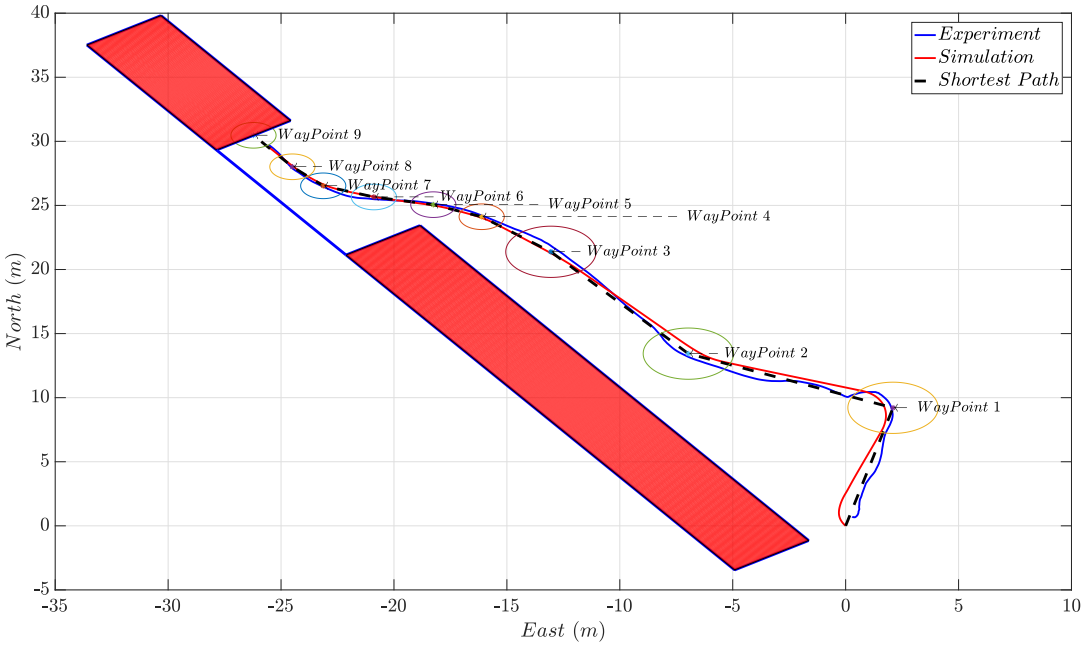

Figure 6.9: Motion of the vehicle during experiment in 2D represented in blue line.

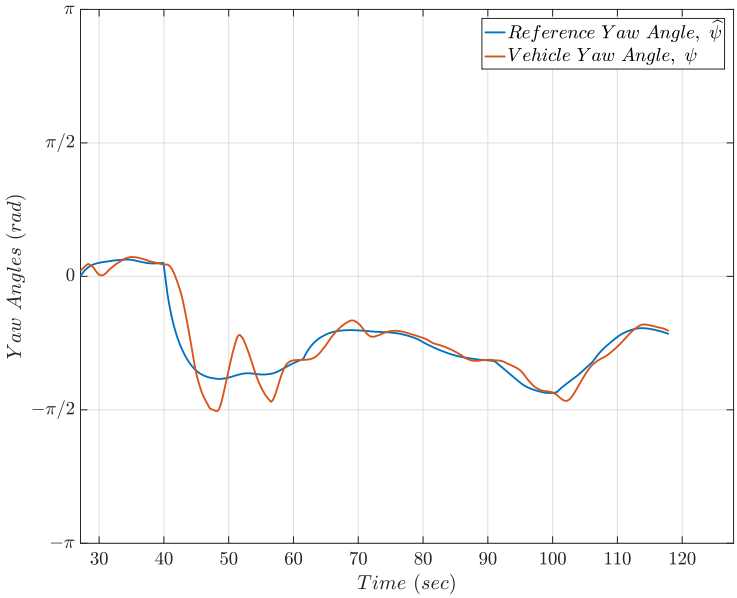

Figure 6.10: Comparison between reference yaw angle $\widehat{\psi}$ and yaw angle of the vehicle, ψ, during experiment

its effect so small, it can affect the system response when manoeuvres occur. When the *surface_vehicle_control* program was run, the reason for starting time simulation is between 40 and 50 is due to the timestamp of the autopilot card.

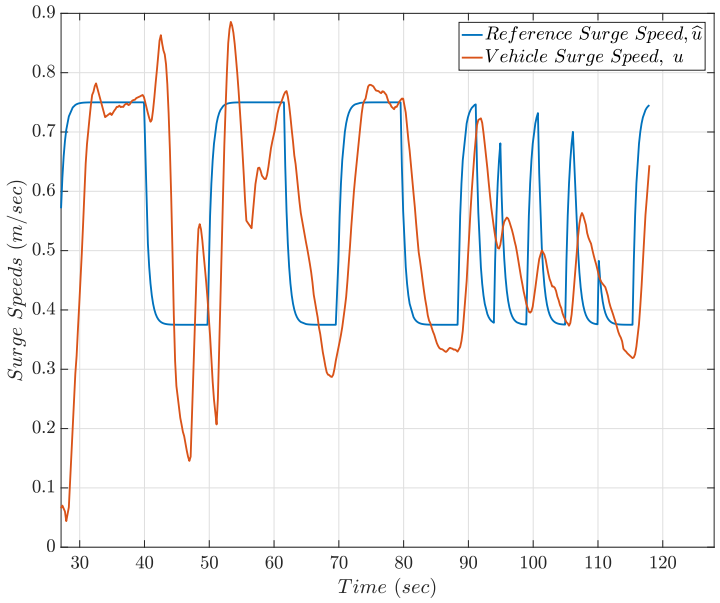

Figure 6.11: Comparison between reference surge speed $\widehat{u}$ and surge speed of the vehicle, u, during experiment

