National Technical University of Athens

Master Thesis

Dynamic Positioning of an Emergency Recovery and Rescue Vessel using Neural Networks

Author:
Nikolaos Kougiatsos

Supervisor:
Assistant Professor George Papalambrou

Committee members:
Professor G. Grigoropoulos
Professor G. Zarafonitis
Laboratory of Marine Engineering

School of Naval Architecture and Marine Engineering

July 18, 2020

Acknowledgements

By writing this thesis, my studies at the School of Naval Architecture and Marine Engineering are coming to an end. The journey so far was pleasant with a lot of interesting opportunities coming my way constantly. I also caught knowledge besides the scope of Naval Architecture both in hardware and software due to the abundance of student groups in the University, the offering of similar subjects as well as interesting projects.

The first person I would like to thank would be my supervisor, professor G.Papalambrou who supported me and my efforts ever since our first encounter in the second semester's subject FORTRAN. I also had the chance to work for the Laboratory of Marine Engineering the following summer contributing to the present air heater's modernization. It was at that point that I was given the chance to learn more about programming and hardware in an effort to achieve the best possible final result.

A special thanks must be attributed to the team of Supertoys Inc which were responsible for keeping the boat in storage and accompanied us in the sea trials. Without them, the scope of this thesis would seem shallow and ordinary.

I would also like to dedicate this thesis to my family who supported me up to this point and especially my sister who made me want to be a better version of myself.

Abstract

This Thesis examines the feasibility of implementing Dynamic Positioning upon an EER Vessel with no blades or potential harmful gear, using a Reinforcement Learning Approach. The work described in this Thesis can be divided into two main tasks. The first one is the derivation of a network architecture capable of controlling the vessel and the second one is the creation of a simulation environment capable of training the network to achieve the expected results.

The Neural Network is created using Tensorflow and the Keras API while the simulation environment is created in the Gym Python Module and also incorporates a graphical interface. A GUI has also been created to control the vessel at sea both manually and autonomously using the trained network itself. In addition, the calibration and fusion of the different sensors to acquire as precise measurements as possible is discussed in the following Chapters.

Περίληψη

Στην παρούσα διπλωματική εργασία εξετάζεται η εφικτότητα της εφαρμογής ενός συστήματος δυναμικού ελέγχου της θέσης ενός πλοίου διάσωσης χωρίς πτερύγια ή πιθανώς επικίνδυνο εξοπλισμό, με χρήση Νευρωνικού δικτύου ενισχυμένης εκμάθησης. Η δουλειά που περιγράφεται στην παρούσα διπλωματική μπορεί να διαιρεθεί σε δύο βασικά κομμάτια. Το πρώτο από αυτά είναι η εξαγωγή μιας αρχιτεκτονικής δικτύου η οποία να είναι κατάλληλη για τον έλεγχο του σκάφους και το δεύτερο είναι η δημιουργία ενός περιβάλλοντος προσομοίωσης ικανό να προπονήσει το δίκτυο για να επιτύχει τα αναμενόμενα αποτελέσματα.

Το Νευρωνικό δίκτυο δημιουργείται με τη βοήθεια του λογισμικού Tensorflow και της μικροεφαρμογής Keras ενώ το περιβάλλον προσωμοίωσης δημιουργείται με τη βοήθεια του πακέτου Gym για την Python και περιλαμβάνει το δικό του γραφικό περιβάλλον. Ένα γραφικό περιβάλλον αλληλεπίδρασης με το χρήστη έχει επίσης δημιουργηθεί για τον έλεγχο του σκάφους στη θάλασσα τόσο χειροκίνητα όσο και αυτόματα με τη χρήση του ίδιου του προπονημένου δικτύου. Επιπρόσθετα, η βαθμονόμηση και ο συνδυασμός των διαφορετικών αισθητήρων για την απόκτηση όσο το δυνατόν πιο ακριβών μετρήσεων συζητάται στα επόμενα καφάλαια

Contents

List of Figures

List of Tables

List of Abbreviations

I^2**C**	**I**nter-**I**ntegrated **C**ircuit
CAD	**C**omputer **A**ided **D**esign
DOF	**D**egrees **O**f **F**reedom
DP	**D**ynamic **P**ositioning
DQN	**D**eep **Q**-**N**etwork
ECEF	**E**arth **C**entered **E**earth **E**earth **F**ixed
ECI	**E**arth **C**entered **I**nertial
GPS	**G**lobal **P**ositioning **S**ystem
GUI	**G**raphical **U**ser **I**nterface
IMU	**I**nertial **M**easurement **U**nit
MEMS	**M**icro**e**lectro**m**echanical **S**ystems
NED	**N**orth-**E**ast-**D**own
NN	**N**eural **N**etwork
PCB	**P**rinted **C**ircuit **B**oard
SCL	**S**erial **Cl**ock
SDA	**S**erial **Da**ta
SPI	**S**erial **P**eriferal **I**nterface

1 Introduction

1.1 Motivation and History

Dynamic Positioning was first inferred as a concept in the novel ”Propeller Island” by Jules Verne, which was first published in 1895. The story follows a group of people living on an artificial island named ”Standard Island”, located in the Pacific. The island is able to float without being stranded by anchors in an area where the depth of water is too large, just by using machines ”which maneuver ahead and astern throughout its stay”.

Figure 1.1: Propeller Island by Jules Verne (1985)

The first real world debut of such a system was made in the 1960s in the industry of offshore drilling. As the drilling was performed in deep water, using an anchor for maintaining the position was considered not economical. One year after, in 1961, a DP system was fitted onto the vessel Cuss 1 that employed four azimuth thrusters of 147 kW each, one fitted in each corner of the vessel. The control scheme used in this case was manual and performed by two operators, whereas the position of the vessel was estimated by visually inpecting four surface buoys delineating a radius of 180 m and a solar screen displaying the vessel’s relative position to four deepwater buoys.

The first truly DP vessel in the world was an 1961 drillship named Eureka which was intended for deepwater coring. Although the vessel was initially destined to be controlled manually, much like CUSS 1, later an automatic control system was suggested by Howard Shatto[1]. Eureka was fitted with two azimuth thrusters of 441 kW total and its control system featured a PID controller for each horizontal degree of

[1]Electical Engineer. He is often inferred to as the ”Father of Dynamic Positioning”

freedom and a taut wire position reference. The control unit received the codename ”APE” and was the first to implement thrust allocation functionality at sea.

FIGURE 1.2: Eureka Ship(1961)

In 1968, the deepwater coring vessel Glomar Challenger was launched with a size significantly larger than that of CUSS 1 and Eureka. In this case, a more sophisticated digital control system was used for DP which was the first to utilize wave filtering to remove unwated 1st order wave-induced motion. It also used a more complex type of position reference which incorporated short baseline (SBL), hydroacoustics, with one transmitter and three hydrophones below the hull and one transpoder on the the seabed.

DP technology continued to develop in the following years with another famous millestone being ”SEDCO 445” in 1971 which was equipped with a Honeywell Automatic Station Keeping System. In 1980, there were about 65 DP capable vessels operating in offshore operations. This number boomed to 150 by 1985 as Satellite Technology was incorporated in the field. In 1990, DPVOA[2] was formed with the responsibility to form quidelines for all DP vessels constructed after 1st of July 1984. DPVOA later merged with the International Association of Diving Contractors in 1995, forming the IMCA [3], the main current DP regulatory body. Up until the present, over 50 guidance documents have been released by IMCA, based on information gathering over the years.

The field of DP remains rather promising even in recent years, especially as a fundamental part in the development of tomorrow’s autonomous vessels. Notorious projects that are currently under development include the partnership of Google with Roll’s Royce to bring an autonoous vessel by 2020, able to transport either passengers or goods using Google’s Cloud Machine Learning Engine to detect, identify and track obstacles and objects the vessel might encounter and ASV’s C-Worker 7 which is

[2]Dynamic Positioning Vessel Owners Association
[3]International Marine Contractors Association

powered by 2 Diesel engines and its main purpose is positioning, surveying and environmental monitoring without the need of a ship on station or sea-bed anchoring.

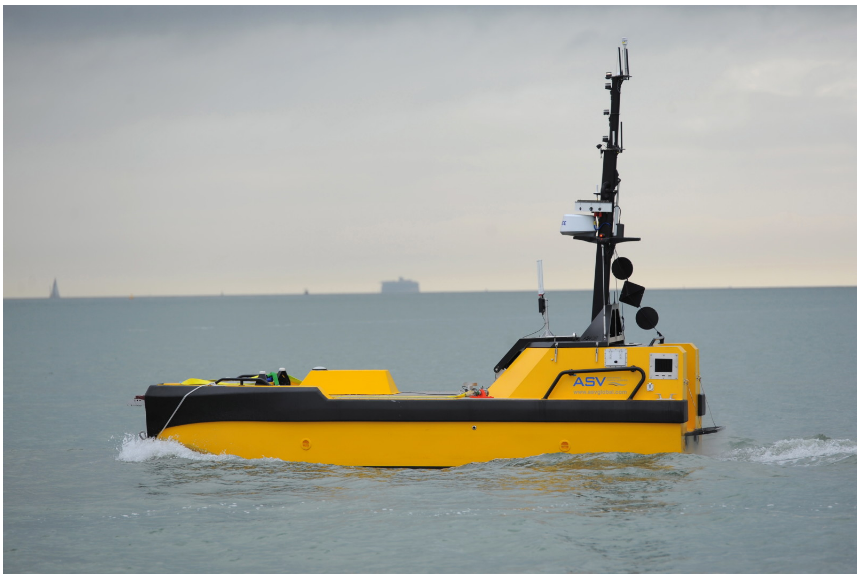


FIGURE 1.3: ASV's C-Worker 7

The low cost of electronic hardware, the vast availability of software and informational resources as well as the trending nature of the topic are the main motivators for this thesis. After all, the field of Autonomous Vessels and Dynamic Positioning presents huge future prospects that make worthwhile to try and explore different variations of the problem and reach new millestones.

1.2 Literature Review

Extensive research has been conducted over the years concerning the use of Neural Networks in DP operations. More specifically, in 1993 Gu, Pao and Yip [18] proposed the use of a neural network without any hidden layers for the estimation of wave drift loads on the ship by measurements of wave motions. For this reason, a clover buoy was used to determine the wave direction, a water probe ro measure the wave elevation at the bow of the vessel and an acoustic ranging system was employed to monitor the trajectory of the ship's drift motion relative to the centroid of the fixed underwater acoustic transmitter array. The study was a success as verified by simulations. Three years later, Li and Gu [4] investigated the use of a multiple policy recurrent neural network that takes the short time histories of the applied control forces and of the vessel's mean positions to perform the dynamic positioning. In this regard, the objective function for training the network weights could either be suited for track precision or for power saving by stepping up or down the actuation accordingly. The concept was indeed verified using computer simulations and accounting both for first-order wave frequency forces and second-order low frequency disturbances that lead to drift motions.

One other notable application has been that of Zhang, Hearn and Sen in 1996 [32] who attempted to use the back propagation algorithm of supervised learning in an online learning scheme without the need for a teacher to regulate both the sway and yaw motions using just the rudder. The aim was to design a controller without the need for knowing the vessel model with the use of a neural net that by nature is also able to predict nonlinear behaviour. The proposed architecture employed 1 input, 1 hidden and 1 output layer of 10,10 and 1 neuron accordingly and as for the disturbances, both a wind disturbance model by Isherwood [16] and a measurement noise for both degrees of freedom was incorporated to make the training phase more difficult. The accuracy of the controller was indeed confirmed by simulating straight line, zigzag and sine curve maneuvers. Later, in 1997, the same team investigated the use of neural networks for automatic a vessel's berthing procedure [31]. A MARAD series model has been employed for simulation purposes with a compilation of a gyro, a Doppler log, a sonar, a GPS and a TPS to monitor the vessel position and using the Isherwood model again to account for the wind loads. A network scheme of 3 layers was used with 11 inputs of data for all three planar degrees of freedom, 9 hidden neurons and 2 outputs, namely the necessary rudder angle and the appropriate propeller revolutions. The study was performed both for shallow and deep water environment and resulted in success with the main remarks being that there is no need for completely identifying the vessel model if modest qualitative knowledge is at hand and that route planning beforehand is extremely important.

In 1999, Unar, Murray and Smith [17] also testified for the feasibility of using neural networks for DP to account for variational operation and in turn used a radial basis functions network for ship steering control and a local model network for the representation of ship dynamics. Zirilli et al [3] in 2000, were preoccupied with the application of an adaptive autopilot for a containership to regulate four degrees of freedom in total using a reference model as a benchmark for the desired behaviour. Cao et al [34] on the other hand went for an auto regressive moving average (ARMA) functional-link neural network that proved to be both effective and fast for the dynamic positioning of offshore structures, as proven by numerical simulations assessing both accuracy and robustness.

One of the most recent attempts was that of Khizer, Yaping and Unar [2], who employed a multilayer feedforward network architecture of three layers in total for the design of a heading angle autopilot. MATLAB simulations using the neural network toolbox were conducted with no environmental disturbances incorporated into the model and their results were later compared according to certain standards. One main conclusion of the paper was that only one hidden layer is indeed sufficient for DP neural nets. Finally, Lin et al [33] in their paper were preoccupied with the devellopment of a neural network observer for DP under uncertain system dynamics and disturbances. The neural network in this case was used to produce noise-free estimates of positions and velocities from corrupted measurements, in an attempt to outperform and eventually replace nonlinear robust controllers that by definition require full state variable feedback control. The study concluded by the deduction that uncertainties can be efficiently learned by neural net approximation.

1.3 Problem Description

This thesis constitutes part of a large European-founded project, named LINCOLN (Lean Innovative Connected Vessels), which aims to develop added-value specialized vessels that are able to run requested services for marine sectors in the most effective, efficient, economic, valuable and eco-friendly way. Part of this project is to develop an autonomous DP system for an emergency response and rescue vessel (ERRV), which is shown in Figure 1.4. NTUA is collaborating with Supertoys Boats, a small boat constructor in Greece, who is one of the partners in the project, and the vessel will ultimately be used for further enhancing sea rescuing operations.

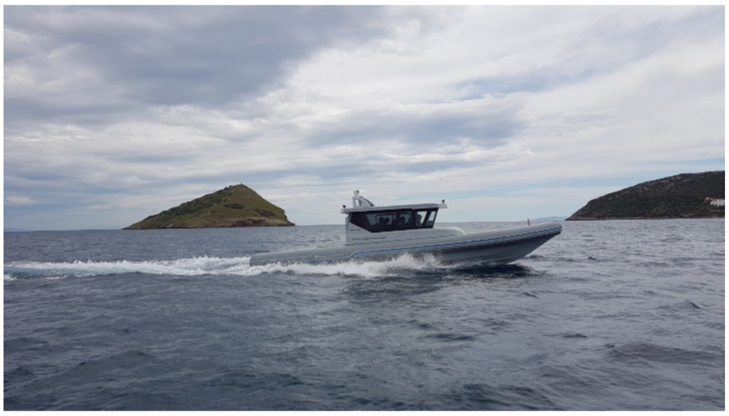


FIGURE 1.4: Lincoln's EER Vessel

1.4 Contributions

Based on the predefined project goal, the contributions of this thesis can be summarized by the following list:

- Derive a solid physical representation of the system that incorporates a model for environmental disturbances
- Define the control allocation scheme
- Train an NN to be able to control the real-world system
- Modify the NN parameters in order to be able to incorporate it in the specified electronics
- Provide a GUI application able to both manually and automously control the vessel's thrusters as well as display useful information for the user such as the vessel's position and speed
- Fine-tune the electronics to cooperate flawlessly with one another with no failures and minimal delays

1.5 Thesis Outline

- Chapter 2: This chapter discusses the vessel's thruster system setup as well as the hardware electronics and the software employed to carry out the DP purposes.
- Chapter 3: This chapter discusses the mathematical modelling of the physical system that is of concern in this thesis.
- Chapter 4: A proposal for evaluating the system noise is the main focus of this chapter. Noise is considered as anything that can disrupt the system equilibrium. Thus, the modelling of wave and wind effects will be discussed.
- Chapter 5: The purpose of this chapter is to investigate the calibration and the fusion of different sensors to minimize the measurement errors.

- Chapter 6: This chapter's focus will be to propose a neural network architecture to create an overall MIMO control system capable of managing the vessel in 2 planar degrees of freedom.
- Chapter 7: Simulation results will be presented in this chapter.
- Chapter 8: This chapter contains the thesis's conclusions and provide a prospective view of future work that could be carried out after its completion.

1.6 Notation for Marine Vehicles

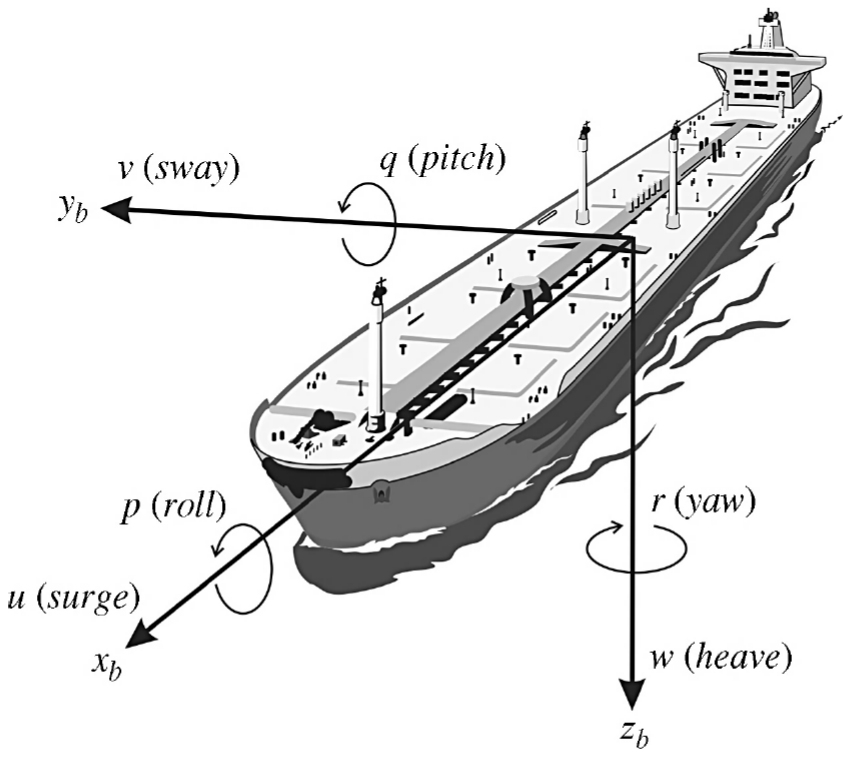


FIGURE 1.5: 6-DOF velocities in body-fixed reference frame

From this point forward, the following notation will be used to describe the kinematics and kinetics of the marine vehicle (see Table 1.1)

DOF	Description	Forces/Moments	Linear/Angular velocities	Positions/Euler angles
1	surge	X	u	x
2	sway	Y	v	y
3	heave	Z	w	z
4	roll	K	p	ϕ
5	pitch	M	q	θ
6	yaw	N	r	ψ

TABLE 1.1: Notation used for marine vehicles

2 System, software and hardware setup

2.1 System setup

The main requirement of the thrust system that will be implemented on the vessel is, as stated in the beginning, that there are no parts exposed, such as blades, which can be potentially harmful during the rescuing operations at sea. For this reason, it was decided to not equip the vessel with the standard bow thruster systems that are commonly used. Instead, a system comprised of a centrifugal water pump and water ejection nozzles will be tested. Due to the fact that the alternation of the way that thrust is produced changes the variables of the system and even the limitations of it, the proposed thrust system will be firstly implemented and evaluated on a small test boat, which has an overall length of 4.1 meters and is shown in Figure 2.1. The small boat is only being used for testing purposes but the final system will be installed on the large EERV boat shown in Chapter 1.

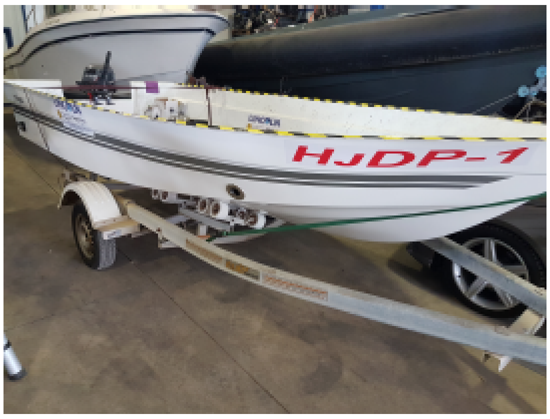


FIGURE 2.1: System Test Boat

2.1.1 Boat Dimensions

The HJDP-1 test-boat portrayed in Figure 2.1 was measured to have the following principal particulars recorded in Table 2.1

Dimension	Value
L	4.1 m
B	2 m
D	1 m
T	0.15 m
C_B	0.6

Table 2.1: HJDP-1 vessel dimensions

The LCG of the vessel is estimated to be 1.5 m from the transom, whereas its mass and yaw moment of inertia with regard to the LCG [7] were calcuted using the following standard expressions:

$$m[kg] = 1027 * L * B * T * C_B \tag{2.1a}$$
$$I_z^{LCG}[kg * m^2] = m * (0.25L)^2 - m * LCG^2 \tag{2.1b}$$

The inertia characteristics found in this manner are presented in the next Table 2.2.

Characteristic	Value
m	757.93 kg
I_z^{LCG}	1479.9 $kg * m^2$

Table 2.2: HJDP-1 inertial characteristics

An approximate virtual 3D model was also designed in Fusion 360, just in case further info were required. The model is diplayed in Figure 2.2

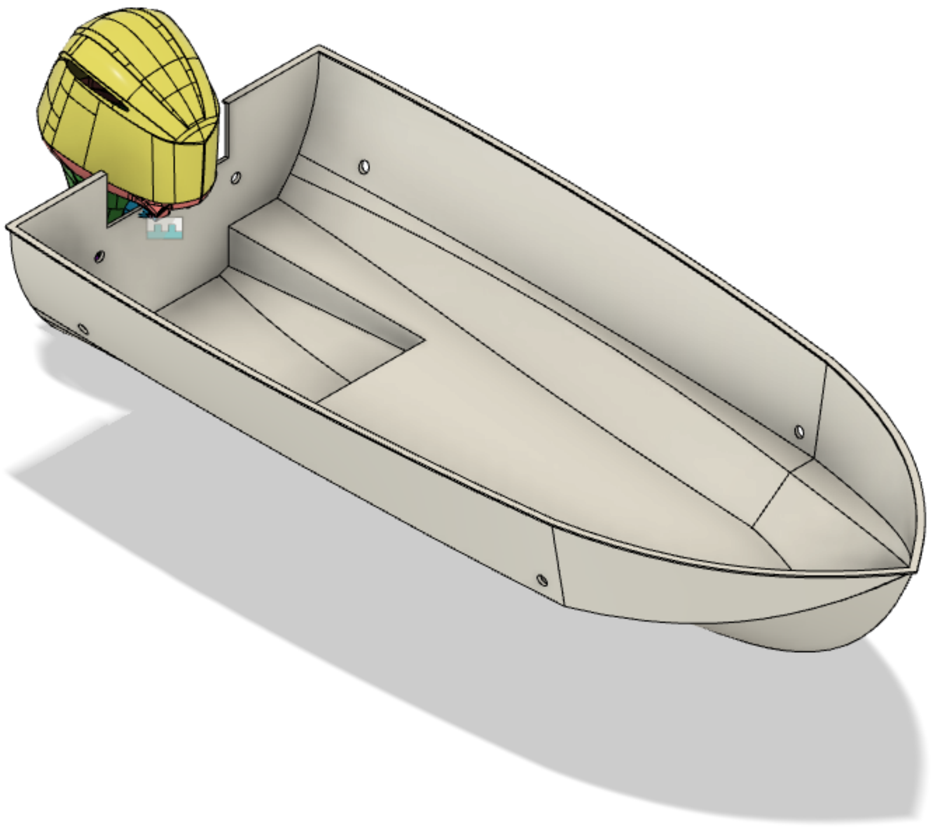


FIGURE 2.2: HJDP-1 vessel Fusion 360 3D Model

2.1.2 Thrust System

The water pump thrust system implemented is the Jet-Thruster JT-30 compact system manufactured by Holland Marine Parts [23]. The main system layout is shown in the next Figure 2.3.

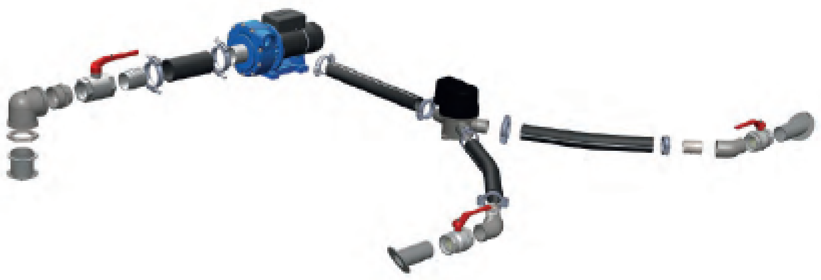


FIGURE 2.3: Exploded view of JT-30 compact system [13]

The set is composed of a centrifugal water pump paired with an electrical 3kW DC motor **(a)** which continuously supplies the piping network with water. The pump is powered using two batteries connected in parallel which also supply power to the other on-board electronics. The water exiting the pump is then split through a two-way outlet**(b)** to 3-way electrically controlled valves **(c)**. These valves are controlled using a controller **(d)** made by the manufacturer, which has been replaced for the project's purposes with an Arduino and RPI combination. Nozzles **(f)** fitted on the boundary of the hull constitute the system outflow regions where the outgoing flow results in an opposite equal resultant force acting upon the vessel. As a safety precaution, before each nozzle exit, ball-valves **(e)** are fitted and controlled by hand, in case there is a need to close a water exit. Standard low speed system bleeding has also been taken care of to reduce high water pressures due to the amassment of water that could potentially lead to failures.

The system is configured to be used in an on/off manner meaning that there is no control over the pump rotations, the exact position of the valves and as a result the magnitude of the control force.

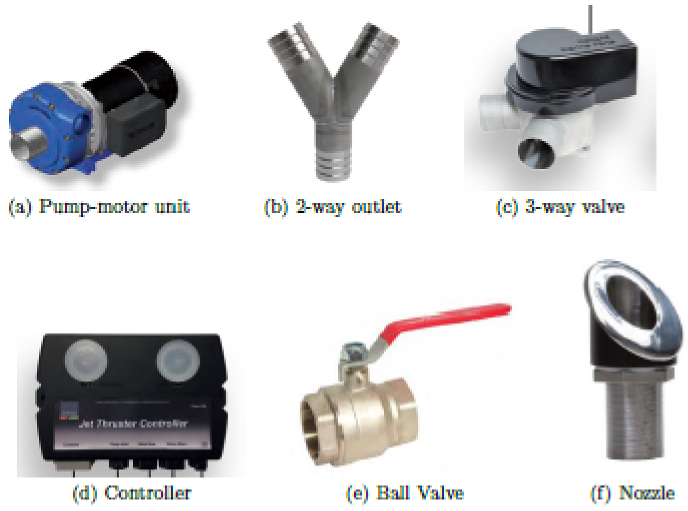


FIGURE 2.4: Jet-Thruster JT-30 system parts detailed [13]

The system configuration can be seen schematically in the next figure 2.5. It is evident that the system is underactuated by nature in the surge direction as backwards motion would require other maneuvers such as rotating the ship by 90 degrees, going left for the needed distance and again rotating the ship by -90 degrees to attain the initial posture. As this problem also involves course-keeping instead of just position-keeping, the surge motion will not be actuated by the proposed neural network. Let's also note that as one can deduce from the schematic representation, it is not possible to simultaneously employ clockwise or counter-clockwise and sway motion.

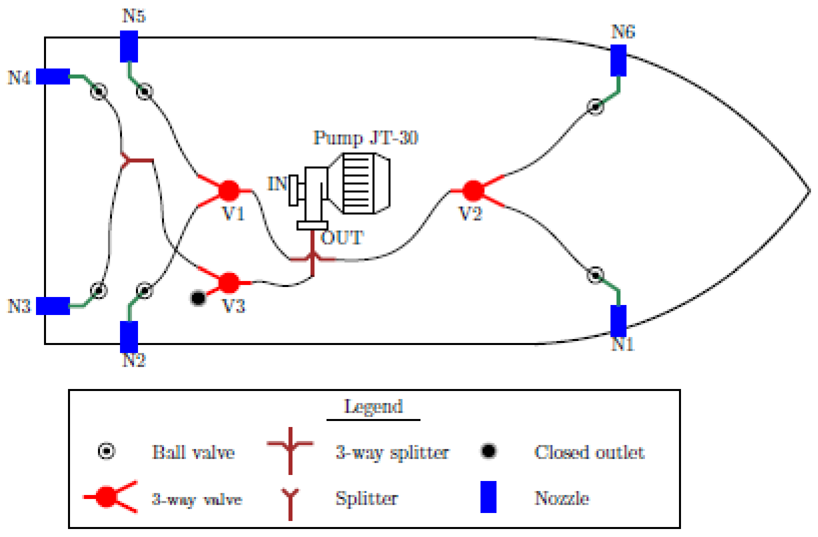


FIGURE 2.5: System nozzle configuration for thrust allocation [13]

2.1.3 Thrust Allocation

After the calculation of the control forces τ that need to act upon the vessel in order to maintain its position and heading, it is nessesary to translate them in a meaningful control input $\mathbf{u}$ for the actuators installed inside the vessel. This procedure is summed up in the next block diagram 2.6.