Surge speed tracking performance of the controller is pointed out in Fig. 6.11. The blue colour plot indicates the reference surge speed, $\widehat{u}$, the red colour plot is the vehicle's surge speed, u during the motion. As it is seen in the figure, reference speed, $\widehat{u}$, is given with lower values. In small surge speeds, bias values of the speed measurements which comes from estimation (*ekf2* program) could approach reference value. It is hard to estimate surge speed in lower values due to biases on estimated linear velocities. Also, using low-cost IMUs sometimes give untrusted measurements which dramatically affect system behaviour. Keep in mind that this study was performed on the first generation Pixhawk autopilot card.

Experimentally produced forces for thrusters are represented in Fig. 6.12. The maximum thrust value is taken 40 N while the minimum amount of torque is -40 N in accord with Fig. 6.4. Duration of the control loop is determined by setting a subscription interval of a uORB messages. This interval also implies the sampling rate. It is shown in Fig. 6.13 that Pixhawk autopilot card performs well with 0.1 second sampling rate which includes quite small jitters.

Pixhawk is capable of measuring current and voltage during the motion. These values

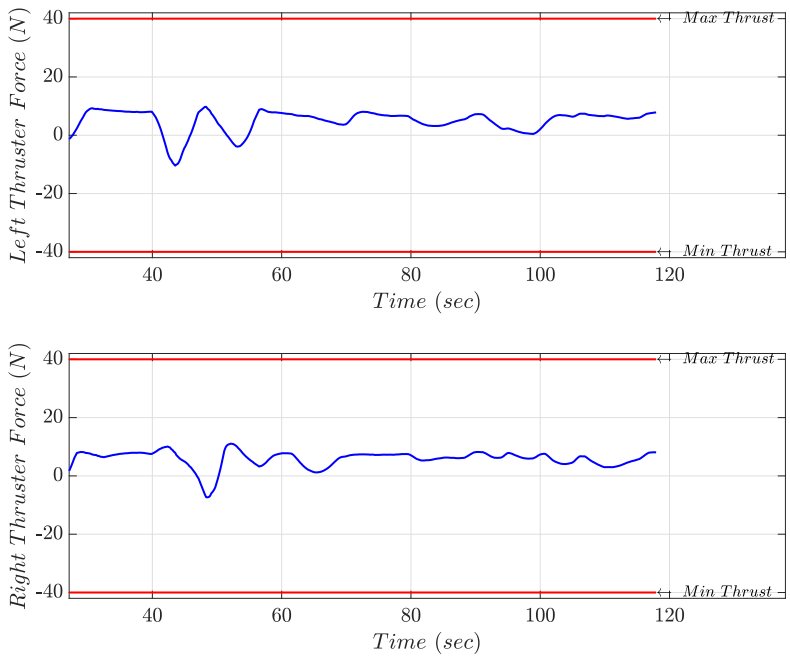

Figure 6.12: Applied left thrust and right thrust commands during the motion, respectively.

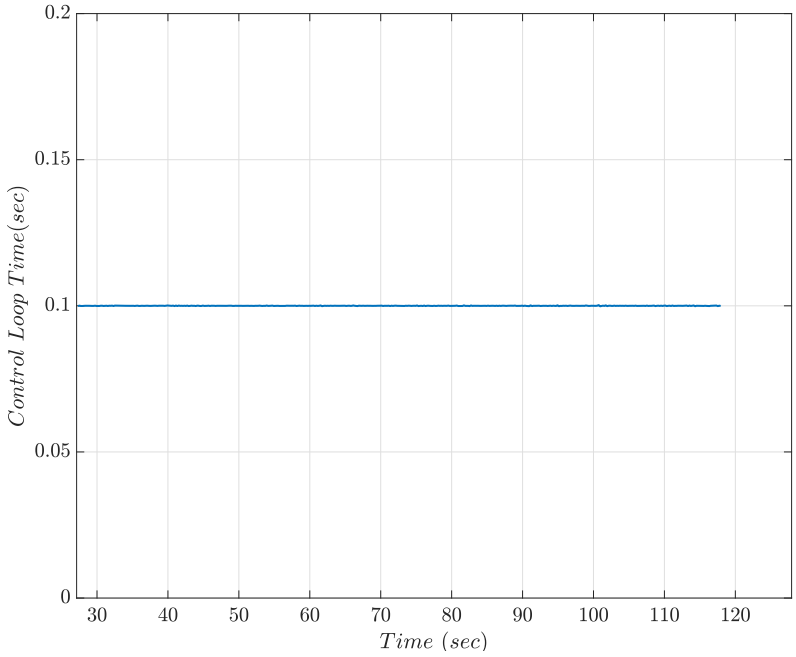

Figure 6.13: Sampling times during the motion with small jitters.

are filtered by autopilot card to be able to observe smooth values. In Fig 6.14, current and voltage values of the battery are demonstrated.

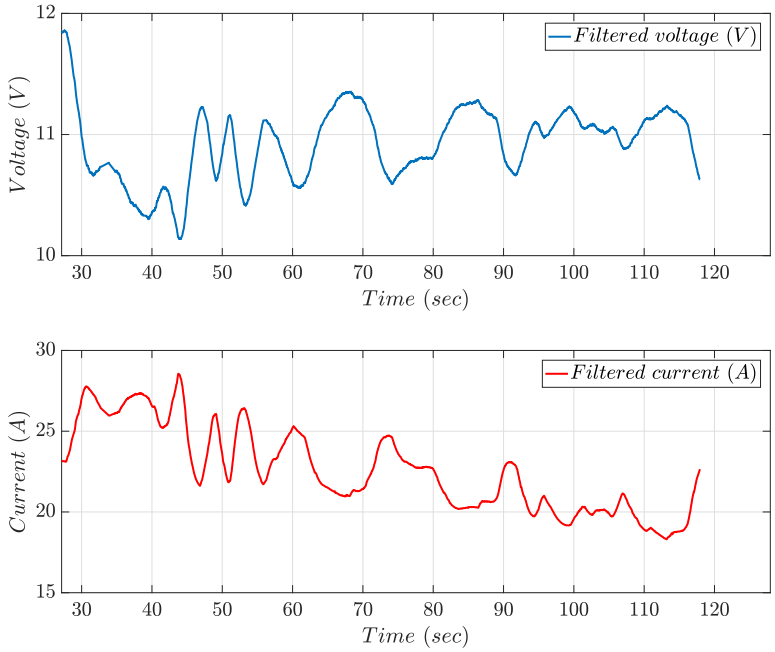

Figure 6.14: Voltage (V) vs. time (sec) and current (A) vs. time (sec) plots.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

7.1 Summary and Remarks

The primary purpose of this thesis is to enhance full autonomy abilities of unmanned surface vehicles. Essential parts of the study are guidance and controller chapters utilized in the parallel docking (parking) problem. These comprise the structural elements of the motion control hierarchy. Desired docking path for the full autonomous motion was obtained via the combination of optimal control and geometric approach.

The mathematical model with identified parameters which was successfully obtained in [17], became important especially controller design. Kinematic equations with dynamic equations which were derived from Fossen's vectorial model [19] established nonlinear six-degrees-of-freedom (6-DOF) mathematical model. After broad literature surveys were done for modeling, it has been decided that the vectorial model was the most appropriate one for controller design. First, kinematic relations between body and NED (reference) frame were formulated to make a meaningful transformation between frames. The model includes the rigid body and added mass, Coriolis and centripetal dynamics together for unmanned surface vehicles. Firstly, rigid body dynamics were obtained by two different methods that are Newton-Euler and Lagrangian with Kirchoff's equations. Then other external forces and moments that come from damping, gravity, buoyancy and air drag phenomena acting on the vehicle's body were added to rigid body dynamics. Environmental forces and moments which result from wave, wind, ocean current are also taken into consideration for stability analysis. Some terms in the dynamic equations were neglected due to their insignificant effect in the motion. The mathematical model was verified with MAT-

LAB simulations along had a series of test cases. So, the reliable mathematical model was validated before controller design.

The following step was to derive guidance algorithms which are defined in the strategic (organizational) level of the motion control hierarchy, in Fig. 3.1. Various guidance algorithms in the literature were searched for solving parallel docking problem. Two of them which are reliable for autopilot design were implemented in real-time and simulation purposes. Guidance algorithms produce reference yaw angle and surge speed information from vehicle's current planar position and related way-point. Propellers of the vehicle have the capability of turning clockwise and counter-clockwise directions. So reference surge speed can be produced either positive or negative according to motion direction. First proposed guidance algorithm was line-of-sight (LOS) guidance which directs the vehicle towards to line segment between two consecutive way-points. The second algorithm, pure-pursuit (PP) guidance or way-point guidance, were implemented to track path. Each method was compared for measuring path tracking performances in parallel docking scenarios. It has observed that LOS gives better results than PP guidance according to cross-track error and energy consumption.

After obtaining reference surge speed and yaw angle, controllers have been investigated and designed to realize motion of the vehicle. Two kinds of autopilot algorithms, model predictive control (MPC) and cascaded PID were implemented. The main purpose was to developed MPC which is more suitable than the PID controller when system dynamics are considered to become powerful for autonomous applications. The second controller cascaded PID has been designed to make a comparison with MPC. Tuned parameters of both controllers were obtained along an "S" shaped curvature path. Adjustable variables of MPC were optimized in the defined range. Particle swarm optimization method was utilized to find best cascaded PID parameters. The linearized mathematical model was used for the MPC method whereas cascaded PID makes use of nonlinear system model. Path tracking and disturbance rejecting performances were compared to each controller. It has predictably observed that MPC has higher performance than cascaded PID even it was designed with the linearized mathematical model.

Next stage was to define the problem of parallel docking which is also named as manoeuvre planning for autonomous parallel parking. It was discussed in two steps. In the first step, the vehicle reaches the parking side by making its orientation in parallel with the port. The desired path was obtained by solving a constraint optimal control problem according to an initial and final position, environmental constraints. Then, it backwardly follows a set of way-points on the continuous-curvature path in the second stage. The route between a parking and initial point of which backward manoeuvres starts is formed from geometric methods. "S" shape 4 parameter logistic continuous curvature equations are used for parallel docking. Combinations of the previously defined controller and guidance algorithms were compared to determine the best method. It has been observed that LOS guidance law derives the vehicle closer to the reference trajectory and MPC performs better compared to cascaded PID autopilot.

Finally, MPC controller and PP guidance methods are utilized in the motion control hierarchy in hardware implementation with Pixhawk Flight controller card on the 1:40 scale Pacific Islander Tugboat. Due to harsh environmental conditions and limited time, tests were done for only forward parallel parking in ODTÜ Yalıncak Gölet. It was found out that vehicle's path following performance is adequate for this guidance and autopilot combination. All mathematical calculations were done in the control loop for the considerably small sampling time meanwhile other background processes like sensor fusion, publishing/subscribing data etc. were executed in real-time.