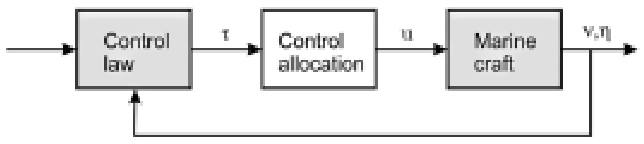


FIGURE 2.6: Block diagram showing control allocation in feedback system [7]

The control force due to an actuator can be written, assumed linearity, as:

$$f = ku \tag{2.2}$$

where k is the force coefficient depending on the geometrical characteristics and thetype of the actuator.

The total forces and moments, with the force vector defined as $\mathbf{f} = [F_x, F_y, F_z]^T$, for a vessel in 6 DOF can be expressed as:

$$\tau = \begin{bmatrix} F_x \\ F_y \\ F_z \\ F_z l_y - F_y l_z \\ F_x l_z - F_z l_x \\ F_y l_x - F_x l_y \end{bmatrix} \tag{2.3}$$

where $\mathbf{r} = [l_x, l_y, l_z]^T$ are the moment arms.

Regarding dynamic positioning, where only the planar motions are of importance, the expression (2.3) can be reduced to the following force vector matrix (2.4).

$$\tau = \begin{bmatrix} F_x \\ F_y \\ F_y l_x - F_x l_y \end{bmatrix} \tag{2.4}$$

In the case of our model test boat, the thruster configuration and the nozzle control forces are shown on the next figure 2.7. The inscribed distances were measured and are presented in Table 2.3.

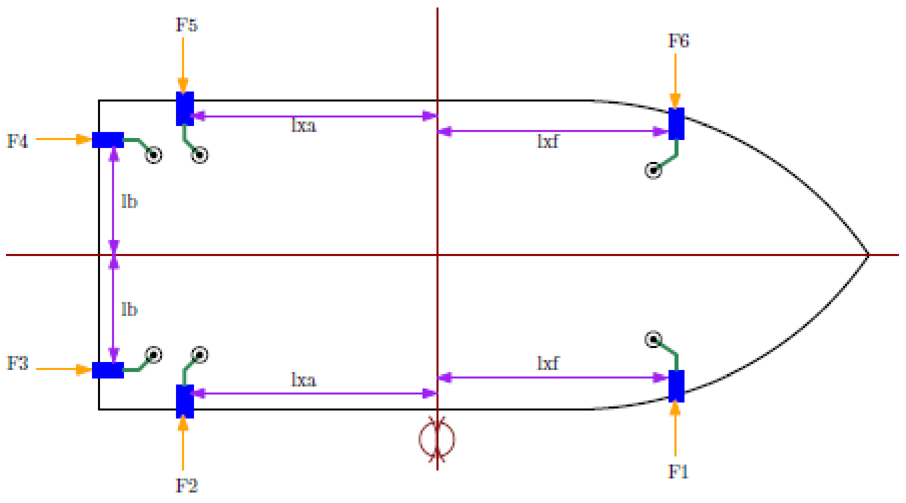


FIGURE 2.7: Schematic representation of nozzle positions and control forces [13]

Distance from Mid	Value [m]
l_{xf}	0.75
l_{xa}	1.67
l_b	0.45

TABLE 2.3: Nozzle distances from body-frame origins

Based on the above Schematic 2.7, the vessel can be controlled based on the following assumptions:

- Transom nozzles only produce a surge forward force F_x
- Aft nozzles produce a sway force F_{1y} that can be either positive or negative depending on the nozzle activated for the required direction
- Aft nozzles produce a sway force F_{2y} that can be either positive or negative depending on the nozzle activated for the required direction
- Activation of either aft or fore nozzles (force couple) as well as both nozzles (lack of symmetry) result in a yaw moment around the body frame z axis that can be calculated by the expression $M_z = F_{1y} l_{xa} + F_{2y} l_{xf}$.

Finally, the equation (2.4) can be rewritten for this specific system layout as:

$$\tau = \begin{bmatrix} F_x \\ F_{1y} + F_{2y} \\ F_{1y} l_{xa} + F_{2y} l_{xf} \end{bmatrix} \tag{2.5}$$

2.2 Hardware

2.2.1 Sensor Selection

In order to measure the vessel's linear accelerations and angular velocities, Sparkfun's 9-DOF IMU LSM9DS1 was selected (see Figure 2.8). These readings will be proven useful for determinining the angular and linear position of the ship at any given time.

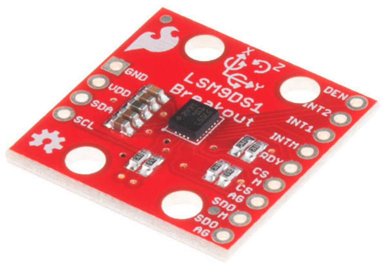


FIGURE 2.8: LSM9DS1 IMU Module

An IMU unit typically incorporates three sensor types, namely:

- Gyroscope: Used to measure angular velocity
- Accelerometer: Used to measure linear acceleration
- Magnetometer: Used to measure the magnetic field's strength

The key features of the LSM9DS1 IMU are the following:

- Up to 16g linear acceleration measurement scale
- Up to 16 Gauss magnetic measurement scale
- Angular rate sensing capability up to 2000 degrees per second

A major disadvantage of using an IMU, is the acccumulation of measurement errors through time via the integration of measurement noise to compute linear velocities, angular and linear positions over time. This leads to a phenomenon called drift which denotes the difference between where the sensor thinks it is located and its actual rotation. To tackle this problem, a GPS module was opted for, that would be used alongside the IMU using an appropriate filter.

As for the GPS, Adafruit's Ultimate GPS Module (see Figure 2.9 was favored with an external antenna to amplify its signal. The use of a GPS solely would not be effective as its readings are commonly suffering from internal lag between the sensor and the data transmission satellites and the position accuracy is very low and thus exceeds the acceptable error in DP operations.

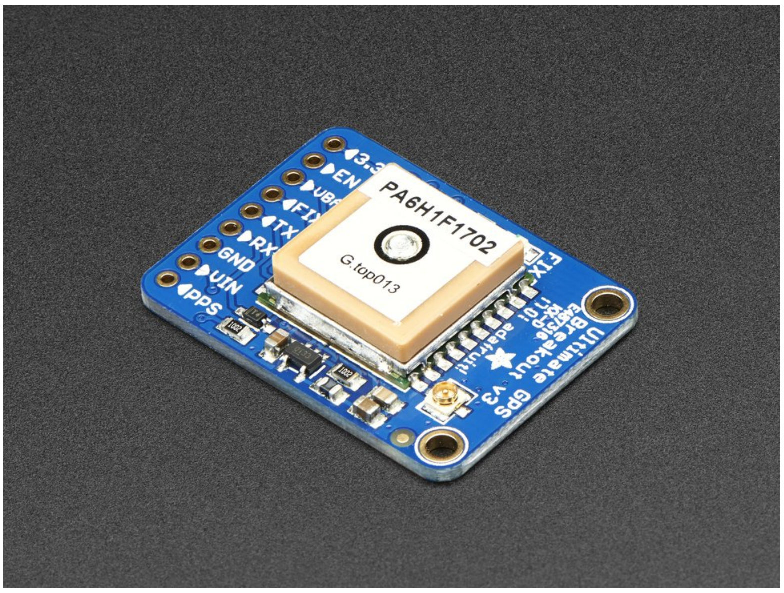


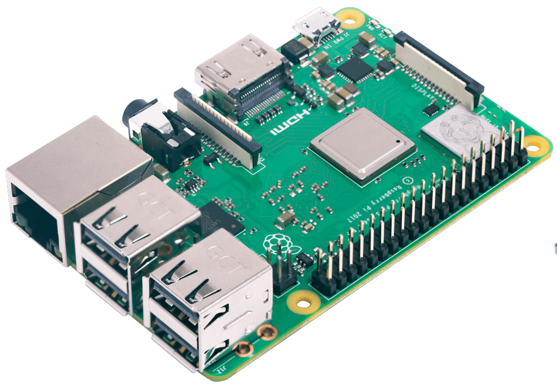
FIGURE 2.9: Adafruit's Ultimate GPS Module

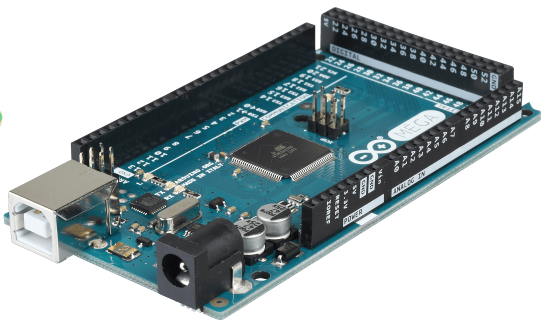
The GPS module's core features are the following:

- Position accuracy up to 3 m. This value is common for all consumer available GPS modules.
- Velocity accuracy: 0.1 m/second
- Maximum measurable velocity up to 515 m/second
- Warm/cold start, meaning the amount of elapsed time until the GPS gets a fix, equal to 34 seconds.
- Update rate 1 to 10 Hz
- Satellites: 22 tracking, 66 searching

2.2.2 Microcontrollers

In order to implement control over the thruster system, a Rasberry PI is used in conjuction with an Arduino Mega 2560.


(A) Raspberry PI 3B+


(B) Arduino Mega

The Arduino board is connected to the terminal switches in each end of each valve as well as the motors through H-bridges.

The Raspberry PI, on the other hand, is connected to the Ultimate GPS module produced by Adafruit via serial communication and Sparkfun's LSM9DS1 IMU module via I^2C. The actual controller is programmed in the RPI board and sends its output to the Arduino through Serial communication for immediate thruster deployment.

I^2C Protocol

I^2C typically uses a 2-wire configuration signifying two distinct lines: the Serial DAta line (SDA) and the Serial CLock one (SCL). The SCL line is rensponsible for data transmission between the two devices under an accurate time frame, whereas the SDA line serves as the path intended for the actual data transmission. This Protocol can support devices from 0 kHz to 5 MHz.

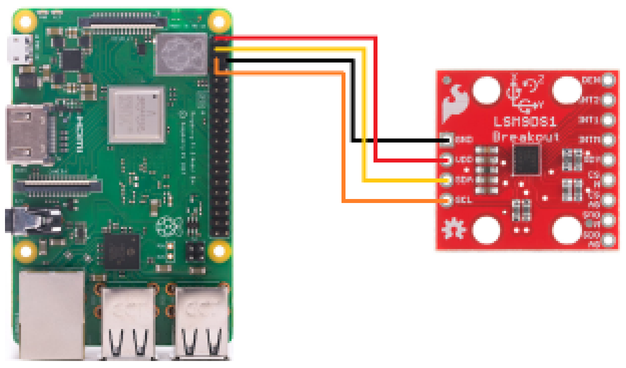


FIGURE 2.11: RPI 3B+ and IMU in I^2C Communication [13]

In every I^2C communication, there is a master device (in this case the RPI) as well as one or multiple slave devices (for example the IMU). The master device generates the clock and initiates communication with the slaves whereas the slave nodes receive the clock and can only respond when addressed to by the master. Each device on the BUS receives a 7-bit address in order to facilitate the overall communication. There are also some cases where a 10-bit extension would be used instead, but that is definitely not the norm.

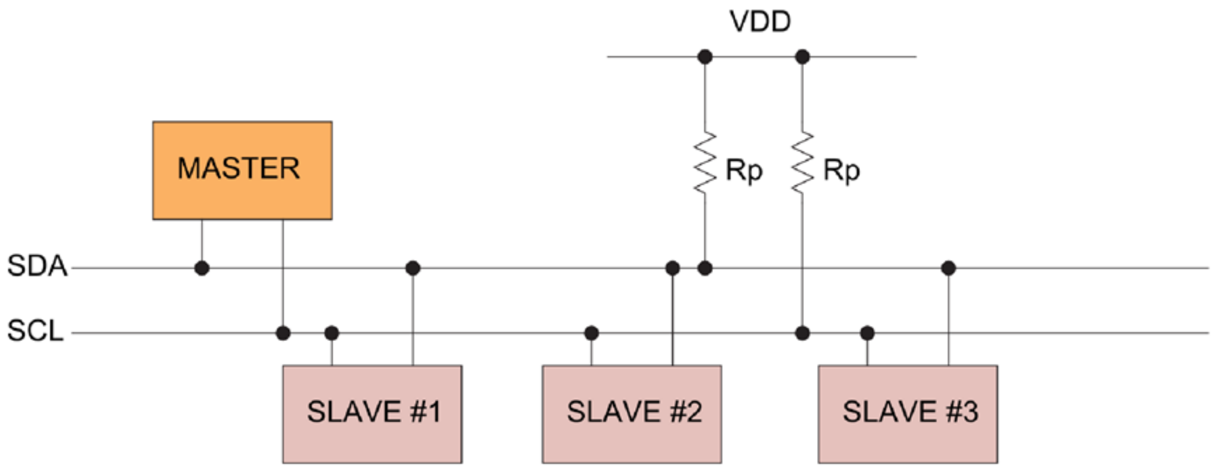


FIGURE 2.12: I^2C used notation

At the start, both lines are kept HIGH by pull-up resistors and when the communication is about to start, the SDA line gets pulled gets LOW. Then, the SCL begins to generate clock pulses and the first byte is sent by the master. This byte contains the slave adress accompanied by a read/write bit which receives the value 0 when a write operation is requested by the master and 1 when the master wishes to read data from the slave. If the slave exists, then it responds with an ACKnowledgement bit (ACK) and then the communication continues with the master in read/write mode and the slave in the complementary mode, as instructed by the read/write bit. In addition to that bit, the I^2C bus allows special START and STOP signals which act as message delimiters and are distinct from the data bits. The above procedure can be more easily visualised using the below Figure 2.13.

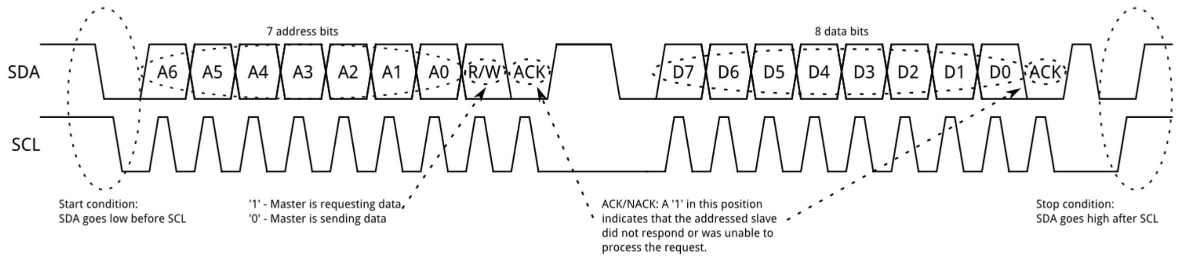

7 address bits
8 data bits
SDA
SCL
A6
A5
A4
A3
A2
A1
A0
R/W
ACK
D7
D6
D5
D4
D3
D2
D1
D0
ACK
Start condition:
SDA goes low before SCL
'1' - Master is requesting data,
'0' - Master is sending data
ACK/NACK: A '1' in this position indicates that the addressed slave did not respond or was unable to process the request.
Stop condition:
SDA goes high after SCL

FIGURE 2.13: I^2C Communication Protocol [Sparkfun]

Serial communication

A serial bus consists of two wires; one for receiving data (RX) and one for transmitting data (TX). Obviously the RX pin of one device should be connected to the TX pin of the other and vice-versa. Another thing to note, is that in this case the communication could either be full-duplex meaning that both devices send and receive data simultaneously or half-duplex meaning that one device only sends and the other device only receives data.

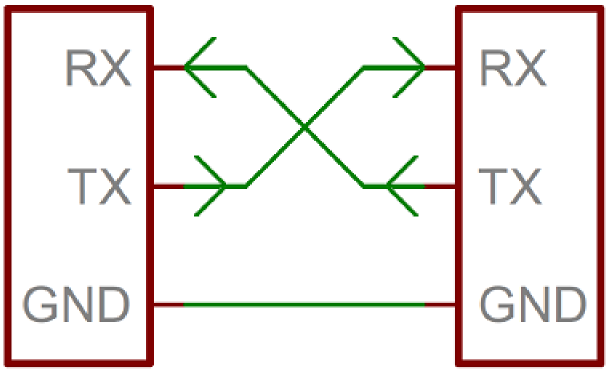


FIGURE 2.14: Serial Communication Connection [Sparkfun]

When microcontrollers and other low-level ICs communicate serially they usually do so at a TTL (transistor-transistor logic) level. TTL serial signals exist between a microcontroller's voltage supply range - usually 0V to 3.3V or 5V. A signal at the VCC level (3.3V, 5V, etc.) indicates either an idle line, a bit of value 1, or a stop bit. A 0V (GND) signal represents either a start bit or a data bit of value 0.

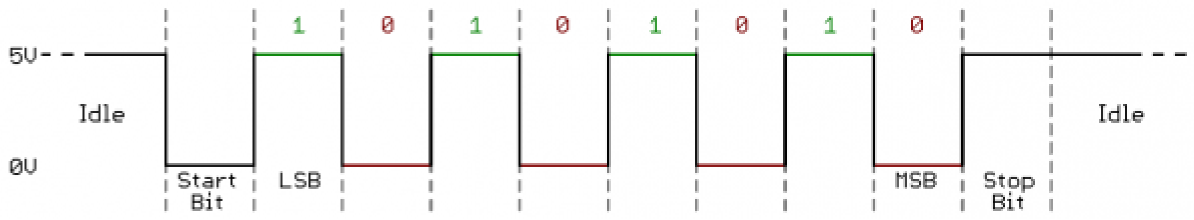


FIGURE 2.15: Serial Communication Protocol [Sparkfun]

2.3 Software

2.3.1 Arduino Code

As stated in the hardware section, the Arduino is responsible for controlling the pump unit and the valve system and receives instructions from the Raspberry PI via Serial communication. The code to implement all the needed functionality is written in the classical Arduino language which is for a large part inspired by the C Programming Language.

2.3.2 Raspberry Code

In contrast to Arduino, the RPI has its own operating system which is based on Debian Linux. The necessary Programming can be achieved using a multitude of programming languages. For the purposes of this thesis, the Python Language was chosen to achieve all tasks mainly due to its simplicity, its variety of available libraries and the fact that it belongs to the extended C family and thus coherence can be maintained easily.

Graphical User Interface

In order to overview the progress of sea trials from a larger PC screen, to efficiently communicate the status of the vessel to the user at any given time and to be able to control the ship without writing commands on a terminal, a GUI solution was implemented. The FLASK Python library was chosen for this task as it provides a simple yet powerful solution to broadcast a Web Server on the local network hosted on a single device, in our case the Raspberry PI. FLASK makes use of HTML and CSS to render its interface which by nature is static. Python is then used to route certain localhost addresses to the respective HTML pages and to broadcast the available data on a JSON page. Finally, parts of the interface are refreshed constantly with the use of Javascript that makes asynchronous requests to the server components at constant intevals. The same applies to when a certain button is pressed i.e the go forward button where Javascript pings the corresponding server page which in turn sends the appropriate command to Arduino using the Python Serial Library. A screenshot of the develloped environment can be seen in the next Figure 2.16.

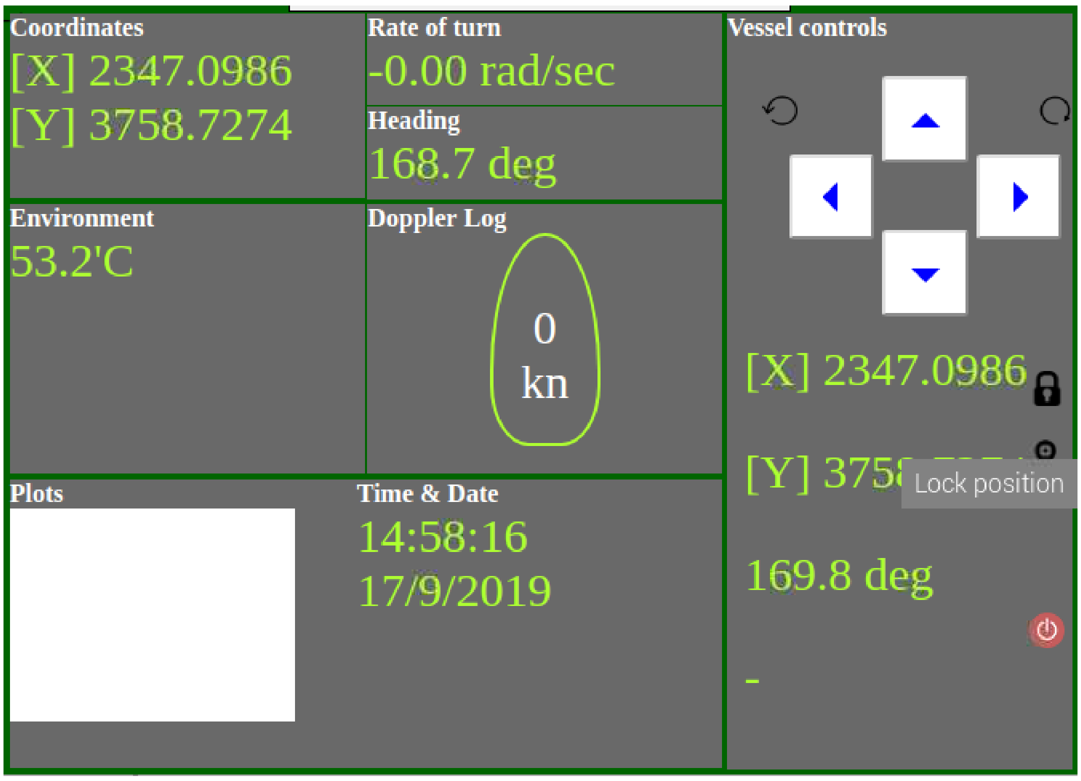


FIGURE 2.16: Flask Graphical User Interface

Control Script

This script is actually responsible for engaging the appropriate actuation meaning communicating the network's desired action from the RPI to the Arduino. Its initialisation is performed using a specific button on the GUI which in turn calls the script at static time intervals using the APScheduler module of the Python language. Thus, the control script is viewed as a background parallel process which can be activated and deactivated using a dedicated button and repeats itself over static intervals. The interval chosen is equal to 1 sec due to the fact that anything below that would render the control of the valves problematic, meaning that the valves would not be able to make it to either one of the terminal switches before the pump would turn on, as seen in the sea trials.

Internally, the control script uses the neural network itself by providing the current state as input at any given time. The state is comprised of the sway position, speed and acceleration as well as the yaw position, speed and acceleration whereas the output (the action of the system) belongs to the discrete space meaning that it can only be full to the left, to the right, clockwise and counter-clockwise as we are not capable of adjusting either the inflow capacity of the pump or the accurate position of the valves due to hardware restrictions.

Another feature of the proposed control script is that it can be used for online training using the Reinforcement Learning DQN Architecture. As the RPI 3B+ suffers from only having 1 GB of available memory, pretraining of the network needs to be done in a simulation environment as described in Chapter 6. During sea trials the trained network can be further fine-tuned to the real-world situation either using the RPI itself in case of small adjustments or by establishing a Wifi communication between the computer which would handle the Processing and posting the weights and the Raspberry Pi which would feed the states to the computer and grab the updated weights output. This proposed communication scheme could be achieved for instance by using a JSON page as the intermediate.

GPS and IMU setup and communication

As can be retrieved by the LSM9DS1's Datasheet, the following measurement modes can be configured:

Parameter	Value	Unit	Sensitivity
Linear acceleration	±2	g	0.061 mg/LSB
	±4	g	0.122 mg/LSB
	±8	g	0.244 mg/LSB
	±16	g	0.732 mg/LSB
Magnetic field	±4	gauss	0.14 mgauss/LSB
	±8	gauss	0.29 mgauss/LSB
	±12	gauss	0.43 mgauss/LSB
	±16	gauss	0.58 mgauss/LSB
Angular Rate	±245	dps	8.75 mdps/LSB
	±500	dps	17.50 mdps/LSB
	±2000	dps	70 mdps/LSB
Working Temperature	-40 to 85	deg C	

Table 2.4: LSM9DS1 IMU measurement modes and errors

In our case the accelerations, the magnetic field as well as the rotational speeds are relatively small. Thus, the selected hardware configurations are $\pm 2g$, $\pm 4gauss$ and $\pm 245dps$ all of which have the lowest possible error and as a result the measurement drift effect is further lowered.

In the case of the Ultimate GPS Module, all that needs to be configured is the frequency of the measurements which can range from 1 Hz (default) to 10 Hz (maximum). In order to facilitate the sensor fusion the selected frequency needs to be as high as possible. Thus, we end up with choosing 10 Hz sampling frequency. In addition, as stated in the Datasheet of the sensor the position accuracy can be lowered from 3 m to 2.5 m using an external Antenna as in the case of the project.

	Description
GPS Solution	MTK MT3339
Frequency	L1, 1575.42 MHz
Position Accuracy	3 m (No aid)/2.5 m (Ext. Antenna)
Velocity Accuracy	0.1 m/sec (No aid)/0.05 m/sec (Ext. Antenna)
Timing Accuracy	10ns (Typical)
Update Rate	1 Hz-10 Hz
Baud Rate	9600 bps
Altitude Measurement Range	Max. 18000 m (60000 feet)
Velocity Measurement Range	Max. 515 m/sec (1000 kn)
Acceleration Measurement Range	Max. 4G
Working Temperature	-40 $\deg C$ to +85$\deg C$

Table 2.5: Adafruit Ultimate GPS Version 3 specs

As previously stated, the communication with the IMU is performed using the I^2C communication protocol by exploiting one of the two I^2C lines present on the RPI. As for the case of the Ultimate GPS, the communication is achieved through Serial with a Baud Rate of 9600 as stated in the Datasheet of the GPS Module itself.

The proposed communication script queries both sensors for a change in their measurements at static intervals and posts the available data on the JSON page described above. These data are then in the disposal of the GUI and the control script.

3 Mathematical Modelling

3.1 Modelling of Marine Surface Vehicles

3.1.1 Coordinate Frames

In Dynamic Positioning there are a number of different coordinate frames that need to be introduced [12]:

- **The body frame b** is the coordinate frame of the IMU. Its center coincides with the center of the accelerometer triad and is aligned to the casing. All the inertial measurements are resolved in this frame.
- **The navigation frame** η is a local geographic frame in which we want to navigate. In other words, we are interested in the position and orientation of the b-frame with respect to this frame. For most applications it is defined stationary with respect to the earth. However, in cases when the sensor is expected to move over large distances, it is customary to move and rotate the n-frame along the surface of the earth.
- **The inertial frame i** is a stationary frame. The IMU measures linear acceleration and angular velocity with respect to this frame. Its origin is located at the center of the earth and its axes are aligned with respect to the stars.
- **The earth frame e** coincides with the i-frame, but rotates with the earth. That is, it has its origin at the center of the earth and axes which are fixed with respect to the earth.

When analysing the ship's motion through water, there are two main reference coordinate frames used: the Earth-fixed and the Body-fixed frame that coincide with the frames i and b mentioned above, respectively. The first is though inertial, meaning that the motion of Earth hardly affects low speed marine vehicles. On the other hand, the Body-fixed frame moves and rotates alongside with the body itself and as a result its axes are always coincident with the principal vessel axes. The origin of the Body-fixed system is almost always chosen to coincide with the ship's center of gravity.

A schematic representation of the two distinct coordinate systems is shown on the next figure 3.1

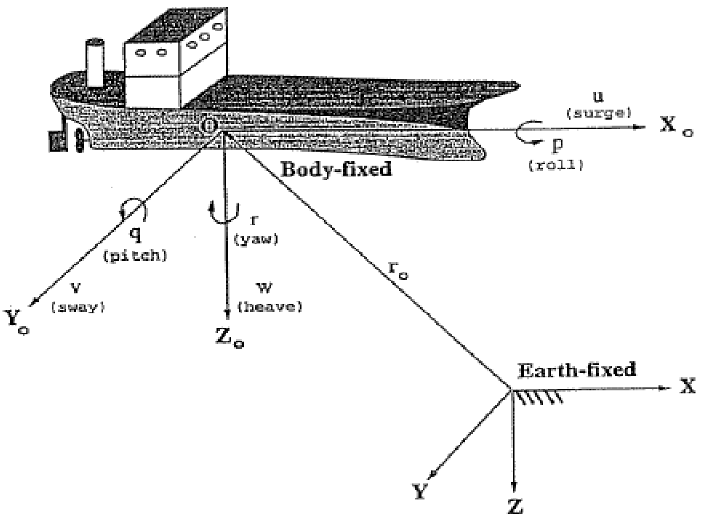


FIGURE 3.1: Coordinate Frames for Marine Vehicles [6]

Due to the above description, it is common sense that the position and rotation of the vehicle should be described using the Earth Fixed system whereas the linear and angular velocities should be expressed in the body-fixed system.