I learned lots of useful knowledge during the thesis study. It was so challenging how to make a vehicle autonomous for an intended purpose. Determination of the guidance algorithms and their implementation for real-time problem made this study difficult especially at angle changes of discontinuous functions. Other challenging work was to design an embedded real-time model predictive controller which includes high dimension matrices multiplication, summation and inversion. Experimental work was implemented with low cost, open source and error-prone hardware and software.

7.2 Future Works

This study covers the parallel docking of an unmanned surface vehicle by the help of way-point generation, guidance and controller design based on a derived mathematical model. Although all the proposed methods were implemented in the simulation environment, only the combination of MPC and PP guidance was executed for experimental study.

As future works, the following studies can be done:

- Tests were done in the large reservoir lake. An open olympic pool can be a good choice to fulfil parallel docking which includes backward motion. So, other controller and guidance combination can be easily achieved.
- Thrust - PWM (pulse with modulation) characteristics of each motor are not known very well. By obtaining a relation between thrust and PWM, more reliable autonomous tasks may be achieved.
- It should be noted that updating hardware such as IMUs, GNSS receiver, compass, autopilot cards and their software will improve future experimental results. It is expected that enhancing the vehicle with extra positioning sensors makes a significant contribution to update state information.

More complex autonomous tasks can be performed with many USVs. In these days, a collaboration between many unmanned vehicles becomes a hot topic in the control area. These works could be achieved via proven hardware and software, such as ROS (Robot Operating System), MAVROS (Micro Aerial Vehicle Robot Operating System), Dronecode SDK (Software Development Kit) etc.

REFERENCES

[1] Leightweight, dependency free matrix library. https://github.com/PX4/Matrix, 2018. [Online; accessed 25-July-2018].

[2] Pixhawk 1 flight controller. https://docs.px4.io/en/flight_controller/pixhawk.html, 2018. [Online; accessed 21-March-2018].

[3] International regulations for prevention of collisions at sea. http://www.jag.navy.mil/distrib/instructions/COLREG-1972.pdf, 22 September 2009. [Online; accessed 13-October-2018].

[4] K. Ahıska. Control and guidance of an unmanned sea surface vehicle. Master's thesis, Middle East Technical University, September 2012.

[5] M. Azzeri, F. Adnan, and M. Zain. Review of course keeping control system for unmanned surface vehicle. *Jurnal Teknologi*, 74(5):11–20, 2015.

[6] M. S. Bazaraa, H. D. Sherali, and C. Shetty. *Nonlinear Programming: Theory and Algorithms*. 2006.

[7] S. P. Berge and T. I. Fossen. On the Properties of the Nonlinear Ship Equations of Motion. *Mathematical and Compuer Modelling of Dynamical Systems*, 6(4):365–381, 2000.

[8] M. Bibuli, M. Caccia, L. Lapierre, and G. Bruzzone. Guidance of unmanned surface vehicles: Experiments in vehicle following. *IEEE Robotics Automation Magazine*, 19(3):92–102, Sep. 2012.

[9] M. Breivik. Nonlinear maneuvering control of underactuated ships. Master's thesis, Norwegian University of Science and Technology, June 2003.

[10] M. Breivik, V. E. Hovstein, and T. I. Fossen. Straight-Line Target Tracking for Unmanned Surface Vehicles. *Modeling, Identification and Control*, 29(4):131–149, 2008.

[11] A. W. Browning. *A mathematical model to simulate small boat behaviour*. PhD thesis, Bournemouth Polytechnic (University), October 1990.

[12] M. Caccia, M. Bibuli, R. Bono, and G. Bruzzone. Basic navigation, guidance and control of an unmanned surface vehicle. *Autonomous Robots*, 25(4):349–365, Nov 2008.

[13] S. Campbell, W. Naeem, and G. Irwin. A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres. *Annual Reviews in Control*, 36(2):267 – 283, 2012.

[14] I. Ç. Yılmaz, K. Ahıska, and M. K. Leblebicioğlu. Parallel docking problem for unmanned surface vehicles. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 744–749, Nov 2018.

[15] J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley series in electrical and computer engineering: control engineering. Pearson-/Prentice Hall, 1989.

[16] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43, Oct 1995.

[17] I. K. Erünsal. System Identification and Control of a Sea Surface Vehicle. Master's thesis, Middle East Technical University, September 2015.

[18] I. K. Erunsal, K. Ahıska, M. Kumru, and M. K. Leblebicioğlu. An approach for system identification in unmanned surface vehicles. In *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, pages 194–199, Oct 2017.

[19] T. Fossen and J. Gravdahl. *Marine Control Systems: Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles*. Marine Cybernetics, 2002.

[20] T. I. Fossen. *Nonlinear modelling and control of underwater vehicles*. PhD thesis, Norwegian Institute and Technology, June 1991.

[21] T. I. Fossen. Guidance and Control of Ocean Vehicles, 1994.

[22] T. I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. 2011.

[23] T. I. Fossen, M. Breivik, and R. Skjetne. Line-of-sight path following of underactuated marine craft. *IFAC Proceedings Volumes*, 36(21):211 – 216, 2003. 6th IFAC Conference on Manoeuvring and Control of Marine Craft (MCMC 2003), Girona, Spain, 17-19 September, 1997.

[24] T. I. Fossen and O.-E. Fjellstad. Nonlinear modelling of marine vehicles in 6 degrees of freedom. *Mathematical and Computer Modelling of Dynamical Systems*, 1(1):17–27, 1995.

[25] P. D. Groves. *Principles of GNSS Inertial and Multi-Sensor Integrated Navigation Systems - GNSS Technology and Applications*. 2008.

[26] F. Haugen. *Advance Dynamics and Control*. 2010.

[27] F. H. Imlay. The complete expressions for added mass of a rigid body moving in an ideal fluid.

[28] K. D. Do, J. P. *Control of Ships and Underwater Vehicles*. 2009.

[29] M. Kumru. Navigation and Control of an Unmanned Sea Surface Vehicle. Master's thesis, Middle East Technical University, September 2015.

[30] M. Kumru, M. K. Leblebicioğlu, I. K. Erünsal, and K. Ahıska. A survey on tactical control algorithms for path tracking unmanned surface vehicles. In *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1–6, Nov 2016.

[31] Z. Li and J. Sun. Disturbance compensating model predictive control with application to ship heading control. *IEEE Transactions on Control Systems Technology*, 20(1):257–265, Jan 2012.

[32] Z. Liu, Y. Zhang, X. Yu, and C. Yuan. Unmanned surface vehicles: An overview of developments and challenges. *Annual Reviews in Control*, 4:71 – 93, 2016.

[33] J. E. Manley, A. Marsh, W. Cornforth, and C. Wiseman. Evolution of the autonomous surface craft autocat. In *OCEANS 2000 MTS/IEEE Conference and*

Exhibition. Conference Proceedings (Cat. No.00CH37158), volume 1, pages 403–408 vol.1, Sep. 2000.

[34] MATLAB. Genetic algorithm. `https://www.mathworks.com/help/gads/genetic-algorithm.html`, 2018. [Online; accessed 13-September-2018].

[35] L. Meier, D. Honegger, and M. Pollefeys. Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6235–6240, May 2015.

[36] A. Motwani, S. Sharma, R. Sutton, and P. Culverhouse. Interval kalman filtering in navigation system design for an uninhabited surface vehicle. *Journal of Navigation*, 66:639–652, 09 2013.

[37] N.A. Shneydor. *Missile Guidance and Pursuit: Kinematics, Dynamics and Control*. 1998.

[38] W. Naeem, R. Sutton, and T. Xu. An integrated multi-sensor data fusion algorithm and autopilot implementation in an uninhabited surface craft. *Ocean Engineering*, 39:43–52, 2012.

[39] N. H. Norrbin. Theory and observations on the use of a mathematical model for ship manoeuvring in deep and confined waters. *Proceedings of the 8th Symposium on Naval Hydrodynamics*, page 123, 01 1971.

[40] S.-R. Oh and J. Sun. Path following of underactuated marine surface vessels using line-of-sight based model predictive control. *Ocean Engineering*, 37(2):289–295, 2010.

[41] D. Pearson, E. An, M. Dhanak, K. von Ellenrieder, and P. Beaujean. High-level fuzzy logic guidance system for an unmanned surface vehicle (usv) tasked to perform autonomous launch and recovery (alr) of an autonomous underwater vehicle (auv). In *2014 IEEE/OES Autonomous Underwater Vehicles (AUV)*, pages 1–15, Oct 2014.

[42] A. Pereira, J. Das, and G. S. Sukhatme. An experimental study of station keeping on an underactuated asv. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3164–3171, Sep. 2008.

[43] A. M. Rothblum. Human error and marine safety. *in US Coastguard Research and Development Centre*, pages 1 – 10.

[44] S. Sagatun and T. Fossen. Lagrangian formulation of underwater vehicles' dynamics. *Conference Proceedings 1991 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1–6, 1991.

[45] D. Simon. *Evolutionary Optimization Algorithms*. Wiley, 2013.

[46] R. Skjetne, T. I. Fossen, and P. V. Kokotović. Robust output maneuvering for a class of nonlinear systems. *Automatica*, 40(3):373–383, March 2004.

[47] R. Skjetne, O. N. Smogeli, and T. I. Fossen. A Nonlinear Ship Manoeuvering Model: Identification and adaptive control with experiments for a model ship. *Modeling, Identification and Control*, 25(1):3–27, 2004.

[48] SNAME The Society of Naval Architects and Marine Engineers. Nomenclature for treating the motion of a submerged body through a fluid, 1950.

[49] K. J. Åström and T. Hägglund. *Advanced PID Control*. ISA - The Instrumentation, Systems and Automation Society, 2006.

[50] S. Upadhyay and A. Ratnoo. A point-to-ray framework for generating smooth parallel parking maneuvers. *IEEE Robotics and Automation Letters*, 3(2):1268–1275, April 2018.

[51] K. P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru, and G. A. Demetriou. Control architectures for autonomous underwater vehicles. *IEEE Control Systems Magazine*, 17(6):48–64, Dec 1997.

[52] A. Visioli. *Practical PID Control*. Advances in Industrial Control. Springer London, 2006.

[53] L. Wang. *Model Predictive Control System Design and Implementation Using MATLAB*. 2011.

[54] Wikipedia. atan2. https://en.wikipedia.org/wiki/Atan2, 2018. [Online; accessed 21-March-2018].

[55] H. Zheng, R. R. Negenborn, and G. Lodewijks. Trajectory tracking of autonomous vessels using model predictive control. *IFAC Proceedings Volumes*, 47(3):8812 – 8818, 2014. 19th IFAC World Congress.