3.1.2 Hydrodynamic Forces acting on a moving vessel

When a marine vessel maneuvers in still water condition, hydrodynamic forces act on it that are directly dependent on the shape of the hull, the existence of appendages,the rudder and the propeller. Even today, the derivation of a complete mathematical model to link a specific ship form to its maneuvering characteristics has not yet been achieved, studies using both the available theory and experimental data can be used for prediction even at the earlier stages of the design.

In order to develop an approximate yet solid representation of the system at hand, the following hypotheses will be made:

- The maneuvering action is slow. This indeed holds true in this thesis' case as the thrust system results in relatively low ship speeds.
- The forces acting on the ship in the time moment t are only dependent on the speed and acceleration values at the same time moment t and not their history from when the maneuvering action started

With these hypotheses in mind, the hydrodynamic forces in the 3 planar DOFs (surge, sway, yaw) receive the following representation:

$$X = X(u, v, r, \dot{u}, \dot{v}, \dot{r}) \tag{3.1a}$$

$$Y = Y(u, v, r, \dot{u}, \dot{v}, \dot{r}) \tag{3.1b}$$

$$N = N(u, v, r, \dot{u}, \dot{v}, \dot{r}) \tag{3.1c}$$

The previous functions can be further expanded using the Taylor series representation:

$$\begin{aligned}
X = {} & f_1(u_0, v_0, r_0, , \dot{u}_0, \dot{v}_0, \dot{r}_0) + \\
& \frac{\partial X}{\partial u} \cdot du + (\geq \textit{second order terms dependent on } u) + \\
& \frac{\partial X}{\partial v} \cdot dv + (\geq \textit{second order terms dependent on } v) + \\
& \frac{\partial X}{\partial r} \cdot dr + (\geq \textit{second order terms dependent on } r) + \\
& \frac{\partial X}{\partial \dot{u}} \cdot d\dot{u} + (\geq \textit{second order terms dependent on } \dot{u}) + \\
& \frac{\partial X}{\partial \dot{v}} \cdot d\dot{v} + (\geq \textit{second order terms dependent on } \dot{v}) + \\
& \frac{\partial X}{\partial \dot{r}} \cdot d\dot{r} + (\geq \textit{second order terms dependent on } \dot{r})
\end{aligned} \tag{3.2a}$$

$$\begin{aligned}
Y = {} & f_2(u_0, v_0, r_0, , \dot{u}_0, \dot{v}_0, \dot{r}_0) + \\
& \frac{\partial Y}{\partial u} \cdot du + (\geq \textit{second order terms dependent on } u) + \\
& \frac{\partial Y}{\partial v} \cdot dv + (\geq \textit{second order terms dependent on } v) + \\
& \frac{\partial Y}{\partial r} \cdot dr + (\geq \textit{second order terms dependent on } r) + \\
& \frac{\partial Y}{\partial \dot{u}} \cdot d\dot{u} + (\geq \textit{second order terms dependent on } \dot{u}) + \\
& \frac{\partial Y}{\partial \dot{v}} \cdot d\dot{v} + (\geq \textit{second order terms dependent on } \dot{v}) + \\
& \frac{\partial Y}{\partial \dot{r}} \cdot d\dot{r} + (\geq \textit{second order terms dependent on } \dot{r})
\end{aligned} \tag{3.2b}$$

$$\begin{aligned}
N = {} & f_3(u_0, v_0, r_0, , \dot{u}_0, \dot{v}_0, \dot{r}_0) + \\
& \frac{\partial N}{\partial u} \cdot du + (\geq \textit{second order terms dependent on } u) + \\
& \frac{\partial N}{\partial v} \cdot dv + (\geq \textit{second order terms dependent on } v) + \\
& \frac{\partial N}{\partial r} \cdot dr + (\geq \textit{second order terms dependent on } r) + \\
& \frac{\partial N}{\partial \dot{u}} \cdot d\dot{u} + (\geq \textit{second order terms dependent on } \dot{u}) + \\
& \frac{\partial N}{\partial \dot{v}} \cdot d\dot{v} + (\geq \textit{second order terms dependent on } \dot{v}) + \\
& \frac{\partial N}{\partial \dot{r}} \cdot d\dot{r} + (\geq \textit{second order terms dependent on } \dot{r})
\end{aligned} \tag{3.2c}$$

Considering that our system starts from an equilibrium we have the liberty of taking the values of f_1, f_2, f_3 as zero. Moreover,under the assumption that the surge motion does not induce sway and yaw forces and that at the start of surge motion $u_0 = U$, $\dot{u}_0 = 0$:

$$\begin{aligned}
X &= X_{\dot{u}} \cdot \dot{u} + X_u \cdot (u - U) \\
Y &= Y_{\dot{v}} \cdot \dot{v} + Y_v \cdot v + Y_{\dot{r}} \cdot \dot{r} + Y_r \cdot r \\
N &= N_{\dot{v}} \cdot \dot{v} + N_v \cdot v + N_{\dot{r}} \cdot \dot{r} + N_r \cdot r
\end{aligned} \tag{3.3}$$

where $X_{\dot{u}}, X_u, Y_{\dot{v}}, Y_v, Y_{\dot{r}}, Y_r, N_{\dot{v}}, N_v, N_{\dot{r}}, N_r$ denote the hydrodynamic derivatives of the vessel.

3.1.3 Equations of planar motion for a Sea Surface Vehicle

As with all bodies, the study starts from Newton's second law which can be expressed as follows regarding the motion of the ship's centre of gravity:

$$\vec{F} = \frac{d}{dt}(m\vec{U}) \tag{3.4}$$

where $\vec{U} = u\vec{i} + v\vec{j}$ is the ship's velocity vector analysed in a frame with its origin on the centre of gravity and axes always parallel to the main ship axes (surge and sway) meaning that the coordinate system follows the rotation of the ship.

(3.4) can be further analysed, keeping in mind that the mass of the vessel is time invariant, as follows:

$$\vec{F} = m \cdot (\frac{d\vec{i}}{dt}u + \vec{i}\dot{u} + \frac{d\vec{j}}{dt}v + \vec{j}\dot{v}) \tag{3.5}$$

For small rotations $d\psi$ the vector time derivatives can be calculated as depicted in the following figure 3.2, meaning that:

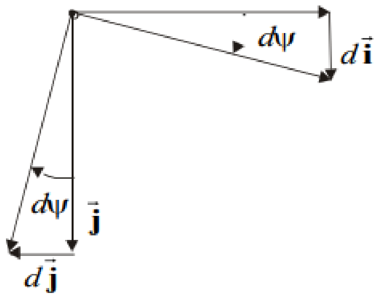


FIGURE 3.2: Planar finite rotation

$$\begin{aligned} d\vec{i} = d\psi\vec{j} &=> \frac{d\vec{i}}{dt} = r\vec{j} \\ d\vec{j} = -d\psi\vec{i} &=> \frac{d\vec{j}}{dt} = -r\vec{i} \end{aligned} \tag{3.6}$$

Subtituting in (3.5) yields:

$$\vec{F} = X\vec{i} + Y\vec{j} \tag{3.7}$$

where: $X = m \cdot (\dot{u} - rv)$ $Y = m \cdot (\dot{v} + ru)$

In the same manner, by utilizing the equation of angular momentum and supposing zero roll and pitch motion, we can deduce that:

$$\begin{aligned} K &= 0 \\ M &= 0 \\ N &= I_z * \dot{r} \end{aligned} \tag{3.8}$$

where I_z denotes the moment of inertia along the ship's yaw axis passing through the vessel's center of gravity.

Combining (3.7) with (3.3) and external forces X_{EXT}, Y_{EXT}, N_{EXT}, yields:

$$\begin{aligned} Surge: &\quad (m - X_{\dot{u}})\dot{u} - X_u \Delta u = X_{EXT} \\ Sway: &\quad (m - Y_{\dot{v}})\dot{v} - Y_v v - Y_{\dot{r}}\dot{r} + (mU - Y_r)r = Y_{EXT} \\ Yaw: &\quad - N_{\dot{v}}\dot{v} - N_v v + (I_z - N_{\dot{r}})\dot{r} - N_r r = N_{EXT} \end{aligned} \tag{3.9}$$

Here, the external forces incorporate the thruster action as well as the system disturbances (see Chapter 4).

Following a similar analysis to that of K. Spyrou [30] to decouple the sway and yaw motions as shown in the system of equations (3.9) results in two Nomoto-like 2nd order differential equations. These take the form:

$$\begin{aligned} T_1^y T_2^y \ddot{v} + (T_1^y + T_2^y)\dot{v} + v &= K_Y^y * Y_{EXT} + K_N^y N_{EXT} \\ T_1^\psi T_2^\psi \ddot{r} + (T_1^\psi + T_2^\psi)\dot{r} + r &= K_Y^\psi * Y_{EXT} + K_N^\psi N_{EXT} \end{aligned} \tag{3.10}$$

Where:

$$\begin{aligned} T_1^y T_2^y = T_1^\psi T_2^\psi &= \frac{(m - Y_{\dot{v}})(I_z - N_{\dot{r}}) - N_{\dot{v}} Y_{\dot{r}}}{C} [s^2] \\ T_1^y + T_2^y = T_1^\psi + T_2^\psi &= \frac{(mU - Y_r)N_{\dot{v}} - N_v Y_{\dot{r}} - Y_v(I_z - N_{\dot{r}}) - (m - Y_{\dot{v}})N_r}{C} [s] \\ K_Y^y &= -\frac{N_r}{C} [s/kg] \\ K_N^y &= -\frac{mU - Y_r}{C} [s/(m \cdot kg)] \\ K_Y^\psi &= \frac{N_v}{C} [s/(m \cdot kg)] \\ K_N^\psi &= -\frac{Y_v}{C} [s/(m^2 \cdot kg)] \\ C &= Y_v N_r + N_v(mU - Y_r) [Abkowitz\, coefficient] \end{aligned}$$

3.2 System Identification

3.2.1 Empirical Methods for the estimation of hydrodynamic derivatives

Across the years, multiple studies have been performed using experiments to correlate basic ship characteristics with its hydrodynamic derivatives. The most notable ones are those of Wagner Smitt (1971-1972), Norrbin (1971), Inoue (1981) and Clarke (1982). The assumed formulas from all these studies are undimensionalised using the terms L for length, L/U for time and $1/2 \cdot \rho \cdot L^3$ for mass where ρ denotes the density of sea water ($1.025t/m^3$) in the following manner:

$$\begin{aligned}
Y_v^{'} &= \frac{Y_v}{1/2 \cdot \rho \cdot L^2 \cdot U} \\
Y_r^{'} &= \frac{Y_r}{1/2 \cdot \rho \cdot L^3 \cdot U} \\
Y_{\dot{v}}^{'} &= \frac{Y_{\dot{v}}}{1/2 \cdot \rho \cdot L^3} \\
Y_{\dot{r}}^{'} &= \frac{Y_{\dot{r}}}{1/2 \cdot \rho \cdot L^4} \\
N_v^{'} &= \frac{N_v}{1/2 \cdot \rho \cdot L^3 \cdot U} \\
N_r^{'} &= \frac{N_r}{1/2 \cdot \rho \cdot L^4 \cdot U} \\
N_{\dot{v}}^{'} &= \frac{N_{\dot{v}}}{1/2 \cdot \rho \cdot L^4} \\
N_{\dot{r}}^{'} &= \frac{N_{\dot{r}}}{1/2 \cdot \rho \cdot L^5}
\end{aligned} \tag{3.12}$$

Wagner Smitt (1971-1972) Wagner Smitt has derived an empirical approach for velocity derivatives prediction after extensive PMM testing. These are expressed as:

$$\begin{aligned}
Y_v^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot 1.59 \\
Y_r^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (-0.32) \\
N_v^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot 0.62 \\
N_r^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot 0.21
\end{aligned} \tag{3.13}$$

Norrbin (1971) Norrbin also used PMM experiments to derive the empirical formulas for the velocity derivatives. These are expressed as:

$$\begin{aligned}
Y_v^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (1.69 + 0.08 \cdot \frac{C_B}{\pi} \cdot \frac{B}{T}) \\
Y_r^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (-0.645 + 0.38 \cdot \frac{C_B}{\pi} \cdot \frac{B}{T}) \\
N_v^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (0.64 - 0.04 \cdot \frac{C_B}{\pi} \cdot \frac{B}{T}) \\
N_r^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (0.47 - 0.18 \cdot \frac{C_B}{\pi} \cdot \frac{B}{T})
\end{aligned} \tag{3.14}$$

Inoue (1981) Inoue investigated the maneuvering performance by studying rotating arm teats and oblique towing tests. In this scope, three oil tankers, three cargo ships, a containership, an LNG tanker and a RO/RO ship were tested in three conditions; full load, half load and ballast condition. The velocity derivatives as a result of the study are expressed as follows:

$$\begin{aligned}
Y_v^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (1.0 + 1.4 \cdot \frac{C_B}{\pi} \cdot \frac{B}{T}) \\
Y_r^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (-0.5) \\
N_v^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (\frac{2.0}{\pi}) \\
N_r^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (\frac{1.04}{\pi} - \frac{4.0}{\pi} \cdot \frac{T}{L})
\end{aligned} \tag{3.15}$$

Clarke (1982) As one might observe, although the aforementioned formulas are similar in their representation, they also differ substantially. This fact might be due to the experimental data used, the curve fitting techniques employed and the variation in the experiments' way of conduct. Clarke in 1982 used all the available acceleration and velocity derivatives available in litterature up to that time. As a result, data from 36 rotating arm experiments and 36 PMM experiments were collected and correlated with the $C_B \cdot \frac{B}{T}$ parameter. Although the scatter of data was indeed substantial due to the fact that correlation with other parameters was at first neglected, Clarke decided to perform linear regression using multiple predictor variables, namely:

- $\frac{B}{T}$
- $\frac{B}{L}$
- $\frac{T}{L}$
- $(\frac{B}{T})^2$
- $(\frac{B}{L})^2$
- $(\frac{T}{L})^2$
- $C_B \cdot \frac{B}{T}$
- $C_B \cdot \frac{B}{L}$
- $C_B \cdot \frac{T}{L}$
- $\frac{B^2}{T \cdot L}$
- $\frac{B \cdot T}{L^2}$

in an expression such as:

$$y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + \cdots \tag{3.16}$$

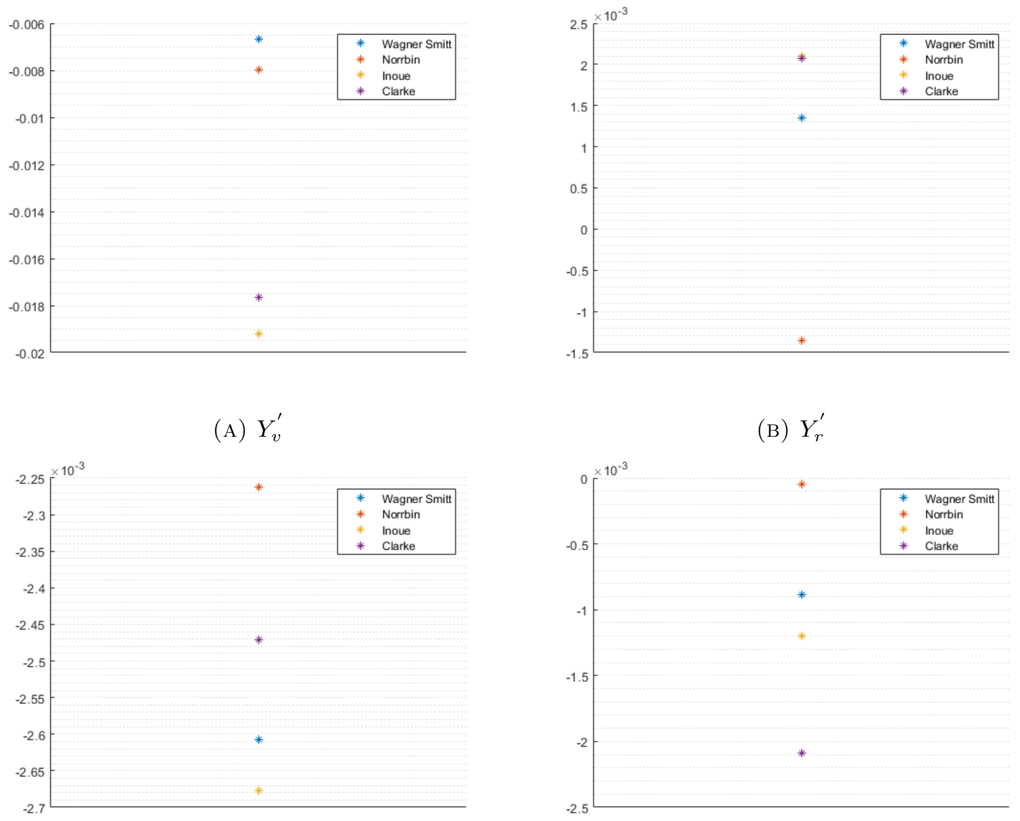
where y denotes the dependent variable (Here y could be any of the velocity and acceleration derivatives), $b_i\,, i = 0, 1, \cdots$ are the regression coefficients and $x_i\,, i = 1, 2, \cdots$ are the predictor variables.

Thus, the following expressions were established:

$$\begin{aligned}
Y_{\dot{v}}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (1 + 0.16 \cdot \frac{C_B B}{T} - 5.1 \cdot (\frac{B}{L})^2) \\
Y_{\dot{r}}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (0.67 \cdot \frac{B}{L} - 0.0033 \cdot (\frac{B}{T})^2) \\
N_{\dot{v}}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (1.1 \cdot \frac{B}{L} - 0.041 \cdot \frac{B}{T}) \\
N_{\dot{r}}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (\frac{1}{12} + 0.017 \cdot \frac{C_B B}{T} - 0.33 \cdot \frac{B}{L}) \\
Y_{v}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (1 + 0.4 \cdot \frac{C_B B}{T}) \\
Y_{r}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (-\frac{1}{2} + 2.2 \cdot \frac{B}{L} - 0.08 \cdot \frac{B}{T}) \\
N_{v}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (\frac{1}{2} + 2.4 \cdot \frac{T}{L}) \\
N_{r}^{'} &= -\pi \cdot (\frac{T}{L})^2 \cdot (\frac{1}{4} + 0.039 \cdot \frac{B}{T} - 0.56 \cdot \frac{B}{L})
\end{aligned} \tag{3.17}$$

3.2.2 Calculation of hydrodynamic derivatives

In order to identify the maneuvering characteristics of our bare bone hull, the aforementioned empirical methods were used according to the vessel characteristics mentioned in Chapter 2. The results for all the methods regarding the speed derivatives are presented in the form of MATLAB plots shown in figure 3.3. Both the velocity and the acceleration derivatives are presented in the following Table 3.1.



(A) Y_v' (B) Y_r'

(C) N_v' (D) N_r'

FIGURE 3.3: Calculation of velocity derivatives using multiple empirical methods

From the above diagrams, it is evident that the results of all methods mostly do not differ substantially. This however does not hold true for N_r'. As a result, in the case of Y_v', Y_r', N_v', there is no reason to neglect any of those results and their mean value is taken as the velocity derivative estimate whereas in the case of N_r' the Norrbin result is chosen to be neglected from the mean value. As for the acceleration derivatives, the lack of results from other methods makes the use of Clarke derivatives a one way solution. The results for all hydrodynamic derivatives from the empirical methods as well as the ones assumed from this point forward as the final estimates are presented in the following table.

Term	Wagner-Smitt	Norrbin	Inoue	Clarke	Final	Final [dim]
Y_v'	-0.0067	-0.008	-0.0192	-0.0177	**-0.0129**	**-399.45**
Y_r'	0.0013	-0.0014	-0.0021	-0.0021	**0.001**	**132.50**
N_v'	-0.0026	-0.0023	-0.0027	-0.0025	**-0.0025**	**-318.56**
N_r'	-0.0009	-0.00005	-0.0012	-0.0021	**-0.0014**	**-724.56**
$Y_{\dot{v}}'$				-0.0045	**-0.0045**	**-158.40**
$Y_{\dot{r}}'$				0.0011	**0.0011**	**648.75**
$N_{\dot{v}}'$				0.000042	**0.000042**	**6.14**
$N_{\dot{r}}'$				-0.0002	**-0.0002**	**-35.54**

TABLE 3.1: Empirical Methods' Results for Hydrodynamic Derivatives

3.3 System Stability

It can be proven that the assesment of the vessel stability can be performed using the Abkowitz coefficient mentioned above. The vessel should be considered stable in seakeeping if C>0 and unstable otherwise.

Solving for C yields: $C = -5.38 * 10^5 < 0$ meaning that the system is inherently unstable.

3.3.1 Phase Portrait

As in most analyses of mechanical system's behaviour, the phase portrait(v, $\dot{v}$) of the sway motion is examined in the next figure 3.4. The vector field is normalised and multiple sample orbits are examined. The start of each orbit is denoted by a circle whereas its end is denoted by a rectangle. The end of the orbit is not to be confused with its final point but symbolises the position of the vessel's center of gravity after a run time of 1 sec.

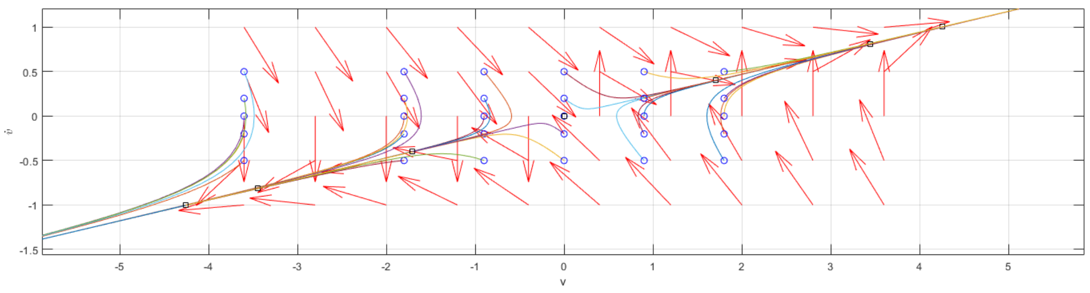


FIGURE 3.4: Sway phase portrait

From the portrait, it is evident that all trajectories go to infinity and no stable points exist on the plane to draw at least some of them. That means that our system is indeed unstable as the Abkowitz coefficient predicted above.

3.3.2 System pz-map

From a control engineer's perspective, the unstable nature of the system can be attributed to the positive and zero pole as presented in the next figure 3.5. Due to the fact that the state space represantation is identical for the sway and yaw motions as can be derived from the equation (3.10), only the sway motion pz-map is presented.

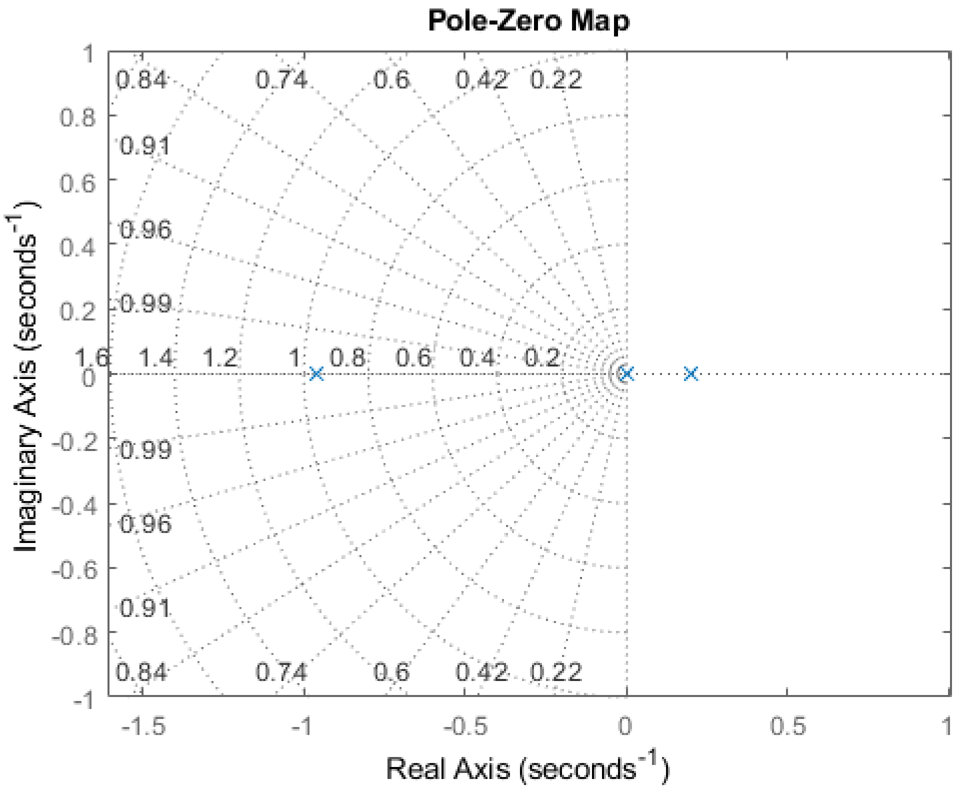


FIGURE 3.5: Sway motion Pole-Zero map

4 Environmental Disturbances

4.1 Types of Disturbances

According to [6] there are three main types of disturbances for a sea vehicle:

- Waves (wind generated)
- Wind
- Ocean currents

The magnitude of influence of each one of those factors varies from one vessel to another based on the type of vessel, its geometrical and operational characteristics and the field where the vessel operates.

According to the principle of superposition, the total external disturbance force and moment that the vessel faces can be expressed as:

$$\begin{aligned} X^{dist} &= X^{dist}_{waves} + X^{dist}_{wind} + X^{dist}_{currents} \\ Y^{dist} &= Y^{dist}_{waves} + Y^{dist}_{wind} + Y^{dist}_{currents} \\ N^{dist} &= N^{dist}_{waves} + N^{dist}_{wind} + N^{dist}_{currents} \end{aligned} \tag{4.1}$$

4.1.1 Wind-Generated Waves

The formation of waves due to effect of wind starts with small wavelets appearing on the sea surface. The drag force is thus increased with leads to growth of the previously small wavelets until they reach their breaking point and the amassed energy dissipates. These type of waves are usually described as a stohastic process of the following form:

$$\xi(x,t) = \sum_i (A_i \cdot cos(k_i * x - \omega_i t + \phi_i) \tag{4.2}$$

where A_i, $i = 1, 2, \cdots$ are the amplitudes of the wave group forming the total wave form, k_i, $i = 1, 2, \cdots$ is each wave's wave number ($k_i = \frac{2\pi}{\lambda_i}$), ω_i, $i = 1, 2, \cdots$ denotes the angular frequency of each wave unit,ϕ_i, $i = 1, 2, \cdots$ is a random phase and ξ stands for the elevation of the sea surface at the position x and at time t.

The basic geometric charactestics for a regular wave, meaning a wave that can be expressed as a singular periodic form is presented on the next figure 4.1

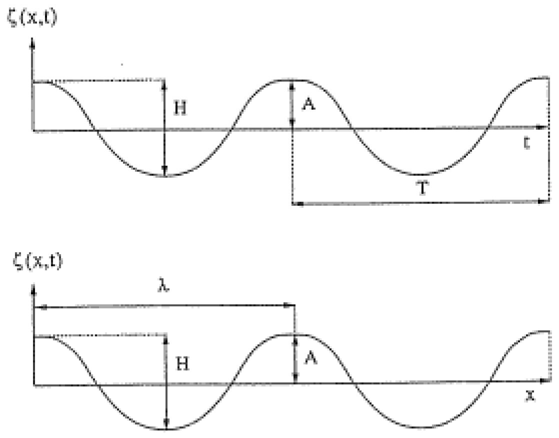


FIGURE 4.1: Characteristics of a regular wave

To summarize the above figure, a regular wave is characterized by its period, its length λ meaning how much the wave has travelled in one period, its amplitude (elevation from zero where zero coincides with the calm sea surface) and its height H=2A. The wave travels with a speed $c = \frac{\lambda}{T}$ (celerity) and has an angular frequency $\omega = \frac{2\pi}{T}$.