APPENDIX A

GUIDANCE DERIVATION

A.1 Calculation of $\boldsymbol{p}_{LOS}$ From LOS Equations

To be able to found an analytic solution for $\boldsymbol{p}_{LOS}$, distance in x axis, $\Delta x = x_w - x_{w-1}$, and y axis, $\Delta y = y_w - y_{w-1}$, of two consecutive way point $\boldsymbol{p}_w$ and $\boldsymbol{p}_{w-1}$ must be known [9]. Solution includes two cases according to Δx which can be either $\Delta x = 0$ or $|\Delta x| > 0$.

A.1.1 Case 1: $\Delta x = 0$

In this case, slope of the straight line between $\boldsymbol{p}_{w-1}$ and $\boldsymbol{p}_w$ becomes infinity. From this condition it can be written as $x_{LOS} = x_w = x_{w-1}$. (3.4) is invalid due to the fraction term. Then, (3.4) is rewritten to solve $y_{LOS}[k]$ as follows:

$$(y_{LOS}[k] - y[k])^2 = (nL_{pp})^2. \tag{A.1}$$

(A.1) results in $y_{LOS}[k] = y[k] \pm nL_{pp}$. For forward motion, selection of $y_{LOS}[k]$ has two criteria,

- If $\Delta y > 0$, then $y_{LOS}[k] = y[k] + nL_{pp}$.
- If $\Delta y < 0$, $y_{LOS}[k] = y[k] - nL_{pp}$.

For backward motion,

- If $\Delta y > 0$, then $y_{LOS}[k] = y[k] - nL_{pp}$.
- If $\Delta y < 0$, $y_{LOS}[k] = y[k] + nL_{pp}$.

A.1.2 Case 2: $\Delta x > 0$

Linear algebraic equation of the straight line between two sequential way-points is written as

$$y_{LOS}[k] = (\frac{\Delta y}{\Delta x})(x_{LOS}[k] - x_{w-1}) + y_{w-1} \tag{A.2}$$

or

$$y_{LOS}[k] = (\frac{\Delta y}{\Delta x})(x_{LOS}[k] - x_{w}) + y_{w}. \tag{A.3}$$

From now on, slope of above line equations is denoted by, $d = \dfrac{\Delta y}{\Delta x}$, x and y axis components are simply represented by $e = x_{w-1}$ and $f = y_{w-1}$, respectively. Using these relation, square of (A.2) is written

$$\begin{aligned} y^2_{LOS}[k] &= ((\frac{\Delta y}{\Delta x})(x_{LOS}[k] - x_{w-1}) + y_{w-1})^2 \\ &= (dx_{LOS}[k] + (f - de))^2 \\ &= (dx_{LOS}[k] + g)^2 \\ &= d^2x^2_{LOS}[k] + 2dgx_{LOS}[k] + g^2 \end{aligned} \tag{A.4}$$

where g is intended to simply show $(f - de)$ in above equation. Now other relation for $y_{LOS}[k]^2$ comes from (3.3) by explicitly writing it:

$$\begin{aligned} (x_{LOS}[k] - x[k])^2 + (y_{LOS}[k] - y[k])^2 &= x^2_{LOS}[k] - 2x_{LOS}[k]x[k] + x^2[k] \\ + y^2_{LOS}[k] - 2y_{LOS}[k]y[k] + y^2[k] &= (nL_{pp})^2 \end{aligned} \tag{A.5}$$

By rewriting the term $2y_{LOS}[k]y[k] = 2y[k](dx_{LOS}[k] + g)$ and using the above (A.4) and (A.5), second order hyperbolic equation is stated with only one unknown which is $x_{LOS}[k]$, [9]:

$$\begin{aligned} (1 + d^2)x^2_{LOS}[k] + 2(dg - dy - x[k])x_{LOS}[k] \\ + x^2[k] + y^2[k] + g^2 - (nL_{pp})^2 - 2gy[k] = 0 \end{aligned} \tag{A.6}$$

Coefficients of $ax^2_{LOS}[k] + bx_{LOS}[k] + c = 0$ is represented by

$$\begin{aligned} a &= 1 + d^2 \\ b &= 2(dg - dy - x[k]) \\ c &= x^2[k] + y^2[k] + g^2 - (nL_{pp})^2 - 2gy[k] \end{aligned}$$

Solution of (A.6) is written in well-known from as

$$x_{LOS}[k] = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \tag{A.7}$$

To find a point on line segment between two sequential way-points, $\sqrt{b^2 - 4ac} > 0$ must be satisfied. This is only provided choosing a reasonable nL_{pp} value. If solution is not exist, then n is carefully increased until finding valid solution. There exists two possible solutions for forward motion:

- If $\Delta x > 0$, then $x_{LOS}[k] = \dfrac{-b + \sqrt{b^2 - 4ac}}{2a}$.
- If $\Delta x < 0$, then $x_{LOS}[k] = \dfrac{-b - \sqrt{b^2 - 4ac}}{2a}$.

$x_{LOS}[k]$ is obtained for backward motion

- If $\Delta x > 0$, then $x_{LOS}[k] = \dfrac{-b - \sqrt{b^2 - 4ac}}{2a}$.
- If $\Delta x < 0$, then $x_{LOS}[k] = \dfrac{-b + \sqrt{b^2 - 4ac}}{2a}$.

After $x_{LOS}[k]$ is obtained from above parabolic (or second order polynomial) equation, $y_{LOS}[k]$ can be simply obtained from (A.2) or (A.5).