According to Neumann's representation, the wave elevation of a longcrested irregular wave propagating along the x-axis can be expressed as:

$$\xi(x,t) = \sum_{i=1}^{N}(A_i \cdot cos(\omega_i t - k_i x + \phi_i)) + \sum_{i=1}^{N}(\frac{1}{2}k_i A_i^2 \cdot cos2(\omega_i t - k_i x + \phi_i)) + O(A_i^3) \tag{4.3}$$

Moreover, the angular frequency of the wave component i is connected with its wave number via the expression:

$$\begin{aligned} \omega_i^2 &= k_i g tanh(k_i h) \\ h &= water\ depth \\ g &= 9.81 m/sec^2 \end{aligned} \tag{4.4}$$

For deep water the above equation (4.4) is simplified to:

$$\omega_i^2 = k_i g \tag{4.5}$$

Wave Spectra

The wave spectrum is a metric that describes the distribution of wave energy among different wave frequencies of wave-lengths on the sea surface. The amplitude of the wave component i is related to the wave spectral density function S using the following expression:

$$A_i^2 = 2S(\omega_i)\Delta\omega \tag{4.6}$$

Neumann Spectrum (1952) Historically speaking, the earliest spectral formulation is that of Neumann(1952) who proposed an one parameter spectrum given by the expression(4.7) [6].

$$S(\omega) = C\omega^{-6} \exp -2g^2\omega^{-2}V^{-2}[m^2 sec] \tag{4.7}$$

where C is an empirical constant, V is the wind speed and g is the acceleration of gravity.

Pierson-Moskowitz(1963) Pierson and Moskowitz in 1963 went on to develop a wind generated two parameter wave spectrum from analyses of wave spectra in the North Atlantic Ocean. The spectrum is represented by the equation (4.8).

$$\begin{aligned} S(\omega) &= A\omega^{-5} \exp -B\omega^{-4}[m^2 sec] \\ A &= 8.1 \cdot 10^{-3} \cdot g^2 \\ B &= 0.74(\frac{g}{V})^4 \end{aligned} \tag{4.8}$$

A graphical representation of the spectrum is shown on figure 4.2 [6].

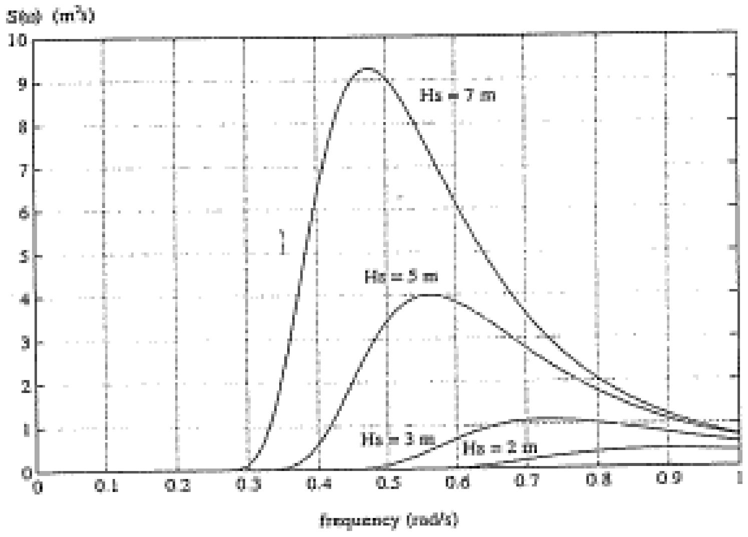


FIGURE 4.2: Pierson-Moskowitz spectrum for different values of Hs

JONSWAP (1968/1969) This spectrum was derived in 1968 and 1969 from analyses of wave spectra in the North Sea and is used for the description of non fully develloped seas. Its mathematical representation is that of equation (4.9).

$$
\begin{aligned}
S(\omega) &= 155\frac{H_s^2}{T_1^4\omega^5}\exp -\frac{944}{T_1^4\omega^4}\gamma^Y \\
Y &= \exp -(\frac{0.191\omega T_1 - 1}{\sqrt{2}\sigma})^2 \\
T_1 &= 0.834T_0 \\
\gamma &= \frac{S(\omega_0)_{JONSWAP}}{S(\omega_0)_{PM}} \\
\gamma &= 3.3\ [Hasselmann\, et\, al.] \\
\sigma &= \begin{matrix} 0.07, \ \omega \leq 5.24/T_1 \\ 0.09, \ otherwise \end{matrix}
\end{aligned}
\tag{4.9}
$$

Encounter angle

In general, sea waves can come from any possible direction in relation to the ship angle. The angle between the the wave direction and the vessel's principal longitudinal axis is referred to as the encounter angle β. According to this property, waves are categorized as follows:

- Head seas ($\beta = 180\,\mathrm{deg}$)
- Following seas ($\beta = 0\,\mathrm{deg}$)
- Quartering seas ($\beta \in (0, 90\,\mathrm{deg})$)
- Beam seas ($\beta = 90\,\mathrm{deg}$)
- Bow seas ($\beta \in (90\,\mathrm{deg}, 180\,\mathrm{deg})$)

A graphical representation of this notation is depicted in the following Figure 4.3[6].

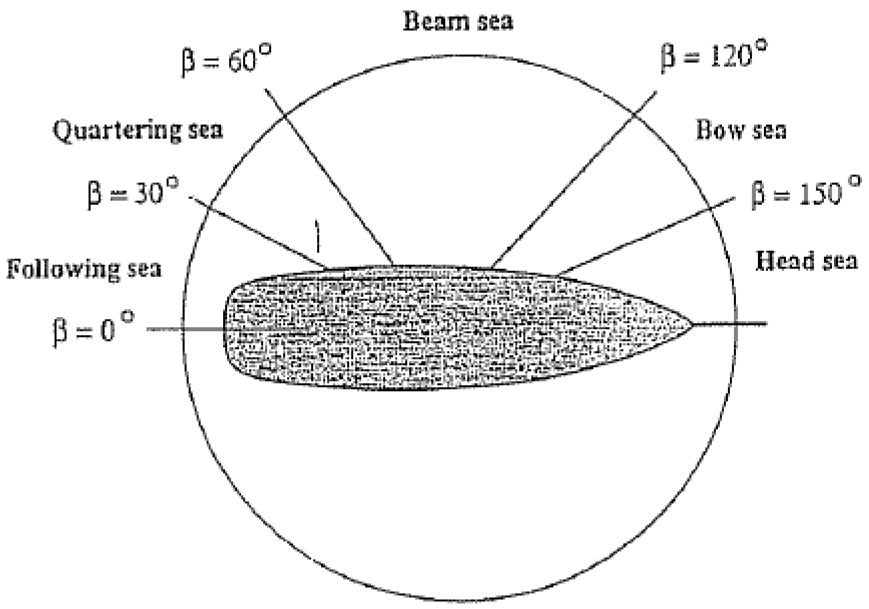


FIGURE 4.3: Definition of the ship's encounter angle

Wave Spectrum moments

The spectrum moments are defined as [6]:

$$m_k = \int_0^\infty \omega^k S(\omega) d\omega \qquad k = 0, 1, \cdots \tag{4.10}$$

In particular the square root of the zero spectrum moment m_0 can be interpreted as the RMS-value of the spectrum [6].

According to Price and Bishop [25] the principal wave height is connected to the zero spectrum moment via the expression:

$$H_s = 4\sqrt{m_0} \tag{4.11}$$

Moreover, the average wave period T_1 and the zero-crossings period T_z are defined as:

$$T_1 = 2\pi \frac{m_0}{m_1} \tag{4.12a}$$

$$T_z = 2\pi \sqrt{(\frac{m_0}{m_2})} \tag{4.12b}$$

As for the sea states, these signify ranges of waves' principal wave height and average period. According to Price and Bishop [25] Table's 4.1 notation is commonly used.

Sea state code	Description of sea	Wave height observed [m]
2	Smooth(wavelets)	0.1-0.5
3	Slight	0.5-1.25
4	Moderate	1.25-2.5
5	Rough	2.5-4.0

TABLE 4.1: Observed wave height and period based on sea state [25]

Wave Induced Forces

As previously described, irregular sea waves are surface gravity waves which may be represented by a stationary, ergodic Gaussian model for most ship dynamics considerations. The waves that move only in one direction may be called long-crested whereas the waves containing components propagating in many different directions are often called short-crested. Dynamic Positioning usually deals with long-crested waves, especially in the case of large ships [18].

There are two types of wave forces acting upon a ship:

- High-frequency wave forces which result in high-frequency motions of the vessel. These forces are proportional to the wave height.
- Persistent low-frequency wave drift forces that tend to drive the vessel off-station. These forces are proportional to the square of the wave heightand are due to the reflection by the vessel of incoming waves (See Figure 4.4).

The main focus in Dynamic Positioning is the second type of forces, since high-frequency forces often require thruster forces which are far of the attainable [24].

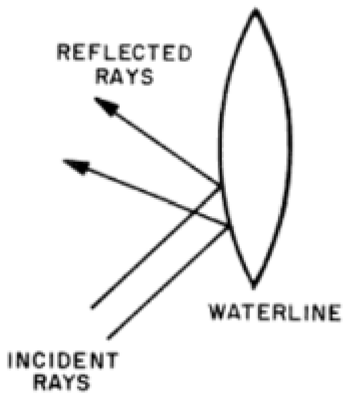


FIGURE 4.4: Reflection of sea waves [5]

The hydrodynamic high-frequency forces and moments acting upon a surface vessel can be analysed in three main components [9]:

- Those caused by undisturbed waves [Froude-Krylov] $F_{0k}, k = 1, \cdots, 6$
- Diffraction forces and moments $F_{dk}, k = 1, \cdots, 6$
- Radiation forces and moments $F_{lk}, k = 1, \cdots, 6$

expressed mathematically by the following equation (4.13).

$$\begin{aligned} F_k^{HD} &= -(j\omega^2)\rho A \int\int_{\partial D_B} \Phi_0 \eta_k dS_B - (j\omega^2)\rho A \int\int_{\partial D_B} \Phi_d \eta_k dS_B - (j\omega^2)\rho \sum_{l=1}^{6} \xi_l \int\int_{\partial D_B} \Phi_l \eta_k dS_B \\ &= F_{0k} + F_{dk} + \sum_{l=1}^{6} F_{lk} \end{aligned} \tag{4.13}$$

where Φ_0, Φ_d, Φ_l denote the undisturbed wave, diffraction and radiation potential respectively, $\xi_l, l = 1, \cdots, 6$ denote the displacement from the equilibrium status for each of the body's 6 DOF's, S_B denotes the wetted surface, A is the amplitude of the oscillation and $\eta_k, k = 1, \cdots, 6$ incorporates the three components of the perpendicular vector to the wetted surface in the three main directions (surge,sway,heave) and three others that stem fro those components and are used for rotations.

Also, let's not forget that the radiation forces and moments have already been incorporated into our mathematical model (Chapter 3) in the terms of damping $(\dot{v}, \dot{r})$ and inertia $(\ddot{v}, \ddot{r})$. This stems from the fact that this term incorporates both the added mass terms and the dampening terms as proven in [9].

With these said, the only terms left to deal with are the diffraction forces and moments and the Froude-Krylov ones.

A model for the approximation of Froude-Krylov force and moments in regular waves A simple model to use for the evaluation of Froude-Krylov sway force and yaw moment has been proposed in [19] with respect to Horizontal body axes (see Figure 4.5).

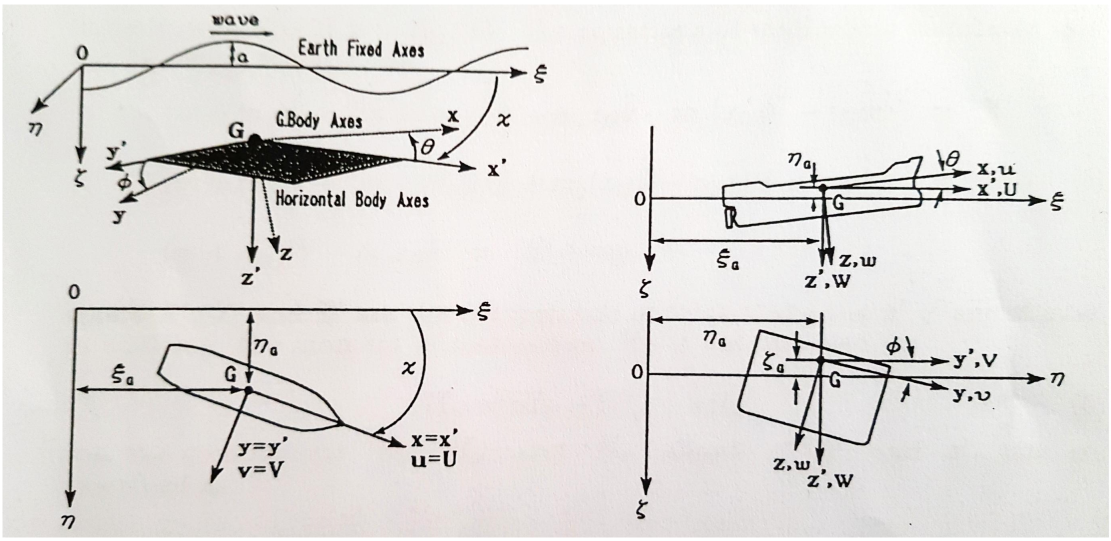


FIGURE 4.5: Horizontal body axes coordinate system [19]

The formulas receive the following form:

$$\begin{aligned} Y_{F.K} = {} & \rho gak \exp(-kz_s)(sin\chi + \psi cos\chi) \cdot \\ & [sin(k\xi_G - \omega t) \int_L A(x) cos(kxcos\chi)dx + \\ & cos(k\xi_G) \int_L A(x) sin(kxcos\chi)dx] \end{aligned} \tag{4.14a}$$

$$\begin{aligned} N_{F.K} = {} & \rho gak \exp(-kz_s)(sin\chi + \psi cos\chi) \cdot \\ & [sin(k\xi_G - \omega t) \int_L xA(x) cos(kxcos\chi)dx + \\ & cos(k\xi_G) \int_L xA(x) sin(kxcos\chi)dx] \end{aligned} \tag{4.14b}$$

where ψ is the yawing angle, $A(x)$ is the sectional area at position x, z_s is the subsurface of waves, χ is the angle between the prescribed ourse and direction of advance of waves and a denotes the amplitude of waves.

According to [11] the subsurface of waves is given by the following expression:

$$z_s = a \exp(-k\zeta) cos(k\eta + \omega t) \tag{4.15}$$

where η denotes the lateral position of interest and ζ the perpendicular position. For instance, if $\zeta = 0$, (4.15) coincides with the equation describing the elevation of the sea surface.

Approximation of drifting forces In regular waves where the height remains constant, the wave drifting force can is a constant but small force in the direction of the propagation of waves. However, in irregular waves, the variation in the wave height induces a similar variation in the drifting force value. In this case, it is common practice to split the total drifting force in two parts; one constant and one slowly oscillating.

A/ Regular Waves

In the case of regular waves, Pinkster [24] proposes the following model for the approximation of the total drifting force:

$$F = \frac{1}{2}\rho g a^2 R^2 \tag{4.16}$$

where R denotes the reflection coefficient dependent on the hull itself and is equal to the ratio of the amplitude of the reflected wave to that of the undisturbed wave.

B/ Irregular waves

In 1974, Newman [20] came up with a method to estimate second-order wave drifting forces in irregular seas.

Supposing a known time history of wave elevations, either from spectrum analysis or from appropriate measurement devices $\zeta(t)$, the linear expression of the wave elevation can be expressed as a finite series:

$$\zeta(t) = Re\sum_m A_m \exp(i\omega_m t) \tag{4.17}$$

where A_m denotes the complex wave amplitude that incorporates the random phase and ω_m denotes the radian frequency of the mth component of the spectrum. It is assumed that the index m has been ascribed in ascending order of frequencies meaning that $\omega_{m+1} \geq \omega_m$ for all values of m. In addition, if the second order transfer function between the drifting force and the wave elevation $F_{mn}$is known then the total drifting force at any time t can be expressed as:

$$f(t) = Re\sum_m\sum_n A_m A_n^* F_{mn} \exp(i(\omega_m - \omega_n)t) \tag{4.18}$$

where the asterisk denotes the conjugate of the respective quantity.

4.1.2 Wind Disturbance

Sea vehicles with high freeboard and/or superactrures often suffer from this type of disturbance. The effect of wind is greater as the projected surface increases and as the center of gravity is higher.

It is common to divide the wind in two components; a mean value and a flunctuating component(gust). The mean component is a monotonically increasing function of the distance from the free surface whereas the gust remains approximatelly constant. Although wind is a three-dimensional phenomenon, it is described by its total velocity magnitude U_w and its direction ψ_w.

According to the above the value of the wind speed can be described at any given time and distance from the free surface as follows:

$$U_w(z,t) = \bar{U}(z) + u(z,t) \tag{4.19}$$

, where **u** corresponds to the wind gust and $\bar{U}$ corresponds to the mean wind component.

A schematic representation of the notation used for wind disturbances can be seen in Figure 4.6 [8].

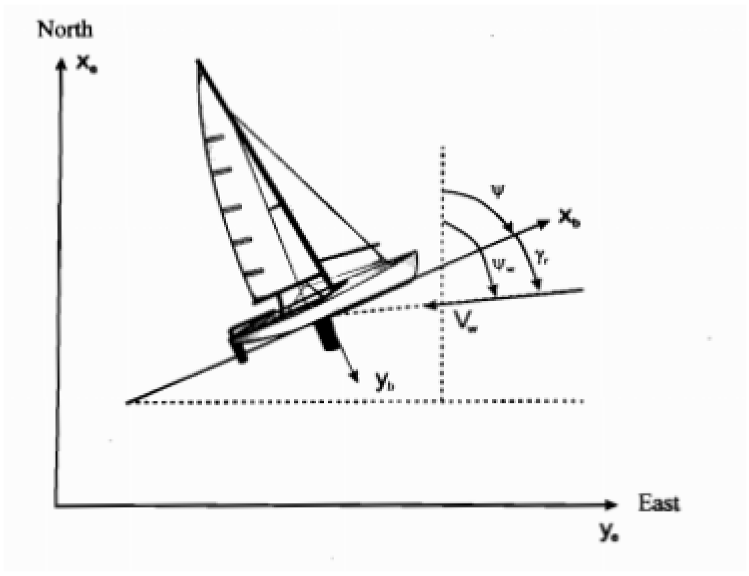


FIGURE 4.6: Definition of wind speed and direction [8]

Mean wind component

The mean velocity $\bar{U}$ at elevation z can be expressed as [29]:

$$\begin{aligned} \frac{\bar{U}(z)}{\bar{U}_{10}} &= 2.5\sqrt{\kappa}\ln\frac{z}{z_0} \\ z_0 &= 10\exp -\frac{2}{5\sqrt{\kappa}} \end{aligned} \tag{4.20}$$

In the above equation (4.20), $\bar{U}_{10}$ is the 1 hour mean wind speed at 10 m elevation and κ is the sea surface drag coefficient.

Another similar form to (4.20) has been proposed by Ochi [21] and is the following:

$$\begin{aligned} \bar{U}(z) &= \bar{U}_{10} + 2.5u_* \ln\frac{z}{10} \\ u_* &= \sqrt{\frac{\tau}{\rho}} \\ &\tau\ [shear\ force\ per\ m^2] \\ &\rho[water\ density] \end{aligned} \tag{4.21}$$

Wind gust

The wind gust, as in the case of the random phase model for waves, can also be described by a spectrum. Multiple spectrums have been formulated for this reason, amongst the most notable being the Davenport and the Harris spectrum.

Davenport spectrum (1961) According to [6] the Davenport spectrum is described by the following expression:

$$S_w(\omega) = k\frac{916700\omega}{((1+(191\omega/\bar(U)_{10}))^2)^{4/3}} \qquad k = 0.05\ (turbulence\ factor) \tag{4.22}$$

Harris spectrum (1971) This type of spectrum is described using the mathematical expression (4.23) [6].

$$S_w(\omega) = k\frac{5286\bar{U}_{10}}{(1+(286\omega/\bar{U}_{10})^2)^{5/6}} \tag{4.23}$$

In 1950, Watanabe correlated the gustiness of the wind with the average wind velocity in multiple and also extreme scenarios. The gustiness of the wind is defined as the ratio of the maximum velocity of the gust to the average wind velocity. The time window for each measurement corresponds to 2 hours. The following Figure 4.7 [26] portrays the experiments' results.

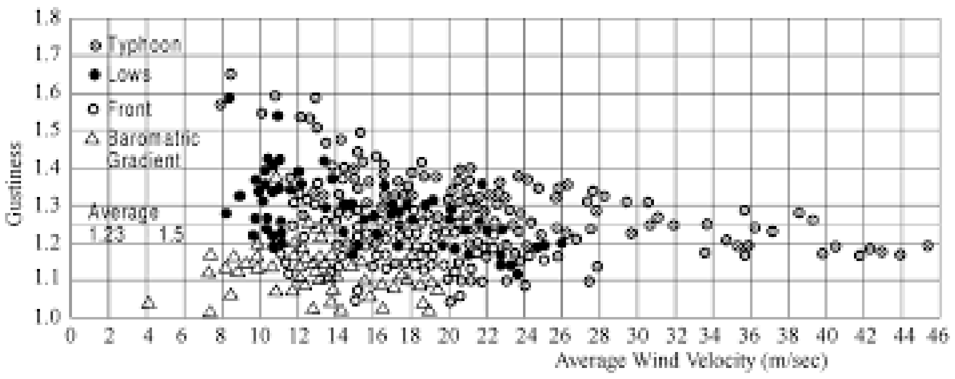


FIGURE 4.7: Wind gustiness versus wind average speed

The result of the study was that, on average, the gustiness can receive the value 1.23.

Wind Forces

The total force and moment acting upon a ship due to the effects of wind can be described as follows:

$$F_{wind} = \frac{1}{2}\rho C_D A U_w^2 \tag{4.24a}$$

$$M_{wind} = \frac{1}{2}\rho C_D A U_w^2 Z_r \tag{4.24b}$$

Substituting (4.19) into (4.24a) yields:

$$\begin{aligned} F_{wind} &= \frac{1}{2}\rho C_D A(\bar{U}^2 + 2\bar{U}u + u^2) \\ &= \bar{F}(1+\frac{2u}{\bar{U}} + (\frac{u}{\bar{U}})^2) \end{aligned} \tag{4.25}$$

where $\bar{F}$ is the force corresponding to the mean wind component.

Assuming that the second order term in the equation (4.25) can be neglected, the expression can be further simplified to:

$$F_{wind} = \bar{F}(1 + \frac{2u}{\bar{U}}) \tag{4.26}$$

Wind Resistance of Merchant Ships [16] In 1972, Isherwood suggested a way of modelling the wind surge, sway forces and yaw moment according to the following expressions:

$$X_{wind} = \frac{1}{2}C_X(\psi_w)\rho U^2 A_T \tag{4.27a}$$
$$Y_{wind} = \frac{1}{2}C_Y(\psi_w)\rho U^2 A_L \tag{4.27b}$$
$$N_{wind} = \frac{1}{2}C_N(\psi_w)\rho U^2 A_L L \tag{4.27c}$$

Where L=length overall, B=beam, A_L =lateral projected area to the wind, A_T =transverse projected area to the wind, A_{SS} =lateral projected area of the superstructure to the wind, S=perimeter of the model's lateral projection, C=distance from bow of centroid of lateral projected area and M=number of masts or kingposts seen in lateral projection.

The coefficients C_X, C_Y, C_N are retrieved using the following empirical formulas:

$$C_X = A_0 + A_1\frac{2A_L}{L^2} + A_2\frac{2A_T}{B^2} + A_3\frac{L}{B} + A_4\frac{S}{L} + A_5\frac{C}{L} + A_6 M \tag{4.28a}$$
$$C_Y = B_0 + B_1\frac{2A_L}{L^2} + B_2\frac{2A_T}{B^2} + B_3\frac{L}{B} + B_4\frac{S}{L} + B_5\frac{C}{L} + B_6\frac{A_{SS}}{A_L} \tag{4.28b}$$
$$C_N = C_0 + C_1\frac{2A_L}{L^2} + C_2\frac{2A_T}{B^2} + C_3\frac{L}{B} + C_4\frac{S}{L} + C_5\frac{C}{L} \tag{4.28c}$$

where A_i, B_i, C_i are tabulated coefficients according to the wind angle [16].

4.1.3 Ocean currents Disturbance

Currents stem mainly from the wind action upon the sea surface. For modelling such a disturbance with regard to a surface vessel positioning problem, a two dimensional model is sufficient [29]. Assuming that V_c is the speed magnitude of the current and ψ_c corresponds to its direction in the NED frame, the velocity vector can be expressed as:

$$\vec{v_c} = [V_c cos(\psi_c), V_c sin(\psi_c), 0]^T \tag{4.29}$$

For computer simulations, ocean currents can be incorporated using a 1st-orde Gauss-Markov Process [6] as in (4.30).

$$\begin{aligned} \dot{V}_c + \mu V_c &= w \\ \mu &\geq 0 \end{aligned} \tag{4.30}$$

where w is Gaussian white noise. Of course the value must be saturated meaning that upper and lowe bounds of current speed must be met [29].

4.2 Disturbance Model Application

4.2.1 General case

The largest concern for DP purposes are the slowly-varying drifting forces which act upon the vessel and tend to drive it off the desired position at any given time. Considering the end goal, which is to actuate the large EER vessel, the wind effects will also have to be incorporated in the model as the freeboard area is definitely significant for the overall dynamic analysis. In this case, Isherwood's model would most probably be handy in modelling the wind effects upon the vessel. Similar work can be found in [32] and [31].

A small example for using Isherwood's model has been written in MATLAB considering the small boat, just for illustrative purposes and is mainly concerned with the modelling of the yaw induced moment by the effect of the wind. The computation of the lateral and transverse projected areas to the wind can be accomplished using the Fusion 360 3D Model presented in Chapter 2. The code can be found in Appendix A.

The high-frequency forces contribute little to the model and thus can be ignored in order to reduce the network's task complexity especially in the initial design phase. However, if they are to be determined as important after the initial network's test phase Hamamoto et al's [19] models could be used. Finally, as it is generally considered difficult to model irregular waves effects, it is preferred to use regular waves as a start point. Irregular waves could be however modelled if the response amplitude operators (RAO) are known for the examined vessel. This could be accomplished for example by performing system identification on known experimental data or by using a raw to slightly-tweaked model of a similar known vessel form.

A small simulation was performed in Simulink concerning just the dynamic positioning in yaw of the HJDP-1 vessel using PID control by incorporating both the wave and wind effects. The model is shown in the next Figure 4.8.

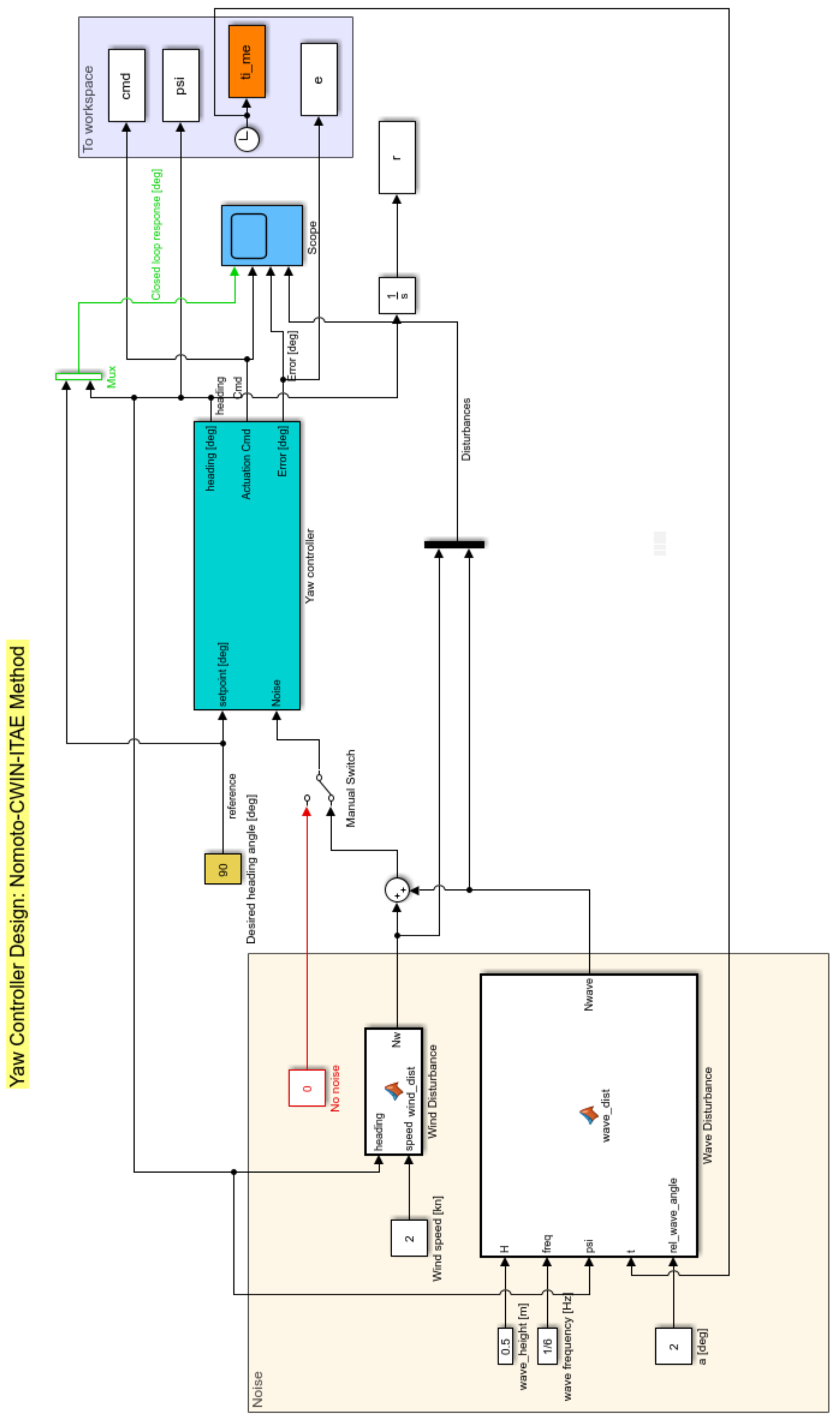


FIGURE 4.8: Simulink yaw controller model

4.2.2 Simplifications for the Neural Network Development

Due to the fact that this thesis deals mainly with a small testboat with minimum freeboard and no superstructures, the wind induced forces are going to be neglected. The same will apply to the sea currents disturbance.

As for the disturbance induced by the action of waves upon the vessel, only the low-frequency drifting forces will be considered as they are the main focus of DP. The subsequent studies will be performed for the case of regular waves of three different wave heights that are chosen according to the real-world thrust system's potential to stabilize the vessel. Pinkster's method [24] of calculating those forces is applied and the coefficient R is taken as the square of the sinus of the wave angle in order to both simulate its flunctuation during the vessel's movement and constrain its value in the interval [0,1].

5 Sensor Calibration and Fusion

5.1 Overview of the problem

As mentioned in Chapter 2 the sensors used in the project have their own respective errors which can affect the operation of the vessel at sea. The accelerometer and gyro of the IMU suffer from drift due to the continuous integration of measurement noise alongside the actual value of each measurement in order to determine the position and angles of the vessel at any given time from available accelerations and rotational speeds. Moreover, the accelerometer is susceptible to vibrations which can of course occure from the valve operation on-board the vessel. Another issue lies with the integration of the gyro data as the initial values of the angles are uknown. As for the magnetometer, its measurements are affected by both hard and soft iron sources that disturb the magnetic field around the vessel. On the other hand, the GPS besides its standard error of 2.5 m using the external Antenna provides relatively low-frequency measurements which on their own could render the control system unusable.

The main focus of the following paragraphs will be to:

- Calibrate the magnetometer by removing the effect of the soft and hard iron sources
- Calibrate the accelerometer and gyro at rest
- Filter the accelerometer's and gyro's measurements using low-pass filtering
- Use the accelerometer to estimate initial attitude
- Compensate the magnetometer for tilt
- Use the magnetometer to estimate heading
- Fuse the different sensors using complementary filters

5.2 Calibration

Each of the measurements of the 9-DOF IMU can be analysed in three main components, namely:

- The true value of the measurement
- The offset value of the sensor b
- The measurement noise w

,as described in the following equations (5.1).

$$Accelerometer : a_{IMU} = a_{true} + b_{acc} + w_{acc} \tag{5.1a}$$
$$Gyro : \omega_{IMU} = \omega_{true} + b_{gyro} + w_{gyro} \tag{5.1b}$$
$$Magnetometer : m_{IMU} = m_{true} + b_{mag} + w_{mag} \tag{5.1c}$$

The sensors' offsets and noise derive from their manufacturing process. As low-end equipment is used for the purposes of the project, the hardware precision is expected to be lower than what is achivable using modern technology. However, depending on the application, all these hardware errors can be minimized through the use of software, a case that holds true for the relevant to the Thesis DP purposes.

In the next paragraphs, methods to exclude the hardware noise and offsets are going to be presented. In general, the offset values are taken while the sensors are holded still and are being substracted from the overall value. In addition, the involved noise is either filtered out using low-pass filtering or calibrated out in the case of the magnetometer. Finally, the sensor measurements are combined using complementary filters.

5.2.1 Magnetometer Calibration

The output of the magnetometer is affected mainly by the wide-band measurement noise, sensor imperfections, installation errors and interfering external magnetic fields. These magnetic fields can be further classified as:

- Soft-iron sources; Materials that generate their own magnetic field in response to the external magnetic field. The generated field is time-varying and gets affected by both the magnitude and the direction of the external field.
- Hard-iron sources; Constant or slowly time-varying fields that are generated by ferromagnetic materials in the proximity of the magnetometers.

The output of the magnetometer including errors can be expressed as [14]:

$$h^2 = (\frac{h_x - B_x}{\gamma_x})^2 + (\frac{h_y - B_y}{\gamma_y})^2 + (\frac{h_z - B_z}{\gamma_z})^2 \tag{5.2}$$

where B_x, B_y, B_z describe the hard iron induced errors, γ_x, γ_y, γ_z are the scale factors due to the soft iron sources and h_x, h_y, h_z are the three components of the actual magnetic field strength.

When no external interferences are present, rotating the magnetometer in 3D Space would render a sphere of constant radius. The hard-iron sources affect the position of the origin of the sphere but not the overall shape whereas the soft-iron sources stretch and tilt the sphere without affecting the position of its center.

The standard procedure used for magnetometer calibration resides in taking multiple measurements m_x, m_y, m_z of the magnetic field by simultaneously rotating the IMU in 3D-space. The maximum measurement in each of the 3 Degrees is taken as the soft-iron sources correction and the average magnitudes in the directions x , y, z as the hard-iron sources correction. This can be expressed through the following expressions:

$$m_{xi}^{corr} = \frac{m_{xi} - E[m_{xi}]}{max(m_{xi})} \tag{5.3a}$$

$$m_{yi}^{corr} = \frac{m_{yi} - E[m_{yi}]}{max(m_{yi})} \tag{5.3b}$$

$$m_{zi}^{corr} = \frac{m_{zi} - E[m_{zi}]}{max(m_{zi})} \tag{5.3c}$$

5.2.2 Accelerometer and Gyro Offset Removal

Accelerometer

An accelerometer that rests flat on a surface should only measure force on the z-axis, where the gravity vector is present. As a result, the x and y offsets can be taken equal to the mean value of the measurements in those DOF while the accelerometer is not moving. As for the z-axis offset, this can be derived by rotating the IMU by 90 deg and taking the average of the z-axis measurements as the respective offset. In the used software, the mean values of 100 measurements in each axis is taken as the offset. Moreover, the offset that is calculated for an axis, e.g. the x axis, is different when the vertical to horizontal plane axis is the y axis than when is the z axis. So the true offset value for each axis is calculated as shown below:

$$\begin{aligned} x - offset &= [(mean.offset - z.vertical) + (mean.offset - y.vertical)]/2 \\ y - offset &= [(mean.offset - z.vertical) + (mean.offset - x.vertical)]/2 \\ z - offset &= [(mean.offset - x.vertical) + (mean.offset - y.vertical)]/2 \end{aligned} \tag{5.4}$$

This procedure should be followed on every initialization of the measuring device and the resulting values be stored and subtracted from the accelerometer values on each iteration. It should be noted that offsets can slightly change, since the environment and mainly temperature influence the results. In advanced systems, temperature compensation needs to be also included but in the case of dynamic positioning is not obligatory.

Gyroscope

A similar procedure as that of the accelerometer is followed to calibrate out the gyroscope offset. While the device is held still, 100 measurements are taken for each of the three rotations and their means are substracted from the respective values during the following sea trials. Deviations due to temperature also exist but are not again very important regarding the purposes of the system itself (DP).

This work of finding and removing the offset values has already been performed in [13] for the exact same vessel that this Thesis deals with.

5.2.3 Low-pass Filtering for Accelerometer and Gyro

Both the accelerometer and Gyro sensors suffer from drift, meaning the continuous integration of errors (noise) alongside the actual measurements which tend to drive both the position and the angle estimates off over time. The noise induced in the measurements is of higher frequencies than the original data and thus could be filtered out using a Low-pass filter approach.

A low-pass filter (LPF) is a filter that passes signals with a frequency lower than a selected cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency. In order to decide the cutoff frequency, first we need to perform a Fast-Fourier Transform of the accelerations and rotational speeds' signals to determine the frequencies involved and in which of them the spectral density is higher. The calculations were performed using Matlab and the available sensor measurements from the conducted sea-trials.

In Figure 5.1 half the spectrum of the x-component of the linear acceleration is given. As for the other two components, the assumption is made that almost the same frequencies will be involved. Then, in Figure 5.2 again half the spectrum of the roll speed timeseries is plotted against the frequency axis. The same hypothesis for the sway and yaw components is made as in the previous case.

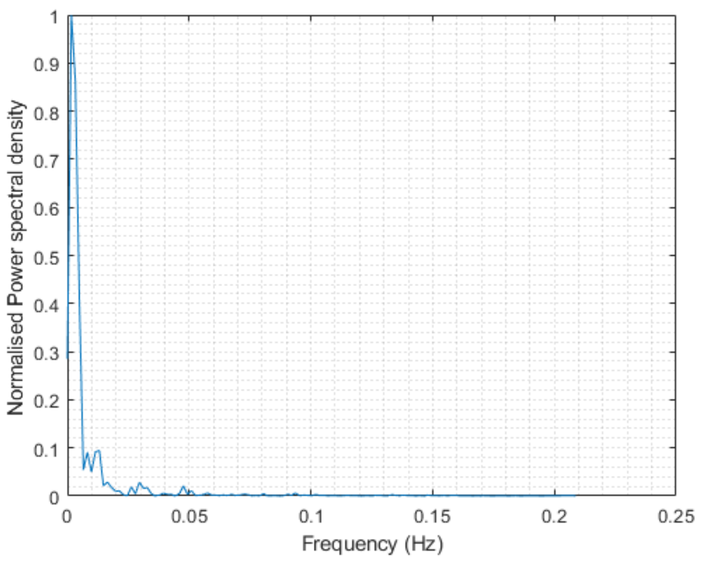


FIGURE 5.1: Accelerometer Spectral Analysis (surge)

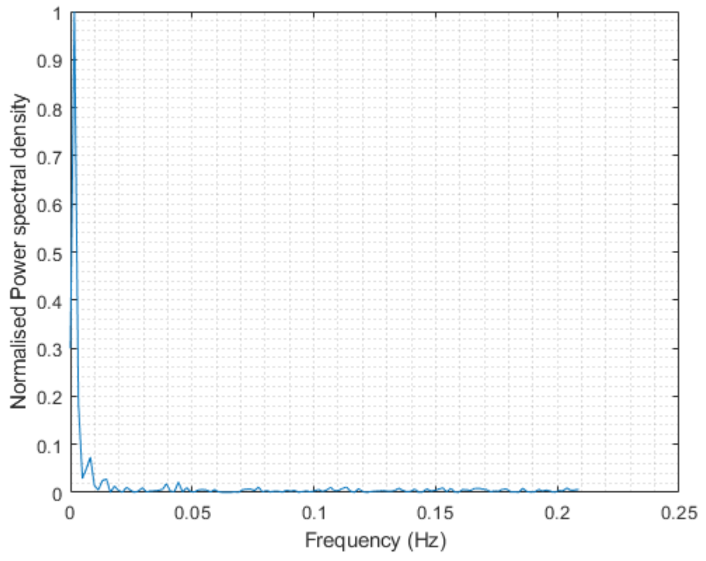


FIGURE 5.2: Gyro speed spectral analysis (roll)

In the above diagrams, the spectral density is normalised using the maximum component involved and thus it is undimensionalised. As one can deduce from looking at the Fourier Transforms, a plausible cutoff frequency would be around 0.05 Hz for both the accelerometer and the gyro measurements. Taking that into account and by zero-ing all components with a frequency higher than the cutoff frequency the inverse Fast-Fourier Transform yields the following results for the accelerations and gyro speed components (see Figures 5.3- 5.8).

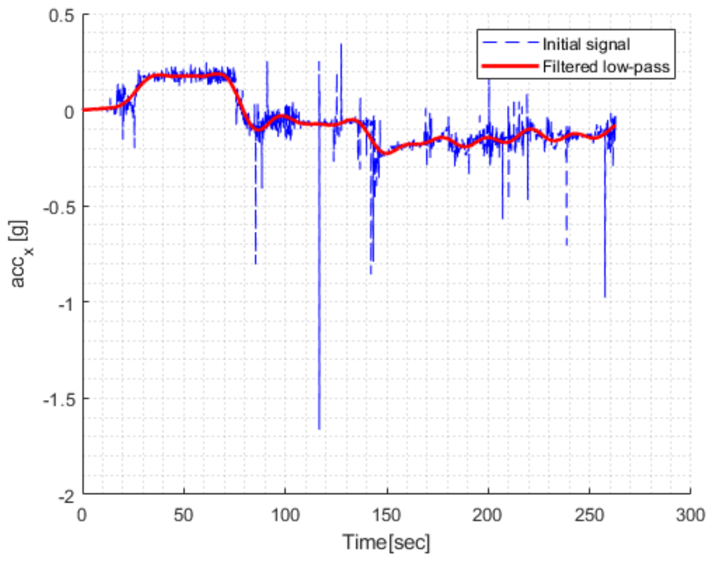


FIGURE 5.3: Accelerometer low-pass filter implementation $cutoff_{freq} = 0.05Hz$ (surge)

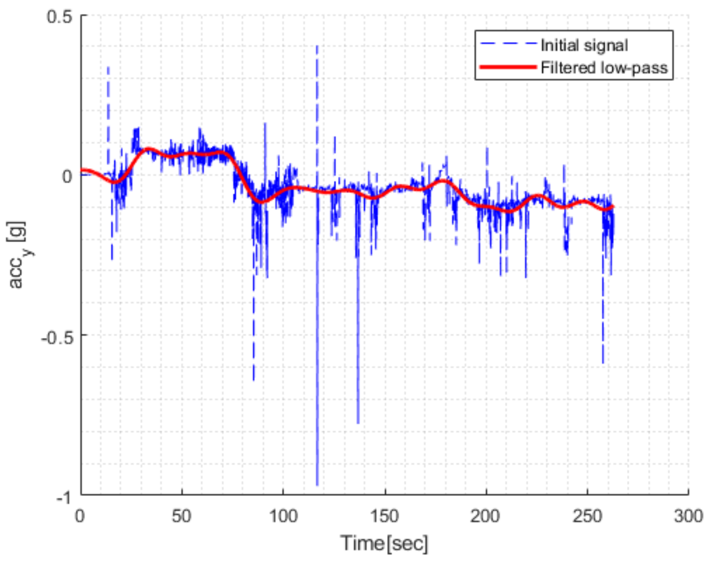


FIGURE 5.4: Accelerometer low-pass filter implementation $cutoff_{freq} = 0.05Hz$ (sway)

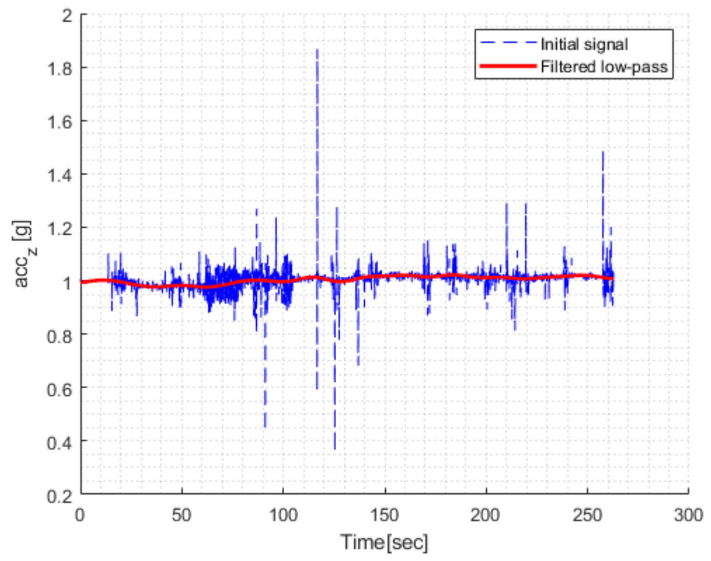


FIGURE 5.5: Accelerometer low-pass filter implementation $cutoff_{freq} = 0.05Hz$ (heave)

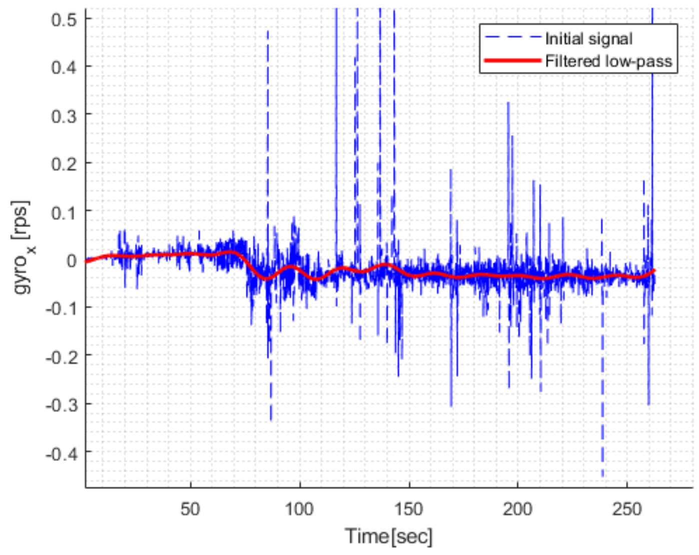


FIGURE 5.6: Gyro speed low-pass filter implementation $cutoff_{freq} = 0.05Hz$ (roll)

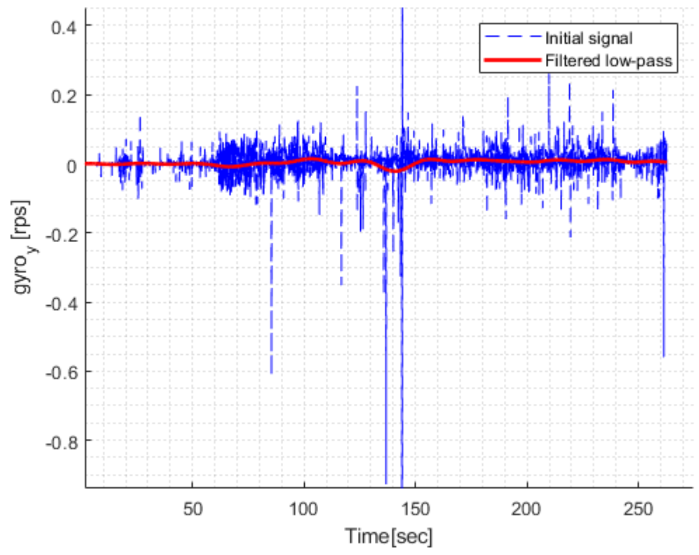


FIGURE 5.7: Gyro speed low-pass filter implementation $cutoff_{freq} = 0.05Hz$ (pitch)

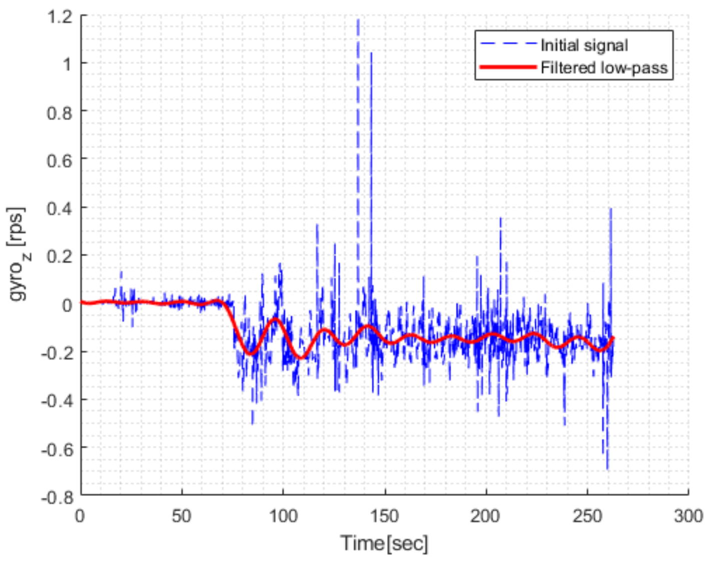


FIGURE 5.8: Gyro speed low-pass filter implementation $cutoff_{freq} = 0.05Hz$ (yaw)

As can be deduced from the above Figures, the low-pass filter is able to follow the dynamics of the initial signal by casting out a big part of the involved noise. The same filter can then be used in the actual plant to filter out the high frequency noise.

5.3 Attitude Estimation using the Accelerometer

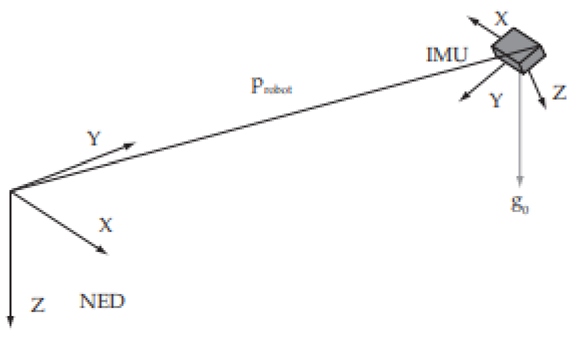


FIGURE 5.9: Initial attitude estimation using the accelerometer

The knowledge of the three vector components of the gravity vector when the system is at rest allows for an initial estimation of the roll and pitch angles. From Newton's second law we have:

$$\vec{a_m} = \frac{1}{m} * (\vec{f} - \vec{f_g}) \tag{5.5}$$

where m is the mass of the overall plant, $\vec{a_m}$ denotes the measured acceleration vector in the Body-Fixed System, $\vec{f}$ is the vector of the externally applied forces excluding gravity and $\vec{f_g}$ denotes the gravity force vector in the Body-Fixed System.

In the inertial Earth-Fixed NED frame the gravity force vector is $\vec{F_g} = [0, 0, mg]^T$. Translating $\vec{F_g}$ in the Body -fixed frame yields:

$$\vec{f_g} = \begin{bmatrix} -mgsin\theta \\ mgcos\theta sin\phi \\ mgcos\phi cos\theta \end{bmatrix} \tag{5.6}$$

Thus, when the system is at rest ($\vec{f} = 0$) we can deduce from (5.5) and (5.6) that

$$\vec{a_m} = \begin{bmatrix} gsin\theta \\ -gcos\theta sin\phi \\ -gcos\phi cos\theta \end{bmatrix} \tag{5.7}$$

As a result the initial roll and pitch angles can be expressed as:

$$\theta_0 = arcsin(\frac{a_{x0}}{g}) \tag{5.8a}$$

$$\phi_0 = arctan(\frac{a_{y0}}{a_{z0}}) \tag{5.8b}$$

5.4 Magnetometer Tilt Compensation

When the magnetometer is tilted, the pitch and roll angles are not equal to 0 deg. In this case, the components X_M, Y_M, Z_M of the magnetometer need to be compensated according to the equations (5.9) and (5.10) to obtain the corrected components X_h, Y_h.

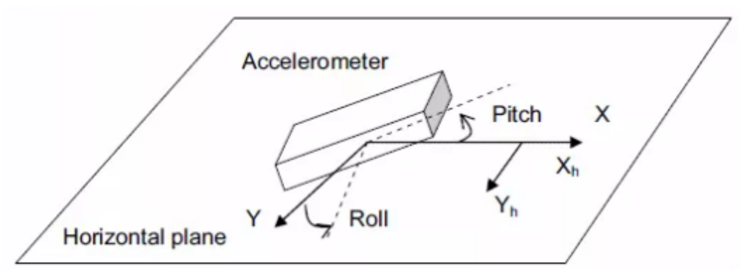


FIGURE 5.10: Tilt compensation for the magnetometer [13]

$$X_h = X_M cos\theta + Z_M sin\theta \tag{5.9}$$

$$Y_h = X_M sin\theta sin\phi + Y_M cos\phi - Z_M sin\phi cos\theta \tag{5.10}$$

5.5 Heading Estimation using the Magnetometer

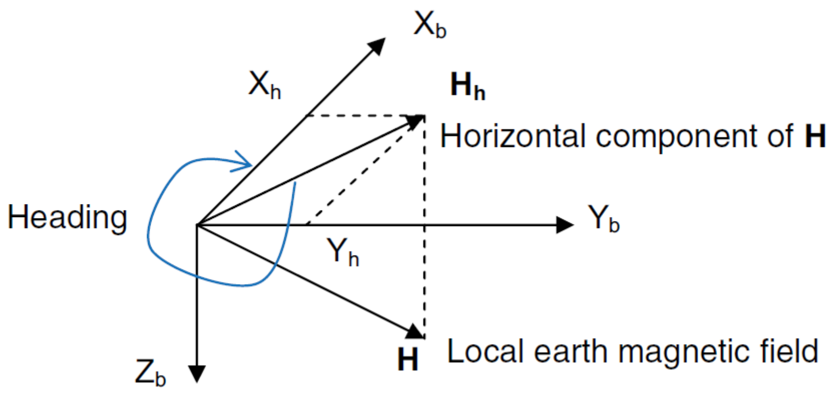


FIGURE 5.11: Local Earth magnetic field and heading calculation [13]

As shown in Figure 5.11 the already corrected and compensated planar components of the magnetometer can be used to obtain the heading angle in reference to the magnetic North using the following expression (5.11).

$$Heading = arctan(Y_h/X_h) \tag{5.11}$$

5.6 Sensor Fusion

Although a Kalman Filter would normally be our best option to get accurate measurements from fusing the different sensors involved, its requirement of an explicit noise model makes it go beyond the scope of this Thesis. Alternatively, a complementary filter approach is followed [12].

This type of filter can also estimate the orientation at time t given the measurements $y_{1:t}$. As aforementioned, the combination of the accelerometer and the magnetometer as well as the gyro by itself both provide information about the orientation of the sensor. The former are noisy but accurate during long periods of time while the former are accurate on a short time scale but drift over time.

The general mathematical representation of a complementary filter, used to fuse two sensors' measurements, can be expressed as follows:

$$final - value = a * value_{sensor1} + (1 - a) * value_{sensor2} \tag{5.12}$$

,where a is a coefficient that describes the degree of trust that we put in each of the sensors. A value of 0.8 means that we trust the values of sensor 1 four times more than those of sensor 2 and vice-versa (a=0.2) while a value of 0.5 means that we place equal trust on both sensors.

5.6.1 Position

The vessel's position regarding surge and sway is given both by the integration of the accelerometer values and the GPS. While the IMU provides data at a much higher and appropriate for DP rate, the drift of measurements, even if slow due to the data analysis described above, will continue to exist to a certain degree. On the other hand, the GPS provides measurements at a lower rate and incorporates a standard error of $\pm 2.5m$. For those reasons, the arbitrary choice of a=0.7 is made meaning that:

$$\begin{aligned} x &= 0.7 * x_{IMU} + 0.3 * x_{GPS} \\ y &= 0.7 * y_{IMU} + 0.3 * y_{GPS} \end{aligned} \tag{5.13}$$

Let's also keep in mind that the GPS coordinates need first to be translated in the x-y plane in terms of m, something that depends on both the geographical area and the type of projection used. Using this complementary filter, the GPS will counteract to a large degree the deviation of the accelerometer position values while maintaining a high measurement refresh rate.

5.6.2 Orientation

As previously stated, the accelerometer can be used to derive the vessel's pose in the roll and pitch DOF's while both the magnetometer and the GPS can be used to estimate heading. As for the gyro, this is the main sensor that can measure all three rotations. The arbitrarily chosen complementary filtering schemes are described in the following equation:

$$\begin{aligned} \phi &= 0.6 * \phi_{gyro} + 0.4 * \phi_{acc} \\ \theta &= 0.6 * \theta_{gyro} + 0.4 * \theta_{acc} \\ h &= 0.6 * h_{gyro} + 0.3 * h_{mag} + 0.1 * h_{GPS} \end{aligned} \tag{5.14}$$

It must be noted that the starting angle values to begin integration using the gyroscope are derives using the accelerometer and the magnetometer.

5.6.3 Speed

The vessel's speed can be obtained using both the on-board GPS and the integration of the accelerometer values. The GPS provides relatively high accuracy to speed measurement as seen from the specs mentioned in Chapter 2 but at a low rate. In contrast, the IMU values incorporate drift but are given at a higher rate. For those reasons, the following complementary filter is used to combine those measurements:

$$V = 0.6 * V_{IMU} + 0.4 * V_{GPS} \tag{5.15}$$

The velocities should always have the same units and thus either the GPS value should be converted from knots to m/sec or more preferably, the accelerometer speed values should be converted to knots. The symbolism V_{IMU} refers to the magnitude of the vector sum of the vessel's speeds in the surge and sway DOF's.