A.2 Computation of Continuous Reference Yaw Angle

As it can be seen in Fig. A.1, $\mathrm{atan2}(\ ,\ )$ takes Δx and Δy which are defined previous section and gives yaw angle of interested LOS guidance method. For PP guidance, it is defined that $\Delta x = x_w - x[k]$ and $\Delta y = y_w - y[k]$, $\Delta x = x[k] - x_w]$ and $\Delta y = y[k] - y_w$ for backward motion. $\psi_{now} = \mathrm{atan2}(\Delta y, \Delta x)$ represent the LOS or PP angle in time step $[k]$. Then mapping from $[-\pi, \pi]$ to $(-\infty, \infty)$ is handled. Mapping offers a solution to discontinuity issue at the $-\pi$ or π junction. To be able to make mapping, extra variables $state$ and ψ_{last} are needed. $state$ is the information where yaw angle produced by guidance law at previous sampling time. It can be thought that $state$ keeps the information of $\mathrm{atan2}(\ ,\ )$ function by utilizing one of the four values described in unity circle in Fig. A.2.

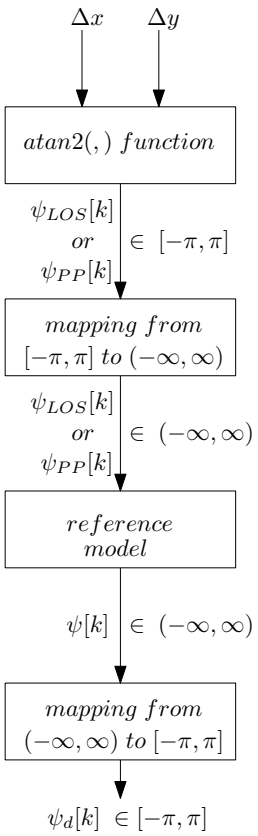

Figure A.1: Process for obtaining continuous desired yaw angle.

For mapped desired guidance angle, last variable $accumulation$ which represents the changes from beginning of motion is also be needed. So, reference model can be fed via continuous angle. It is worthy of note that, all these defined variables for continuous mapping must be taken as zero when control loop starts. The Guide Rule computes the $accumulate$ variables that indicates angular change from ψ_{last} to ψ_{now} in each sampling time [9]. It should be kept in mind that following relation, $accumulate = \psi_{now} - \psi_{last}$, is not always valid due to the $-\pi/\pi$ discontinuity. So, memory variable $accumulation$ needs to preserves angle information when jumps occur. This relation is formulated as $accumulation = accumulation + accumulate$.

Unity circle defined in A.2 is divided into four quadrant how to determine $state$ variable which takes value from 1 to 4. It means that results $\mathrm{atan2}(\ ,\ )$ function has a relation with corresponding state. Furthermore, each $state$ case is divided into four instances due to the changes from ψ_{last} to ψ_{now}. So there are 16 possible and 1 true $accumulation$ calculation to update it in each sampling time. These 16 possible up-

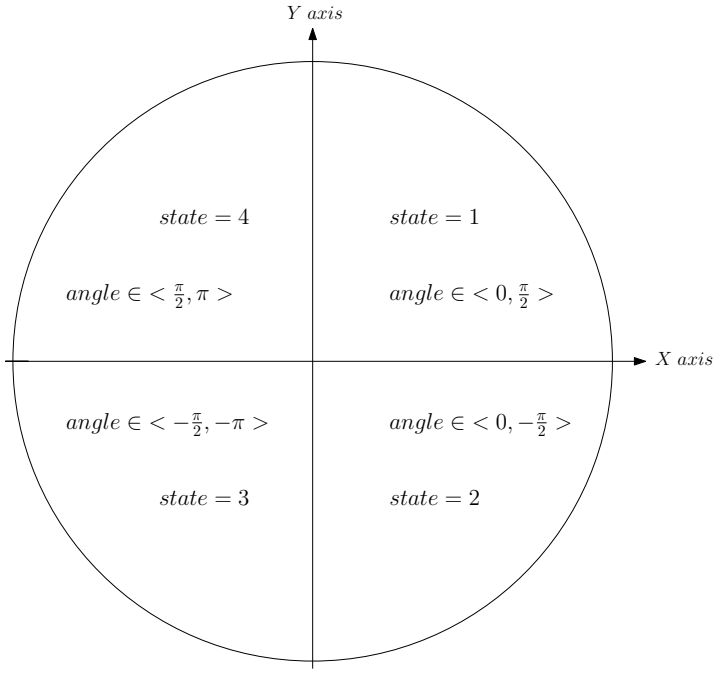

Figure A.2: $state$ and $angle$ information on unity circle

dates are listed below from $state = 1$ to $state = 4$ by using Table A.1.

- if($current\_state == 1$) which means $\text{sgn}(\Delta y) = 1$ and $\text{sgn}(\Delta x) = 1$
  - if($prev\_state == 1$ *or* $prev\_state == 2$ *or* $prev\_state == 4$)
    - $accumulate = \psi_{now} - \psi_{last}$
  - else if($prev\_state == 3$)
    - if($(\psi_{now} + abs(\psi_{last})) \leq \pi$)
      - $accumulate = \psi_{now} - \psi_{last}$
    - else
      - $accumulate = (\frac{\pi}{2} - \psi_{last}) + \frac{\pi}{2} + (\pi + \psi_{now})$
        $= \psi_{now} - \psi_{last} + 2\pi$
    - end
  - end

  $prev\_state = 1$

- else if($current\_state == 2$) which means $\text{sgn}(\Delta y) = -1$ and $\text{sgn}(\Delta x) = 1$
  - if($prev\_state == 1$ *or* $prev\_state == 2$ *or* $prev\_state == 3$)
    - $accumulate = \psi_{now} - \psi_{last}$
  - else if($prev\_state == 4$)

Table A.1: Relation between $state$ and unity circle quadrants by using Δx and Δy.