5.7 Sensor Fusion implementation

The mathematical model derived in Chapter 3 is purposed to be used for offline training of the network through computer simulation. However, in the testing phase of the network and the subsequent online training, the aforementioned model is going to be substituted by discrete sensor measurements. The measurements of each sensor will have to be be calibrated as discussed above to minimize the included errors and combined to further reduce the errors and determine a plausible set of measurement values. The network will have to receive those values as an input and produce an output according to its previous training. The system can be used both for online training or testing depending on the user preference. The process followed is described in the below flow chart 5.12.

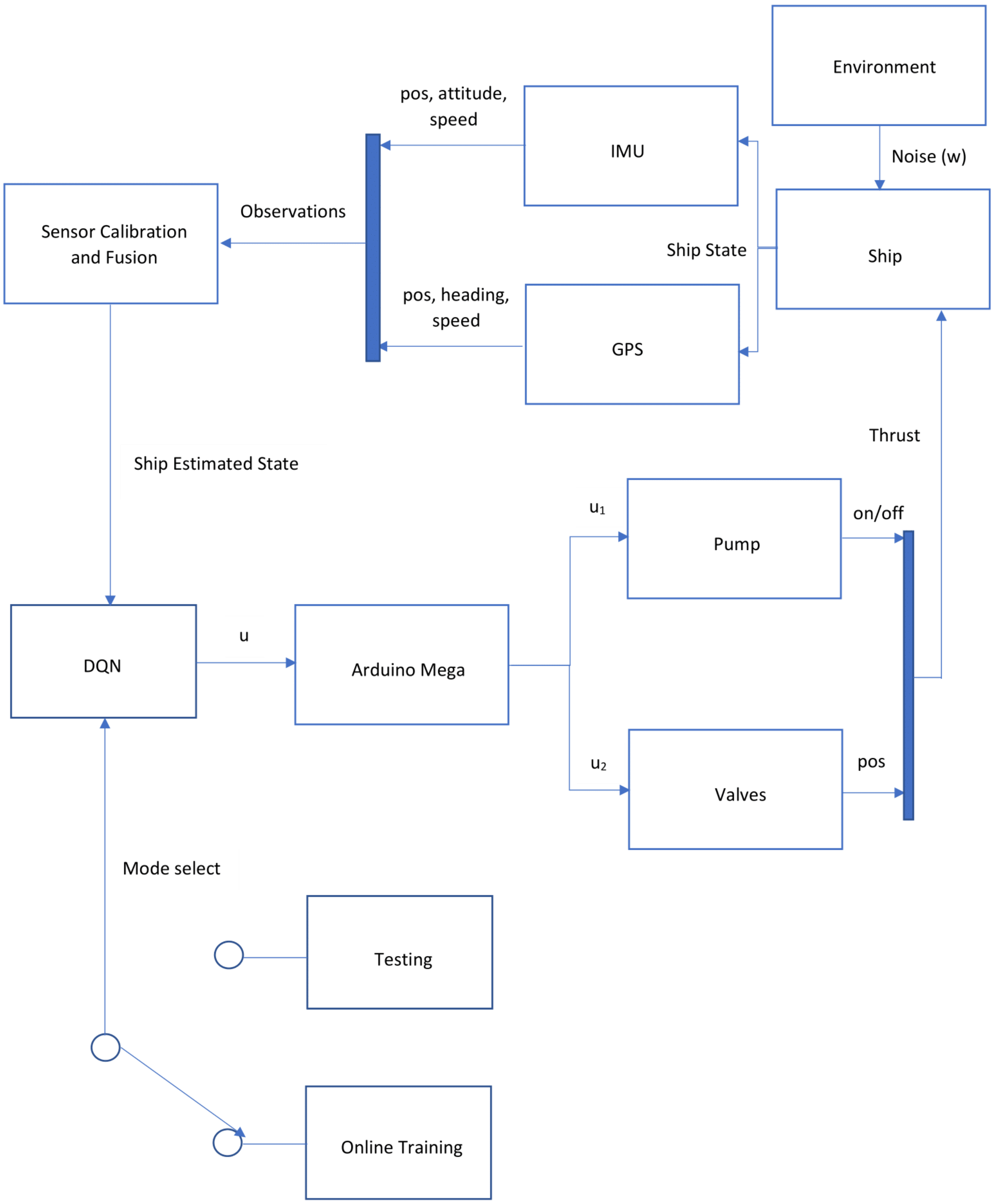


FIGURE 5.12: Control flowchart

6 Dynamic Positioning using Neural Networks

6.1 Neural Networks

Neural networks resemble the action of the human brain by figuring out the correlation between the input and the output data through pattern recognition. They are comprised of an **input** and an **output** layer as a bare minimum but they often incorporate one or multiple **hidden layers** in-between the input and the output nodes. A **node** signifies a place where communication happens and which fires up when it encounters sufficient stimuli. Each of those nodes stores a value at any given time that is derived from data input and a set of coefficients, commonly refered to as **weights**, that either amplify or dampen the input, thereby assigning significance to inputs with regard to the task that the network gets trained in. These input-weight products are summed and then the sum is passed through a node, the so-called **activation function**, to determine whether and to what extent that signal should progress further through the network to affect the outcome, say, an act of classification. If the signals passes through, the neuron has been "activated". A typical representantion of an NN node can be seen in Figure 6.1 whereas a sample Neural net architecture is depicted in Figure 6.2.

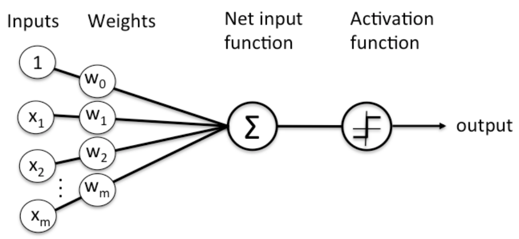


FIGURE 6.1: NN node representation [Pathmind]

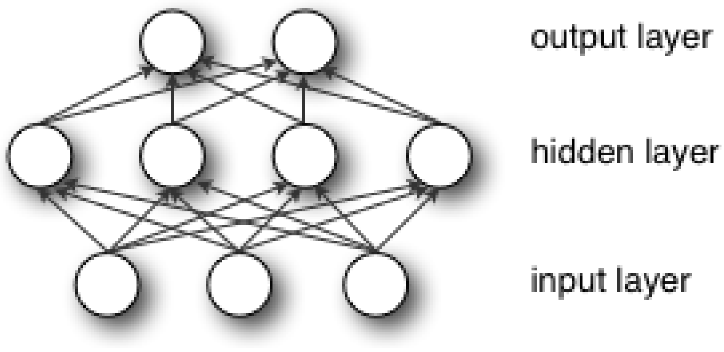


FIGURE 6.2: NN sample architecture (simplified) [Pathmind]

Before its actual use, a neural network of a given architecture must be trained sufficiently. Over the years, three main types of learning were developped in Machine Learning; supervised learning, unsupervised learning and reinforcement learning.

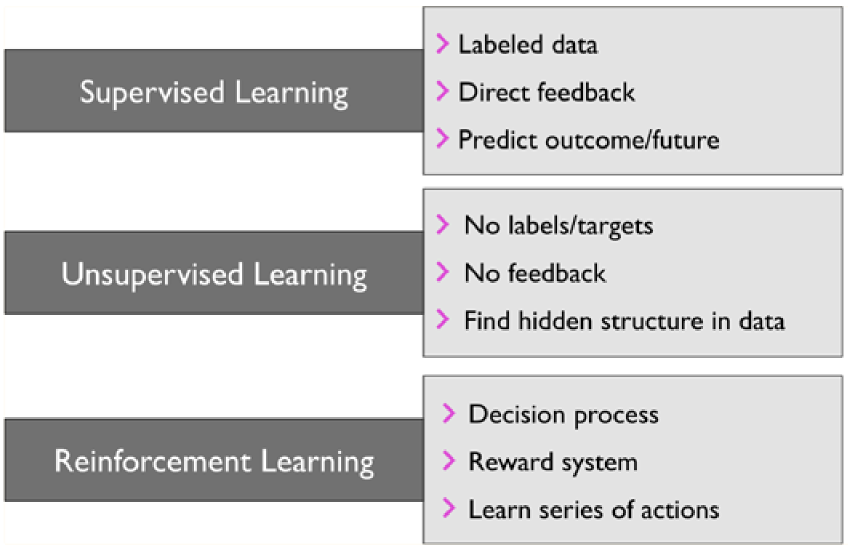


FIGURE 6.3: Types of machine learning [27]

6.1.1 Supervised Learning

Supervised learning is based on the use of pre-labeled training data meaning a competent set of input and the respective output data in order to tune the network weights. The NN can be later used to make predictions about unseen or future data.

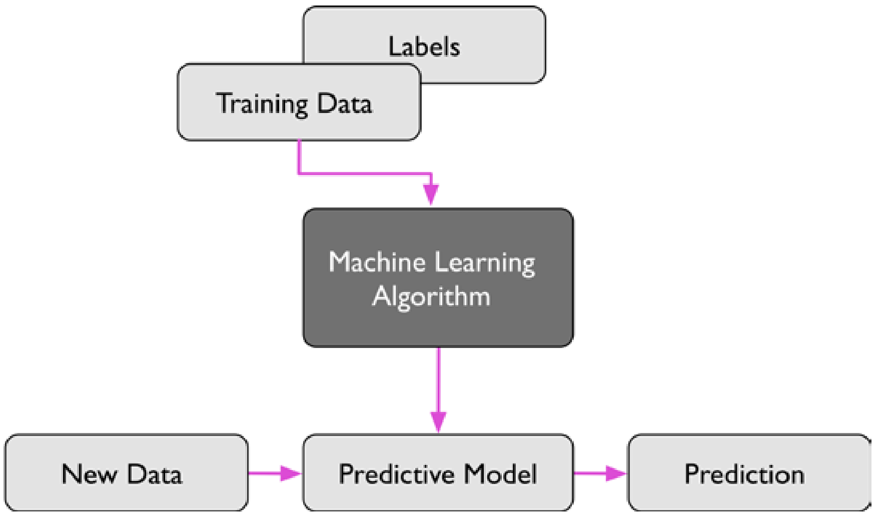


FIGURE 6.4: Supervised Learning [27]

6.1.2 Unsupervised Learning

In contrast to supervised learning, in this case we are dealing with unlabeled data or data of unknown structure. The main scope of this type of learning is therefore to explore the data structure at hand and extract useful information regarding a known outcome variable or reward function.

6.1.3 Reinforcement Learning

The third type of machine learning and the main focus of this thesis is reinforcement learning. In this case, an agent is deployed to a specific environment with the goal to improve its performance at a predefined task over time by learning through its interactions with the environment. A reward function is used to measure how well the network performs in each iteration of the algorithm based on both the state vector and the action the agent decides to take at that given time. Through its interaction with the environment, an agent can then use reinforcement learning to learn a series of actions that maximizes this reward via an exploratory trial-and-error approach or deliberative planning. Thus, the agent develops a policy meaning a strategy/behaviour that makes it receive more rewards over time.

Terminology

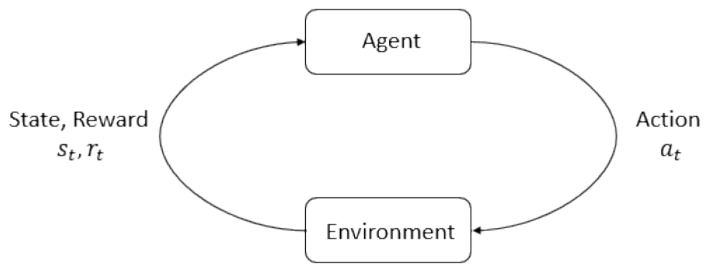


Figure 6.5: Agent-environment interaction loop [22]

As shown in the above Figure 6.5 the main characters in RL are the agent and the environment. The **agent** is defined as the system that at any given time perceives a part of the environment and is called to make a decision on what action to take. On the other hand, the **environment** is the world that the agent lives and interacts with. The environment changes when the agent acts on it, but may also change on its own.

The agent takes a **reward** signal from the environment, that means a measure of how good or bad the current world state is according to the action taken. It is the agent's goal to maximise the cumulative reward after a number of **episodes**, meaning the total number of subsequent actions that kept the world state better than an accepted worst-case condition. This condition is usually defined by the acceptable thresholds in the state values of the system.

A **state** s is defined as the complete description of the environment at any give time. An **observation** o is defined as a part of the total state with possibly omitted information.

The agent uses the observation at any given time and procedes to take an action that will alter the environment and produce a new observation. This procedure goes on until the environment reaches the worst acceptable state.

One issue in RL is how much more or less new information is appreciated over the old one. This is regulated through the use of a factor called the **learning rate** or step size. A learning rate of 1 makes the agent to consider only the new information presented to it whereas a value of 0 makes the agent only retain old information and learn nothing. In the first case, exploration of the environment is preferred exclusively over exploitation of old knowledge. In the later case, the exact opposite happens. In practice, the learning rate is chosen constant and close to 1.

Classification of RL Algorithms

Reinforcement Learning Algorithms can be further classified using the sample taxonomy depicted in the next Figure 6.6.

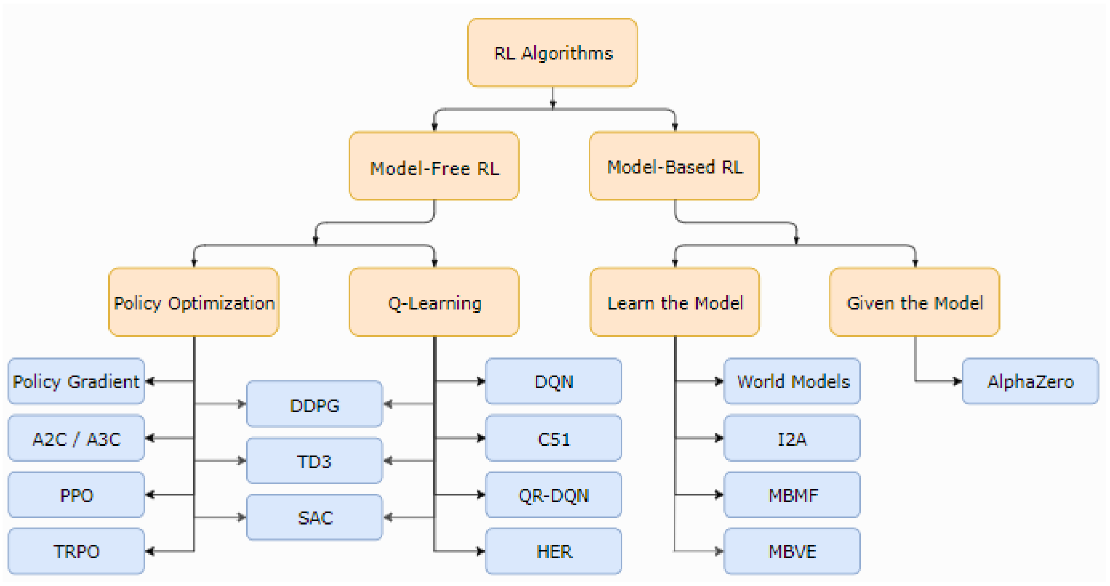


FIGURE 6.6: A non-exhaustive, but useful taxonomy of algorithms in modern RL [22]

As seen from the sample taxonomy, there are two main categories of Reinforcement Learning Algorithms, namely Model-Free (MF) and Model-Based (MB). By using the term model, a function which can predict state transitions and rewards shall be implied.

The main advantage offered by the existence of a model is the given ability to the network to think ahead and explicitely decide between all its different options. If the model is solid then a substantial impovement in efficiency over networks that do not use a model can be seen. The problem with this approach is that it requires a trustworthy and accurate model of the system, otherwise the agent has to fine-tune the model purely from experience. In the second scenario, the agent would perform well with respect to the training model but sub-optimaly in the physical world. Algorithms which use a model are called model-based whereas those who do not are referred to as model-free. The later may forego the potential gains in efficiency by using a model but are in general more safe and easier to implement. The majority of nowadays existing RL algoritms belong to the model-free category of available methods.

6.2 DQN and Reinforcement Learning

As stated above, one of the main focuses of Reinforcement Learning is to define a Reward Function capable of training the network as accurately and fast as possible. Taking this into account, future rewards would need to be discounted by a factor γ. Thus, the Total Reward at time t can be expressed as follows [1]:

$$R_t = \sum_{i=t}^{\inf} = \gamma_t r_t + \gamma_{t+1} r_{t+1} + \cdots + \gamma_{t+n} r_{t+n} + \cdots \tag{6.1}$$

where γ_i denote the discount factors ($0 < \gamma_i < 1$) and r_i signify the instantaneous rewards.Commonly a simplification of the above total reward function (6.1) is adopted by setting $\gamma_i = \gamma^{i-t}$, yielding the following final standard expression:

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots \tag{6.2}$$

6.2.1 The concept of the Q-function

As Reinforcement Learning Problems are in fact sequential decision problems, an estimator for the optimal value of each action is needed. Keeping in mind that our primary concern would be the accumulation of the maximum possible reward when taking the specific action a under a given policy π, an appropriate estimator would be defined as shown in Equation (6.3) [10].

$$Q_\pi(s_t, a_t) = E[R_t | s_t, a_t] \tag{6.3}$$

Strategy π^* is defined as [1] the policy π that maximizes the accumulated reward, meaning:

$$\pi^*(s) = argmax_a Q(s, a) \tag{6.4}$$

Deep Reinforcement Learning Algorithms can thus be categorized as follows based on the approach taken: [1]

- Value Learning Algorithms where the goal is to derive the set of actions that maximize the Q-function (Find Q(s,a) where $a = argmax_a Q(s, a)$),
- Policy Learning Algorithms where the goal is to derive the appropriate strategy (Find $\pi(s)$ and sample a from $\pi(s)$).

6.2.2 Deep Q-Networks

As previously stated, in Value Learning Algorithms the goal is to find the optimal $Q^*(s, a)$. In this task, deep neural networks can be of great help, especially due to their capabilities to estimate nonlinear functions and perform on both discrete and continuous time spaces. In order to train the network, the Q function Q(s,a) can be estimated at any given time and its target value could be expressed as $r + \gamma max_{a'} Q(s', a')$ where r is the accumulated reward up until that moment. We can thus derive a Q-loss function that would be minimized when the network is sufficiently trained.

$$L = E[||(r + \gamma max_{a'} Q(s', a')) - Q(s, a)||^2] \tag{6.5}$$

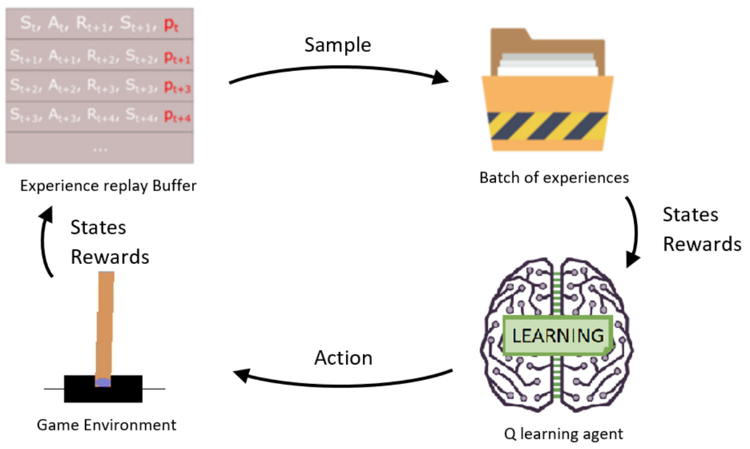
6.2.3 Experience Replay [15]

Just like a human learns from his/her past experiences, Deep Q-Networks use a feature called experience replay. By using it, the learning agent remembers its past experiences that it keeps stored in a memory of specific capacity and repeatedly presents the experiences to its learning algorithm when it finds itself in a similar situation. The memories are chosen as a batch of specific size from the set of the relevant memories to the current episode.

The main advantages of this feature is that the learning process is sped up significantly, the credit or blame of holding a specific memory is assessed continuously and as a result the agent can continuously refresh its experiences to optimize itself to the needs of the task.

Without the experience replay, if an input pattern is not presented to the network for some time, it might need to be relearned afterwards, thus consuming valuable time of training. Experience replay thus offers the capability to refresh previous experiences for sets of input not encountered for some time.

The way experience replay ties in with the overal DQN architecture is depicted in the next Figure 6.7



$S_t, A_t, R_{t+1}, S_{t+1}, p_t$
$S_{t+1}, A_{t+1}, R_{t+2}, S_{t+2}, p_{t+1}$
$S_{t+2}, A_{t+2}, R_{t+3}, S_{t+3}, p_{t+3}$
$S_{t+3}, A_{t+3}, R_{t+4}, S_{t+4}, p_{t+4}$
...
Experience replay Buffer
Sample
Batch of experiences
States
Rewards
LEARNING
Q learning agent
Action
Game Environment
States
Rewards

FIGURE 6.7: Experience Replay

Prioritized Replay [28]

This type of Experience Replay optimizes the choice of relevant experiences to replay but not the way of how old experiences are stored in the memory. The key concept that is being considered to rank the different acquired memories is that some transitions can teach the agent more things than others. Transitions may be more or less relevant, surprising or redundant and some of them may be more useful for the agent in the future than in the present. In the regular experience replay, memories are not replayed in the exact same order as they were experienced but are chosen randomly according to the input itself. However, prioritized replay optimizes the selection of memories and discourages the frequent choice of the same transitions.

6.3 Available Tools

6.3.1 Tensorflow

FIGURE 6.8: Google's Tensorflow Platform

Tensorflow is an open-source machine learning platform released by Google in 2015. Since then, its reputation has only been growing and it has actually been used in a variety of Machine Learning and Neural Network subfields. Its production started in 2011 under the name DistBelief to accomondate the computational needs of researchers and engineers working the Google Brain team within Google's Machine Intelligence Research organization to conduct machine learning and deep neural networks research. The system is general enough to be applicable in a wide variety of other domains, as well.

Tensorflow has full included support for C++ and Python but some of its features can be used also by other languages such as JavaScript and Swift. It can also be implemented on both GPU and TPU computational modules. A lightweight version of the platform has also been made for the case of lower-end devices such as the RPi.

6.3.2 Keras

FIGURE 6.9: Keras API

Keras is one of the most well-known API's written on top Tensorflow alongside PyTorch, MXNet,caffe with the focus of simplifying the workflow needed by the human researcher or engineer to pass from an initial idea to its implementation. It classifies itself as an approchable, highly-productive interface for solving machine learning problems, with a focus on modern deep learning. It also provides essential abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity.

6.3.3 OpenAI Gym

FIGURE 6.10: OpenAI's Gym

In the field of Reinforcement Learning the main platform used to enable simulation is OpenAI's Gym. Gym has been used in a multitude of problems ranging from Control Systems and Robotics to Retro PC Games and Text Generation.

A gym environment is comprised of four main functions namely:

- **init()**; This function initialises the environment by setting the system parameters as well as the action and state space type and morphology.
- **step()**; The main focus of this function is to take an action variable and return a list of four things — the next state, the reward for the current state, a boolean representing whether the current episode of our model is done and some additional info on our problem.
- **render()**; This function renders the simulation environment in each one of the different episodes using OpenGL
- **reset()**; This function is responsible for resetting the environment if its previous run has ended. The states, actions and other variables of the environment are reset to the start state.

An overview of a Gym's environment structure is given in the next Figure 6.11.

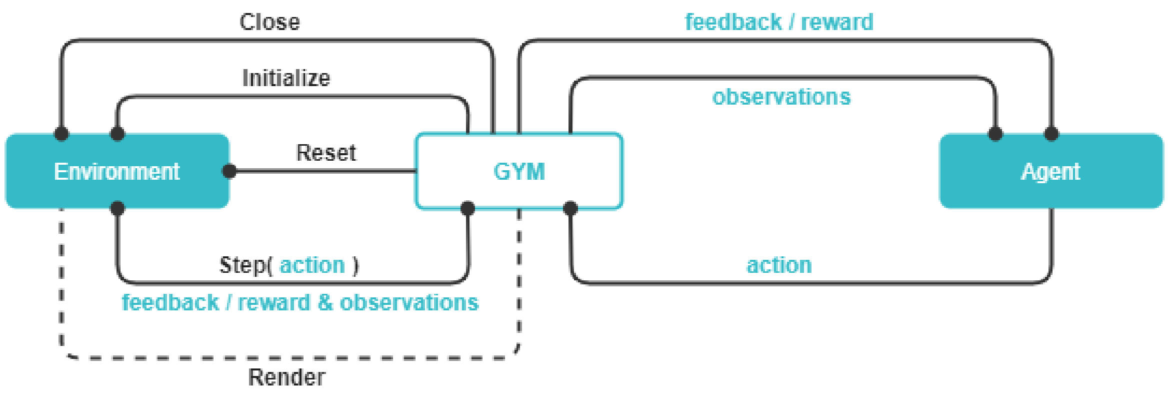


FIGURE 6.11: Gym's Environment Structure [**TowardsDataScience**]

6.4 The LINCOLN Environment

As no previous Gym environments were preoccupied with the Dynamic Positioning of vessels at sea, the initiative was taken to create one from scratch. Due to the fact that the surge motion is underactuated and thus positioning in surge would require maneuvering the vessel rather than stabilizing it, only the sway and yaw degrees of Freedom are controlled.

The environment was named after the European Project it was a part of and uses the equations of Nomoto to perform the necessary simulations. In the **init()** function, the Nomoto coefficients and system characteristics derived in Chapters 2 and 3 are set and the State and Action spaces are created as follows:

- **State space**; A continuous time-space that incorporates data such as the sway position, velocity and acceleration and the yaw angle, rotational velocity and acceleration that can all be found using the on-board sensors of the vessel as described in Chapter 5. Both the sway position and the yaw angle are constrained by a limit of $\pm 2.5m$ and a limit of $\pm 15deg$ respectively.
- **Action space**; As the thruster system can only perform discrete actions, the type of this space is also chosen Discrete. The actions that can be performed at any given time are sxay to the left(0), sway to the right (1), clockwise rotation (2), counter-clockwise rotation (3) and nothing (4).

While in the **step()** function, as the state of the vessel is known and an action has been decided by the DQN involved, a sea current disturbance is calculated due to the effect of regular waves that tend to drive the vessel off the specified position. Since both the disturbance force and the actuation force are known, the Nomoto equations can be used to solve for the new accelerations and by Euler integration the new velocities, sway position and yaw angle of the vessel. If we remain below the specified thresholds for both the angle and the position, meaning that the boolean variable **done** is False then the simulation continues in the next episode whereas in any other case, the environment is reset and a new simulation process is initiated. It must be noted that a **reward** of one is given for every successful episode and one of zero in every other case. Finally, in both cases, a set of variables containing the new state of the environment, the done boolean value and the reward retrieved in this episode is being returned to the main script.

In every episode, the environment is rendered on-screen as an OpenGL window by the use of the dedicated **render()** function. All the graphics have been also created from scratch for the purposes of this Thesis. In Figure 6.12 below, an instance of the develloped graphical environment is shown. The values of the main states, the disturbance's sea height and angle of attack as well as the action performed in each episode is present on the upper left of the window. The vessel is portrayed by the black rectangle and the red axes serve as a depiction of the yaw angle threshold. The encompasing circle has a radius of 2.5 m signifying that if passed, the simulation has to be reset. The thrusters are depicted by the red rectangles that turn green when activated according to the decided action course in this specific episode.

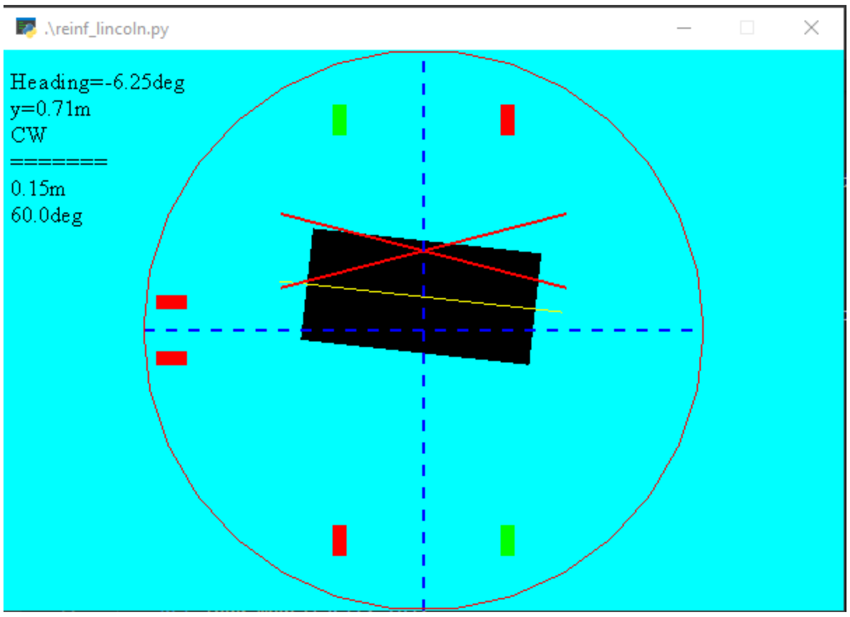


FIGURE 6.12: LINCOLN Gym Environment

The **reset()** function assigns a uniform random value of ±0.5 to all of the involved states in order to simulate different initial conditions for the problem. This function is called before each new simulation, including the very first.

6.5 A DQN approach for Dynamic Positioning

The agent deployed in the LINCOLN Environment presented above uses a DQN approach to make decisions regarding the different thruster actions that need to be taken to stabilise the vessel within the chosen thresholds for both the heading angle and the sway position.

The structure of the network is composed of 1 input layer of 6 nodes derived from the observations at each moment in time,2 hidden layers of 128 and 32 nodes respectively and an output layer of 1 node that produces discrete action values that correspond to the action space mentioned in the above paragraph. The relu activation function is chosen for all the layers except the output which uses a linear activation scheme. An overview of these functions can be seen in the next Figure 6.13.

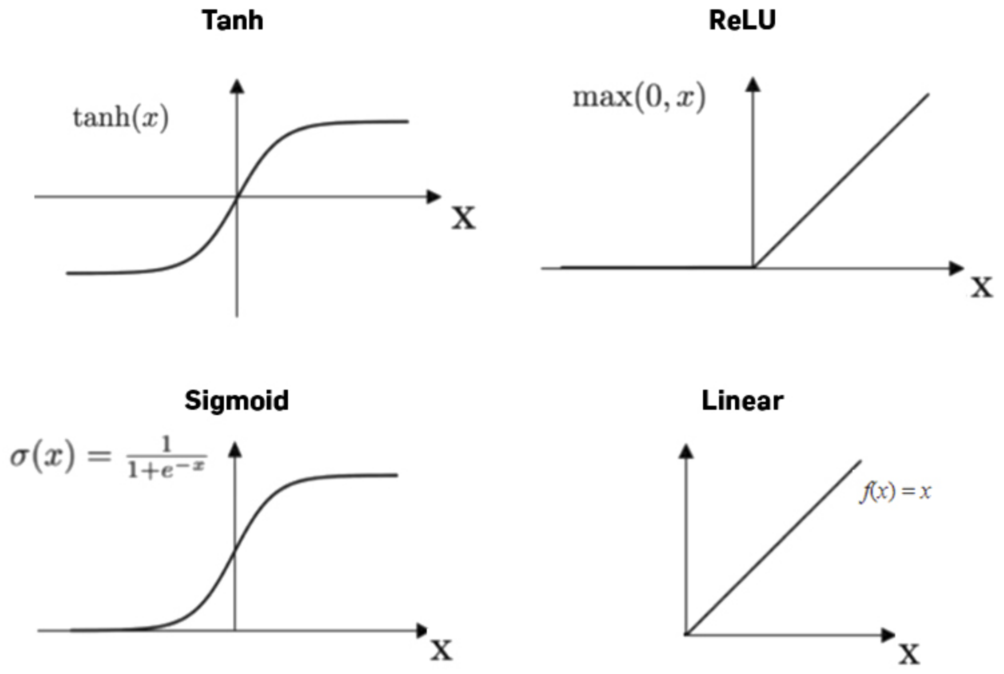


FIGURE 6.13: Standard Activation Functions

A memory size of 4 GB is appointed to the algorithm for experience storage in the simulation stage in order to train the network as rigolously as possible beforehand. However, this would be brought down to 0.5-0.8 GB to continue the online training at sea trials using the Raspberry PI. Alternatively, sea trial data would need to be collected for a sufficient amount of time and stored in log files. Those log files would then substitute the Nomoto equations in the LINCOLN Environment and the network would be trained with sufficient memory in the actual operational conditions offline. The relevant memories are chosen in a batch of 40 when the experience replay function is required.

The choice of the optimizer of the derived network was the Adam one, mainly due to its frequent use in the other control system environments made using the Gym Module. The learning rate of the optimizer was set to 0.0001 to favor constant and slow convergence to the optimal solution, thus avoiding suboptimal solutions that could be derived using a larger learning rate. A discount factor γ of 0.9 was chosen for future rewards and as for the learning rate of the network, in order to avoid confusion with the learning rate of the optimizer, it is going to be referred to from this point onwards as the **exploration rate**. The exploration rate in this case gradually diminishes from a value of 1 to a value of 0.01 with a decay factor of 0.995 as the simulation progresses.

6.6 Neural Network Training

The Network training is performed using computer simulations in the Lincoln Environment (offline training). Waves with a height of 0.05, 0.1, 0.15 m and a random wave angle in the range $[0,\pi]$ are used in the creation of the agent's environment. The agent is required to stay in position, meaning not to pass the asigned thresholds for a total of 5 minutes in each environment before passing to the next. The trained network is saved after each change of the environment while the network's memory is refreshed throughout all the algorithm runs. The training is stopped by user input after visual inspection of the network performance with the help of the rendered simulation window. The training process is illustrated in the next Figure 6.14.

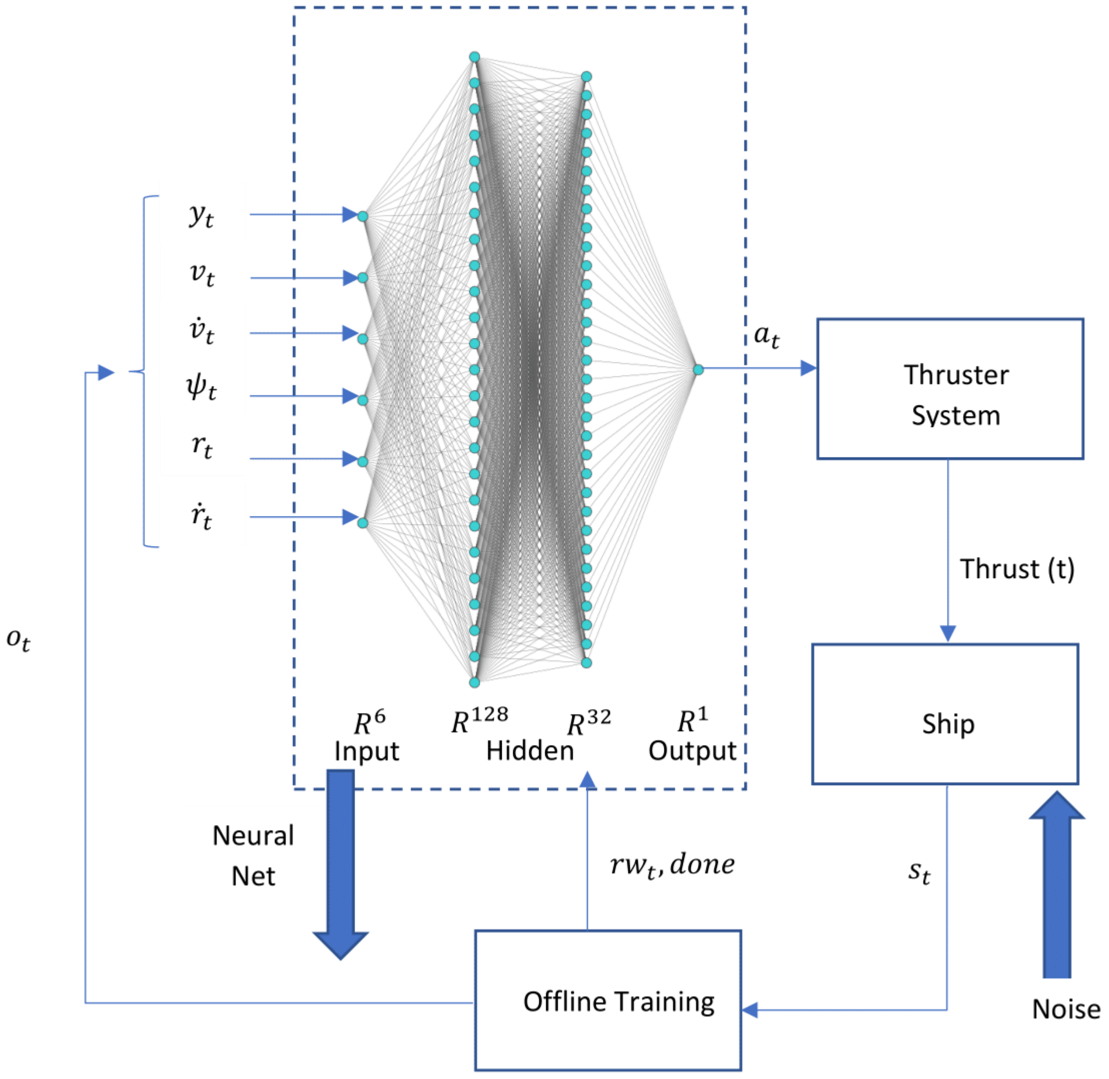


FIGURE 6.14: Neural net Architecture and Training Process

In the above Figure, a_t denotes the actuation command (network output), s_t denotes the state vector of the agent, o_t the observation vector which contains the sway position y_t, sway velocity v_t, sway acceleration $\dot{v}_t$ as well as the yaw angle ψ_t, speed r_t and acceleration $\dot{r}_t$. The reward at each time step is symbolised as rw_t and the boolean *done* denotes the end of the training with the current configuration and the passing to the next environment. Finally, the network architecture is depicted graphically.

7 Simulation Results

7.1 Neural Network Testing

After training, the results of the pre-training of the network need to be put on test. For that reason, three distinct case studies were performed where the time histories of the sway position, yaw angle and actions performed were all recorded. As it is evident from the below Figures 7.1-7.3,7.5-7.7,7.9-7.11, the time limit of a minimum of 5 min for the DP is met in all of those cases. Moreover the frequency of choice for each distinct action is shown in Figures 7.4, 7.8, 7.12. The time histories are plotted only for the first 10 minutes of DP, however the bar graphs incorporate all the actions performed by the system until one of the specified thresholds is surpassed.

The test cases that are going to be considered are given in the below Table 7.1. All of the below considered waves are taken as regular with different heights and wave angles. The two first wave cases belong to the Sea State 2 group whereas the third case belongs to the Sea State 1 group. This choice was based on benchmarking the network using three distinct case studies, an easy, an intermediate and a hard one. This also allows for grasping a more broad picture of the network performance.

	Hs [m]	Initial Wave angle [deg]	Sea state
1	0.15	30	2
2	0.10	50	2
3	0.05	-10	1

Table 7.1: Network Test cases

Test case 1

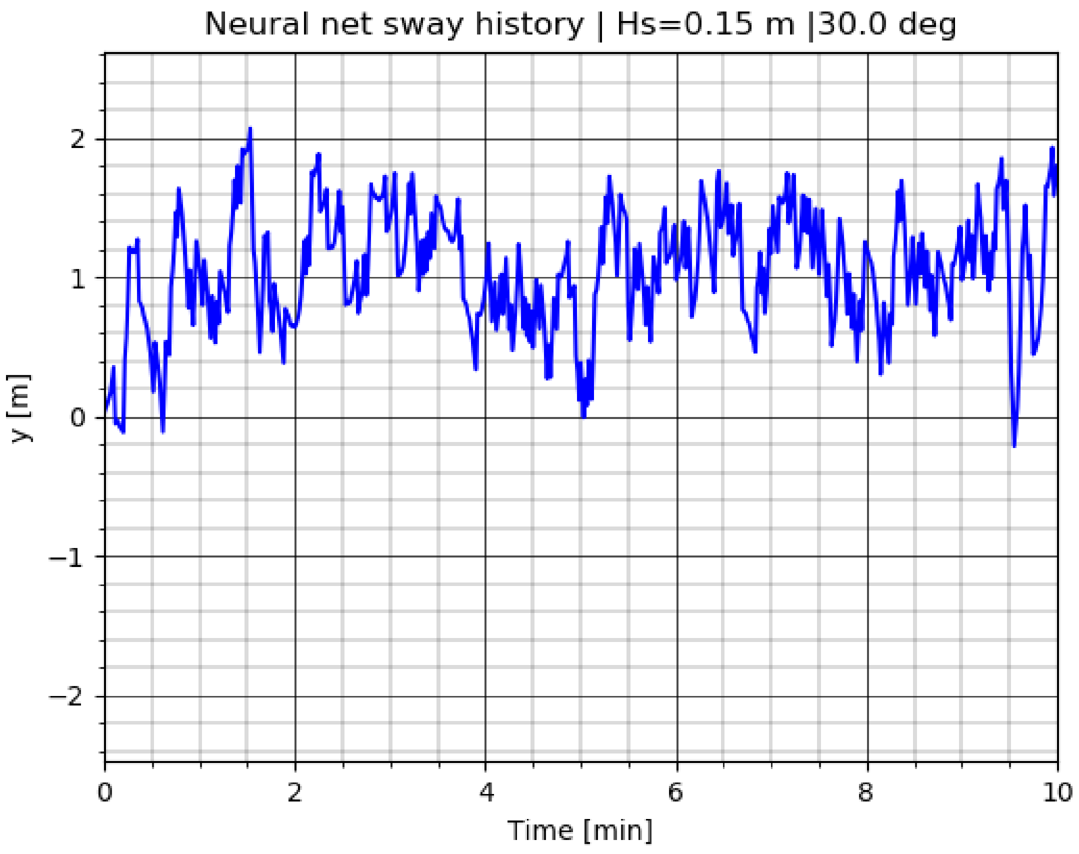


FIGURE 7.1: Sway time history (H=0.15 m)

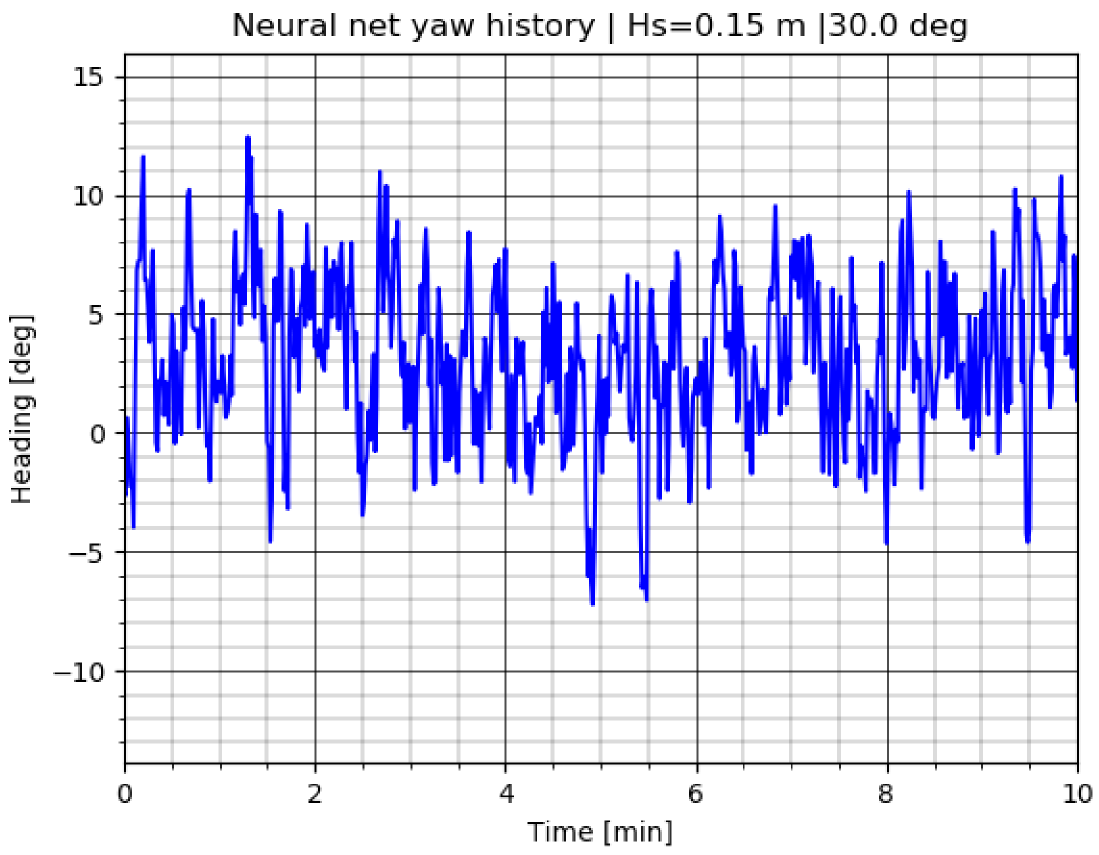


FIGURE 7.2: Yaw time history (H=0.15 m)

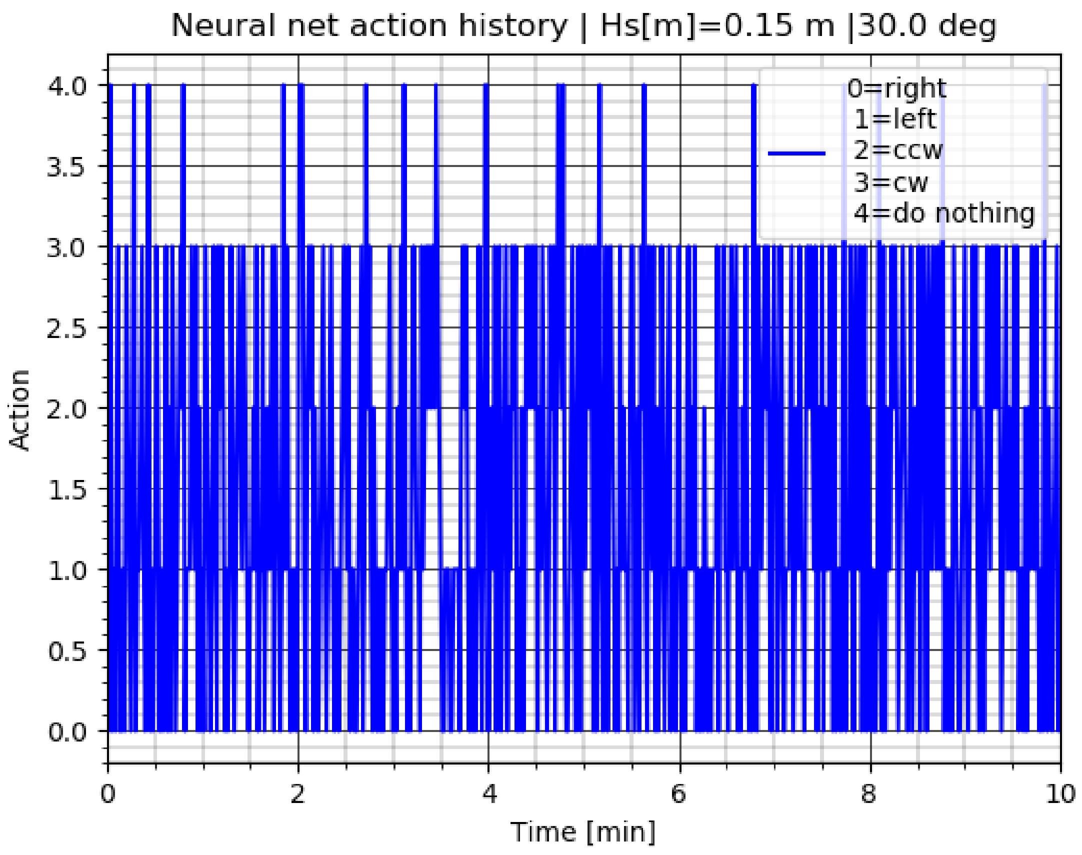


FIGURE 7.3: Actions time history (H=0.15 m)

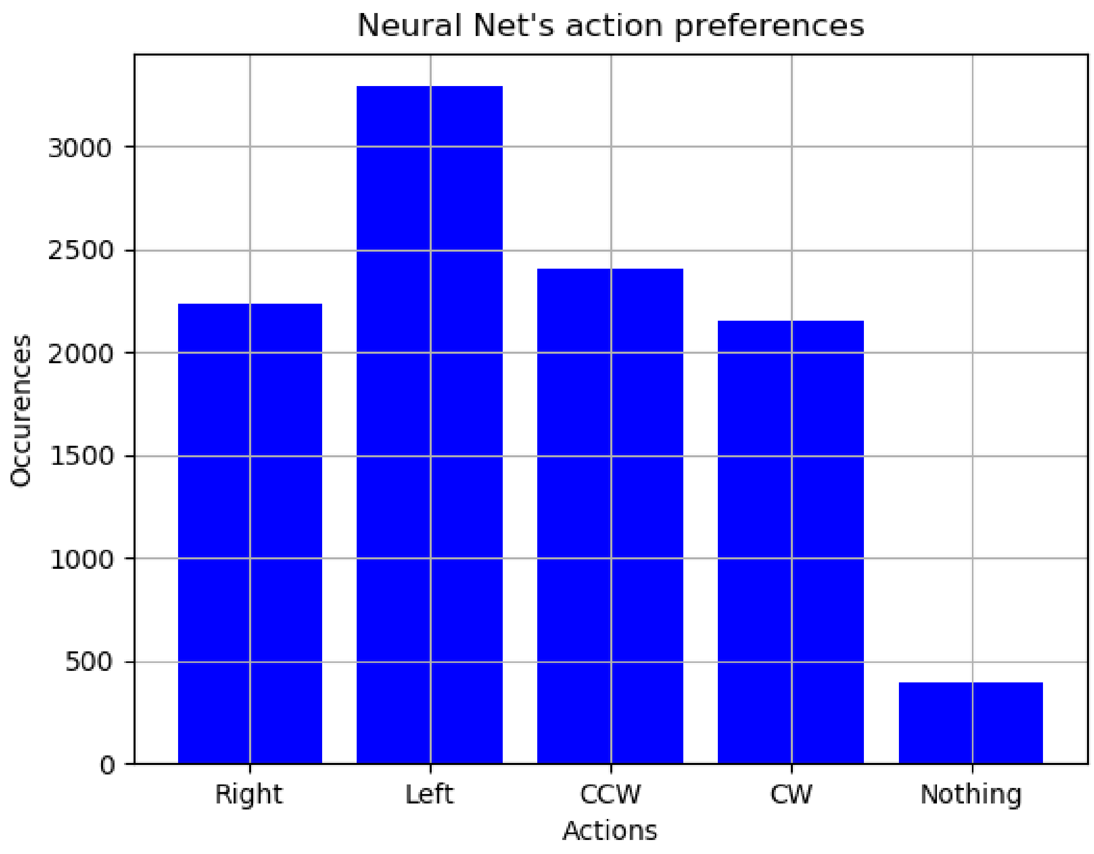


FIGURE 7.4: Actions performed bar graph (H=0.15 m)

Test case 2

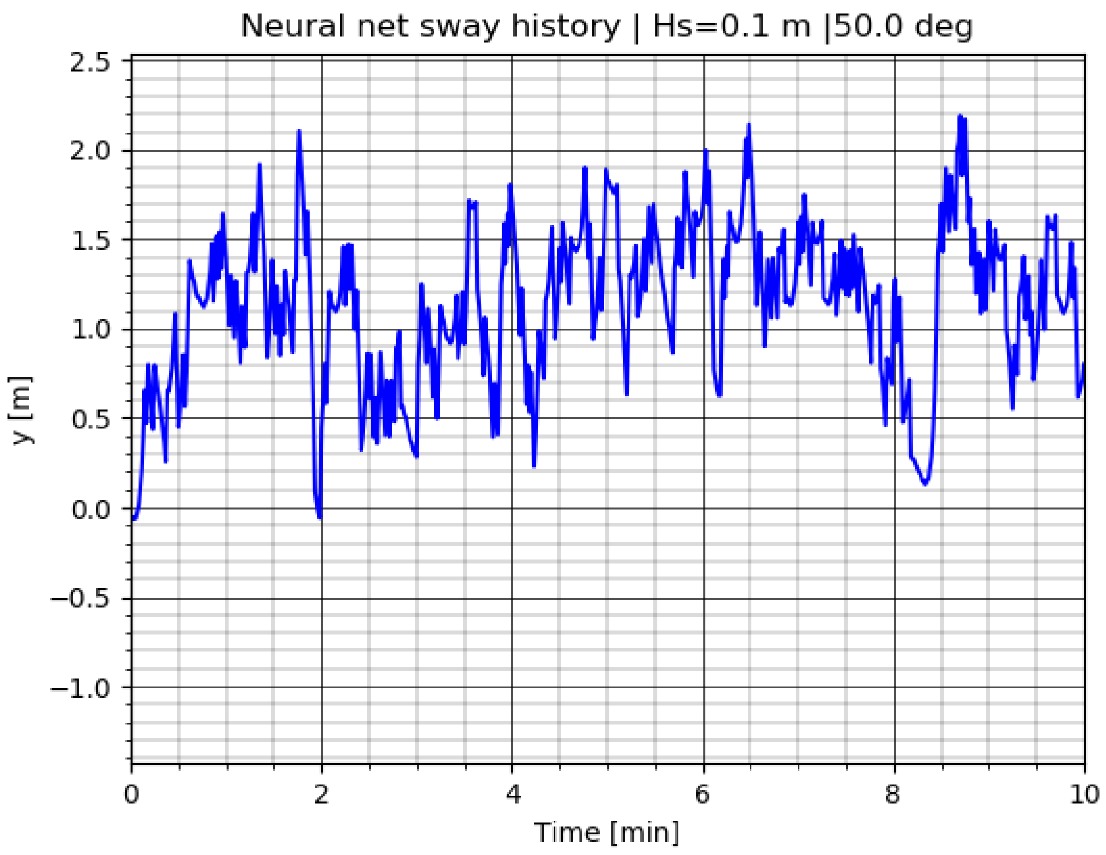


FIGURE 7.5: Sway time history (H=0.1 m)

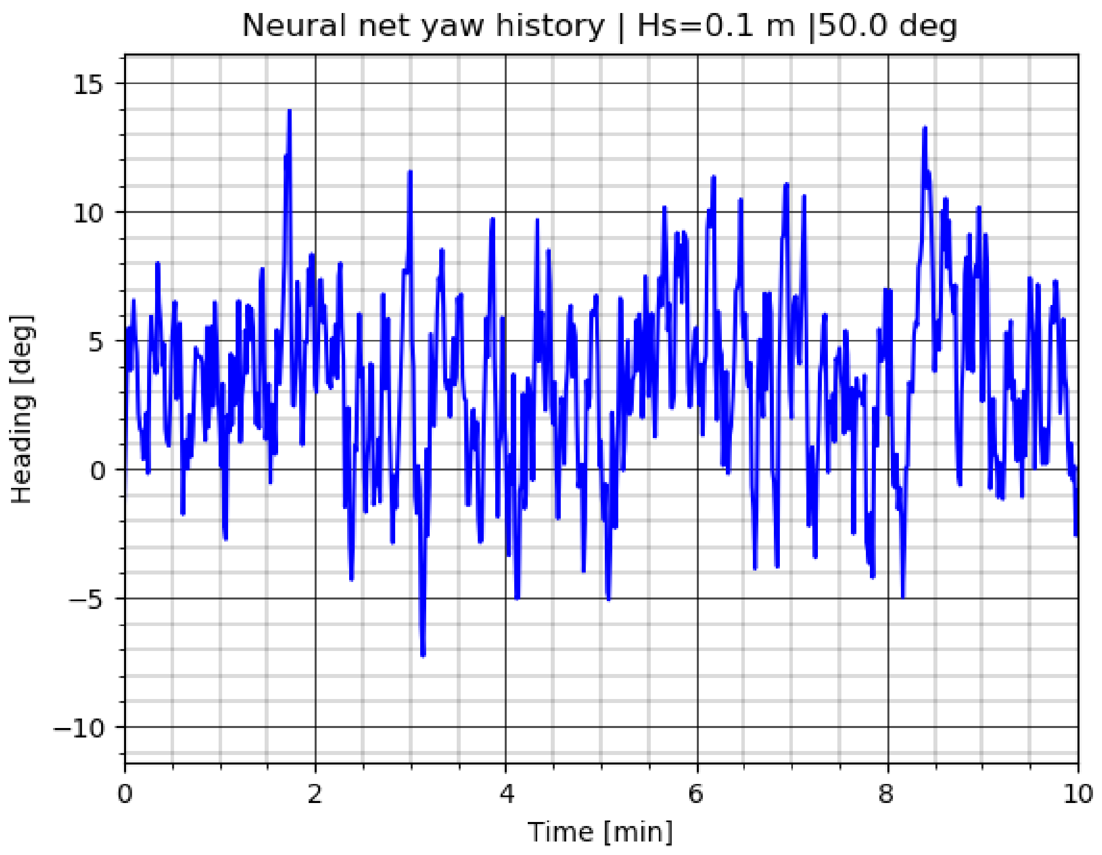


FIGURE 7.6: Yaw time history (H=0.1 m)