$\mathrm{sgn}(\Delta y)$	$\mathrm{sgn}(\Delta x)$	Quadrant
1	1	1
-1	1	2
-1	-1	3
1	-1	4

if$((abs(\psi_{now}) + \psi_{last}) \leq \pi)$

$accumulate = \psi_{now} - \psi_{last}$

else

$$accumulate = (\frac{\pi}{2} + \psi_{last}) + \frac{\pi}{2} + (\pi - \psi_{now})$$
$$= \psi_{last} - \psi_{now} + 2\pi$$

end

end

$prev\_state = 2$

- else if$(current_state == 3)$ which means $\mathrm{sgn}(\Delta y) = -1$ and $\mathrm{sgn}(\Delta x) = -1$

if$(prev_state == 2$ *or* $prev\_state == 3)$

$accumulate = \psi_{now} - \psi_{last}$

else if$(prev_state == 4)$

$$accumulate = -(\pi + \psi_{last}) - (\pi - \psi_{now})$$
$$= \psi_{now} - \psi_{last} - 2\pi$$

else

if$((abs(\psi_{now}) + \pi_{last}) \leq \pi)$

$accumulate = \psi_{now} - \psi_{last}$

else

$$accumulate = (\pi + \psi_{last}) + \frac{\pi}{2} + (\frac{\pi}{2} - \psi_{now})$$
$$= \psi_{last} - \psi_{now} + 2\pi$$

end

end

$prev\_state = 3$

- else if($current\_state == 4$) which means $\mathrm{sgn}(\Delta y) = 1$ and $\mathrm{sgn}(\Delta x) = -1$

  if($prev\_state == 1$ *or* $prev\_state == 4$)

  $accumulate = \psi_{now} - \psi_{last}$

  else if($prev\_state == 2$)

  if($abs(\psi_{now}) + \pi_{last}) \leq \pi$)

  $accumulate = \psi_{now} - \psi_{last}$

  else

  $$accumulate = (\pi - \psi_{last}) + \frac{\pi}{2} + (\frac{\pi}{2} + \psi_{now})$$
  $$= \psi_{now} - \psi_{last} + 2\pi$$

  end

  else

  $$accumulate = (\pi - \psi_{last}) + (\pi + \psi_{now})$$
  $$= \psi_{now} - \psi_{last} + 2\pi$$

  end

  $prev\_state = 4$

  end

  $accumulation \leftarrow accumulation + accumulate$

  $(prev\_state \leftarrow current\_state)$

  $(\psi_{last} \leftarrow \psi_{now})$

Only one $accumulation$ value is calculated from 16 possible equations according to Δy, Δx and $prev\_state$ value, i.e. $accumulation = accumulation + accumulate$. Previous state, $prev\_state$, variable is updated with current state, $current\_state$, variable for next sampling time, $(prev\_state \leftarrow current\_state)$. In order to track angle change, calculated yaw angle from $\mathrm{atan2}(\ ,\ )$ function is also updated: $(\psi_{last} \leftarrow \psi_{now})$. At the end, mapping from $[-\pi,\ \pi]$ to $(-\infty, \infty)$ is obtained.

Continuous mapped reference yaw angle, which is obtained as $accumalation$ variable, is filtered to limit this angle in the vehicle heading dynamics by using nonlinear (reference) model. This angle is also filtered through reference model to get rid of higher order derivatives (high frequency components) of the reference signal [19]. So, desired reference angle in continuous domain, $\psi[k] \in (-\infty, \infty)$, is handled.

In the last step, continuous filtered desired yaw angle at time k, $\psi[k]$ should be

mapped back from $(-\infty, \infty)$ to discontinuous region which is defined in $(-\pi, \pi)$. So angle information is transformed from high information region to lower information region. There is no need to utilize state and memory variables in this instance. Reverse mapping algorithm is fulfilled in two steps as follows. First integer number, n, is obtained, later mapped yaw angle, $\psi_d[k] \in (-\pi, \pi)$, is calculated by using this number.

- Determination of number of 2π for continuous desired yaw angle, $\psi[k]$

  if $(\text{sgn}(\psi[k]) > 0)$

  $$n = \left\lfloor \frac{\psi[k]}{\pi} \right\rfloor$$ /* floor function calculation */

  else if$(\text{sgn}(\psi[k]) < 0)$

  $$n = \left\lceil \frac{\psi[k]}{\pi} \right\rceil$$ /* ceil function calculation */

  else

  $n = 0$

  end

  $remainder = n \pmod 2$ /* modulo operation of n */

- Determination of $\psi_d[k]$ variable

  if $(remainder == 0)$

  $\psi_d[k] = \psi[k] - n * \pi$

  else /* $remainder$ is not equal to zero*/

  if $(\text{sgn}(remainder) > 0)$

  $\psi_d[k] = \psi[k] - (n+1) * \pi$

  else /* sign of $remainder$ is smaller than zero*/

  $\psi_d[k] = \psi[k] - (n-1) * \pi$

  end

  end

So, continuous yaw angle, $\psi[k]$, is mapped to discontinuous desired yaw angle $\psi_d[k] \in [-\pi, \pi]$. $\psi_d[k]$ has importance for simulation and real-time implementation because of the fact that yaw angle is generally calculated in the region of $[-\pi, \pi]$.