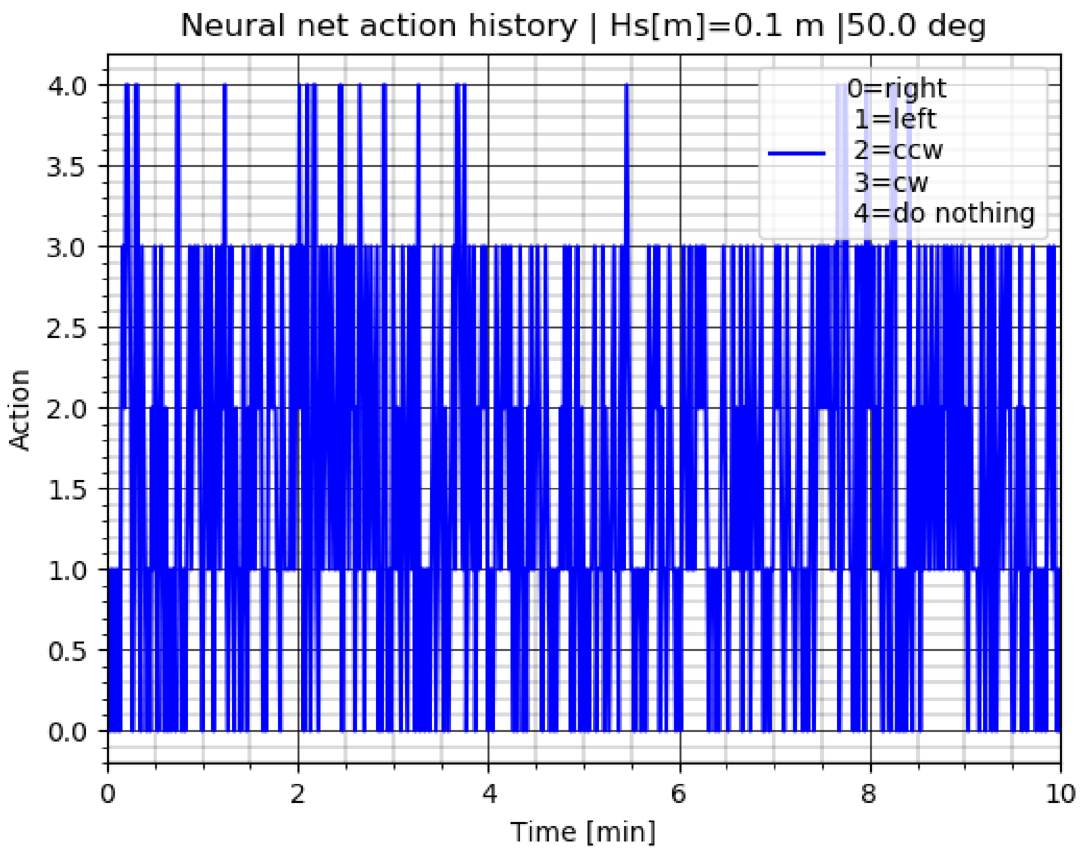


Figure 7.7: Actions time history (H=0.1 m)

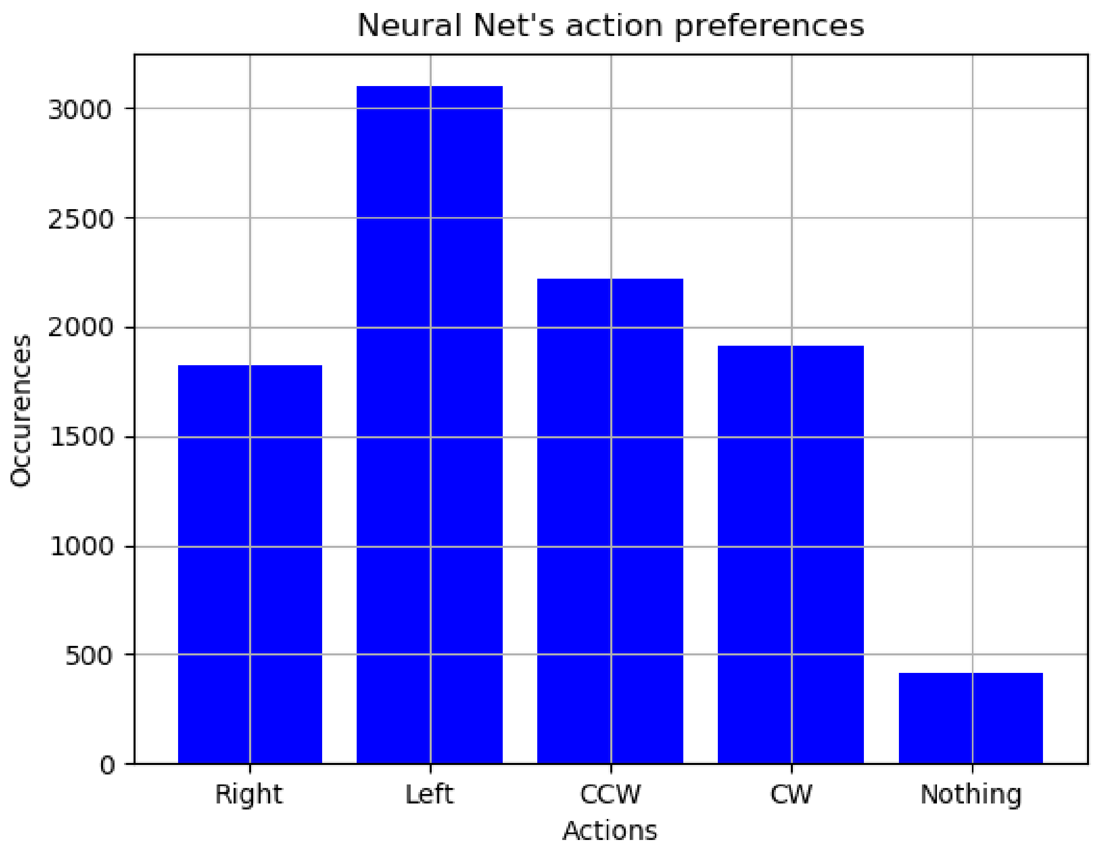


FIGURE 7.8: Actions performed bar graph (H=0.1 m)

Test case 3

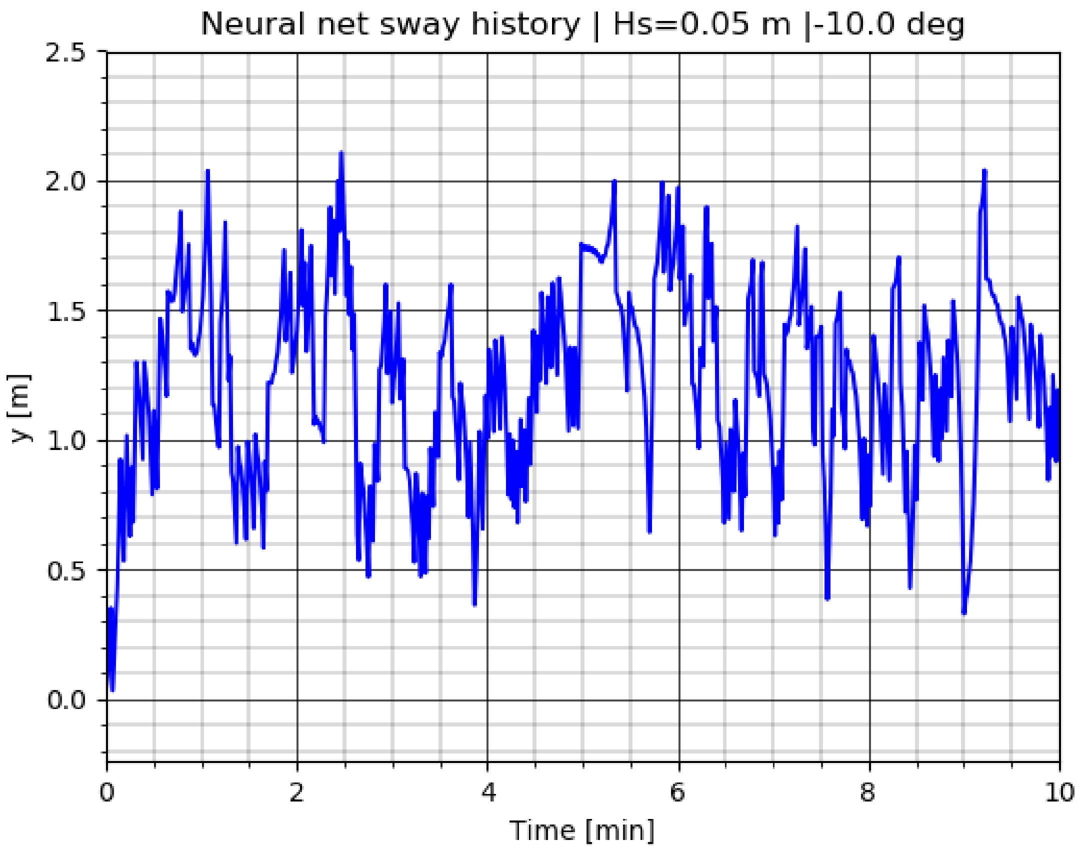


FIGURE 7.9: Sway time history (H=0.05 m)

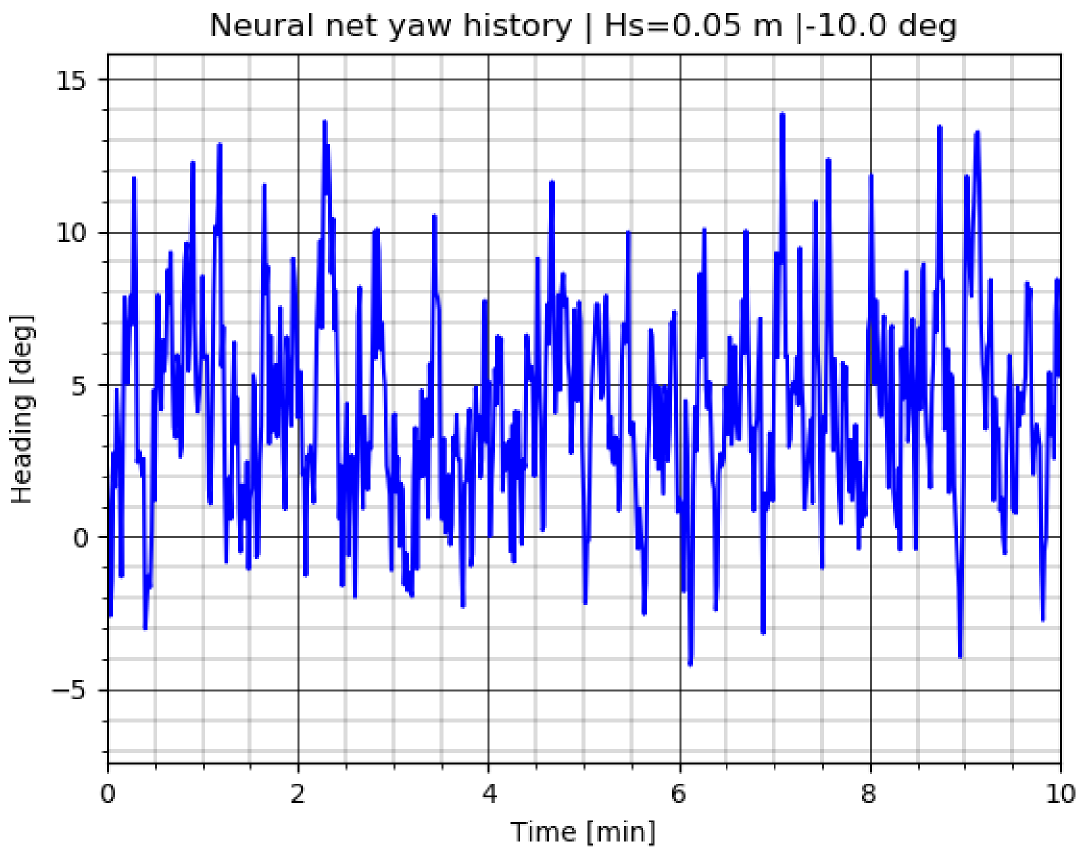


FIGURE 7.10: Yaw time history (H=0.05 m)

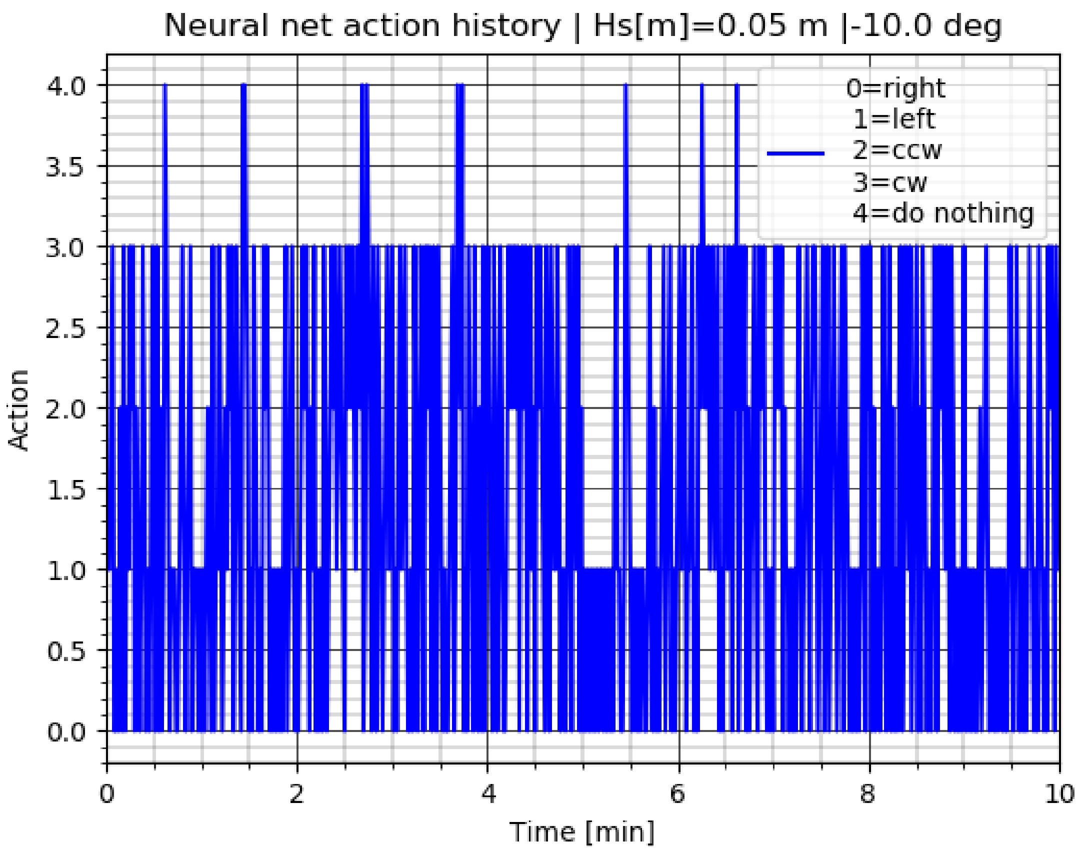


FIGURE 7.11: Actions time history (H=0.05 m)

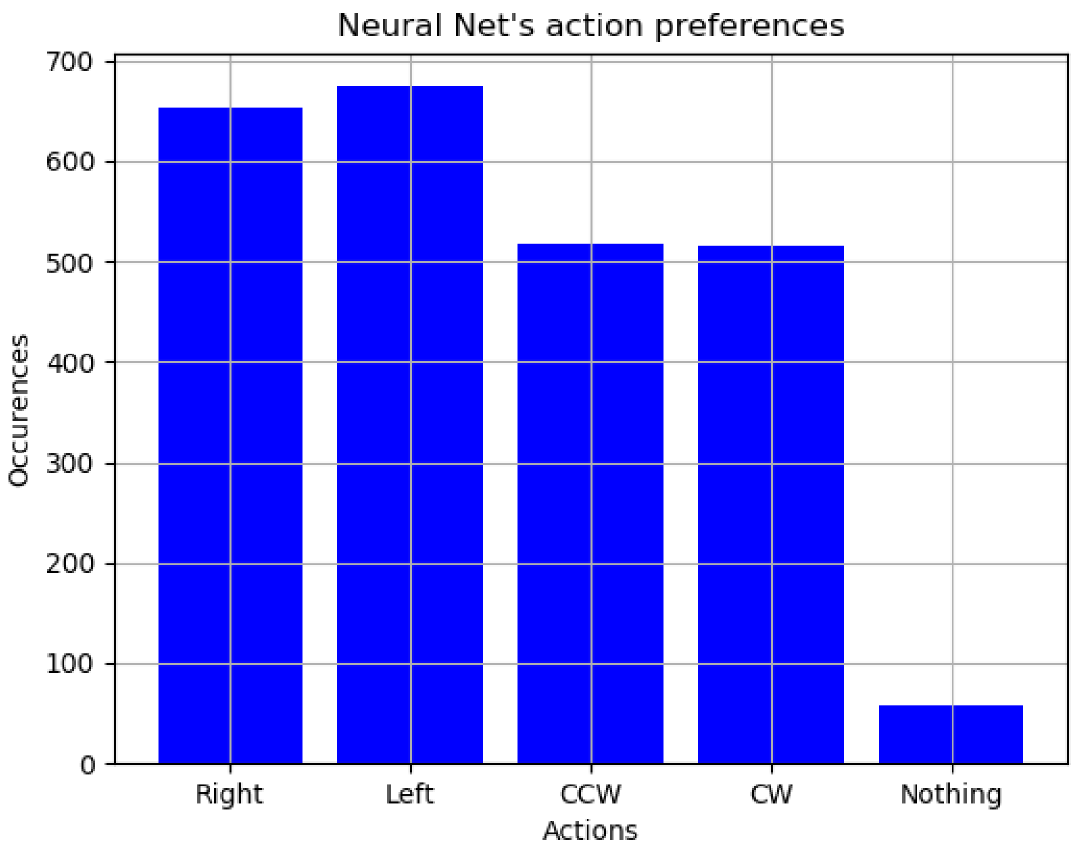


FIGURE 7.12: Actions performed bar graph (H=0.05 m)

As seen from the above time histories, the network manages to position the vessel effectively after training, for a more than sufficient amount of time. It must be noted that the time histories are recorded only for the first ten minutes of positioning, whereas the vessel takes a longer time to lose its locked position. The time is of course expected to be lower in the actual testing phase at sea due to the simplifications made to our model, but the large margin already built in is expected to compensate this loss to a certain degree.

As mentioned in the end of the previous Chapter, the network has been trained for wave angles in the range of $[0,\pi]$, thus explaining its preference of moving to positive sway positions (port side of the vessel) in order to be as far away as possible from the boundary starboard sway threshold (-2.5 m). This tendency is seen even in the third case where a negative wave angle is present but, nevertheless, the ship manages to stay in position.

Regarding the bar graphs, as expected, the network prefers to engage the two couples of sway thrusters as the sway disturbance is actually greater than yaw. Also, in the third test case, the 'right' and 'cw' actions are performed more regularly due to the use of a negative initial wave angle. The negative angle also seems to lead to a more scarce employment of the 'nothing' action as the invocation of the aforementioned actions is needed more frequently.

Finally, the actions time histories graphs appear very dense as the network takes an action decision every 1 sec.

7.2 Performance Assessment

Although time histories often provide clear insight regarding what is achieved using the trained network, they can only be extracted for a handful of inputs. In order to summarize the performance achieved as well as put to the test as many case studies as possible, the use of the polar diagrams shown in Figures below was opted for. Each circle with its center at the origin has a radius which represents the time in minutes during which DP was achieved without passing any of the alloted position and angle thresholds. Each point is placed upon a line rotated by the wave angle or the wave height translated into an assigned angle in reference to the x-axis and is colored green if the time limit of 5 min is surpassed and red if the opposite holds true. The polar diagrams 7.13-7.15 are extracted for variable wave angles and standard wave heights whereas the diagrams 7.16-7.20 are extracted for variable wave heights and standard wave angles.

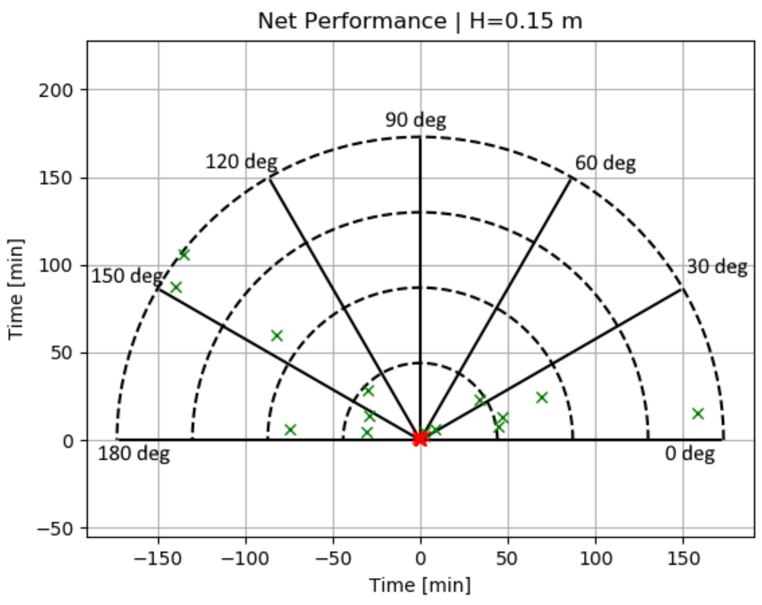


FIGURE 7.13: Neural Net Performance (H=0.15 m)

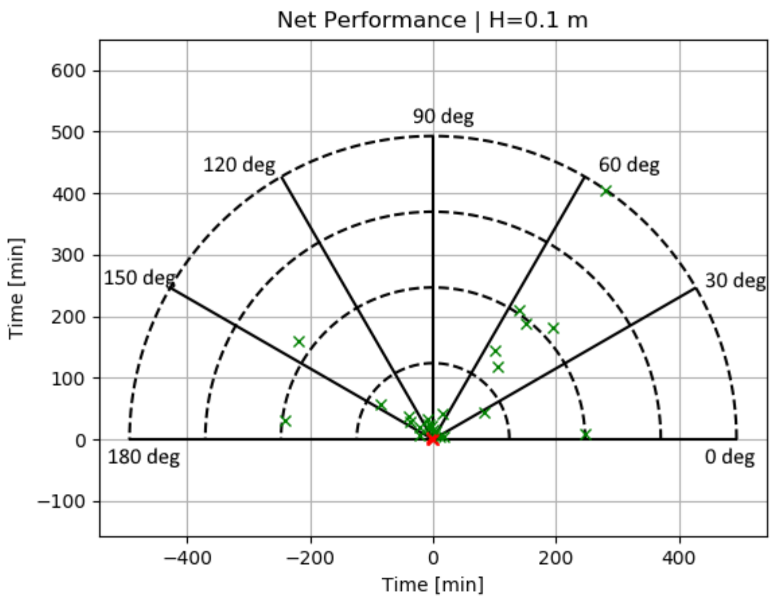


FIGURE 7.14: Neural Net Performance (H=0.1 m)

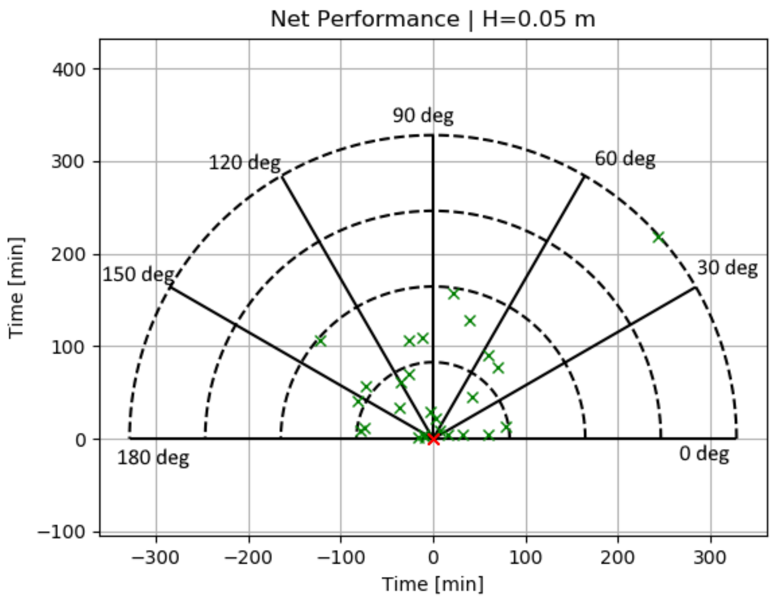


FIGURE 7.15: Neural Net Performance (H=0.05 m)

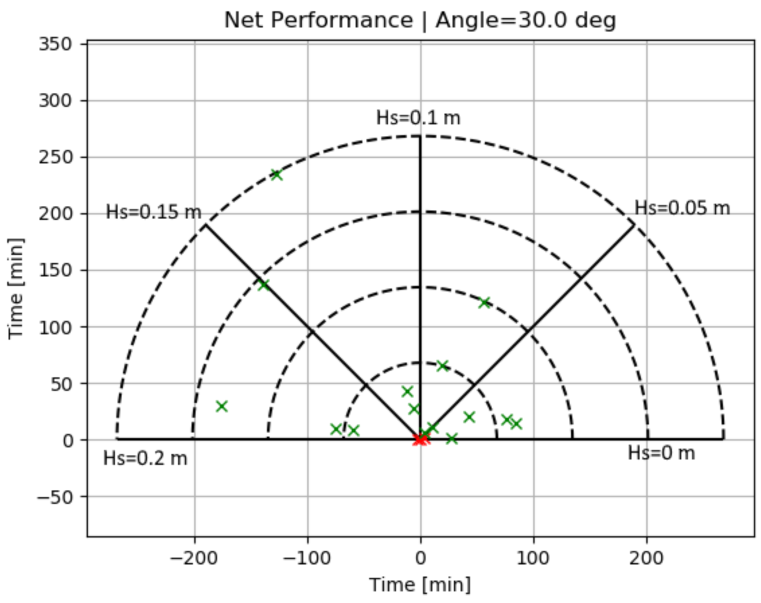


FIGURE 7.16: Neural Net Performance (wave angle=30 deg)

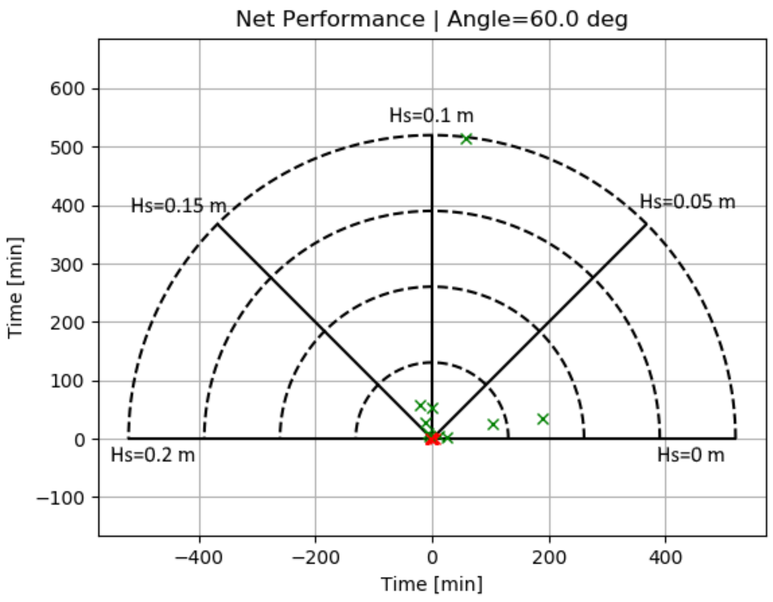


FIGURE 7.17: Neural Net Performance (wave angle=60 deg)

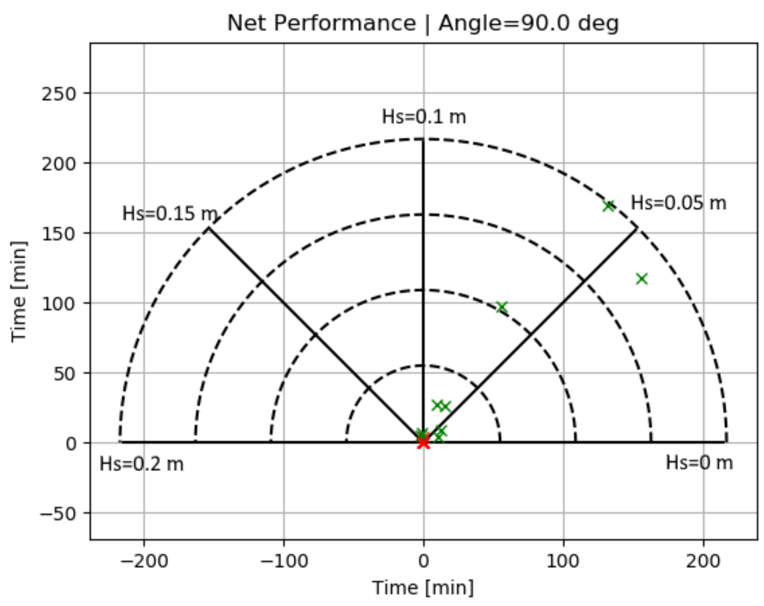


FIGURE 7.18: Neural Net Performance (wave angle=90 deg)

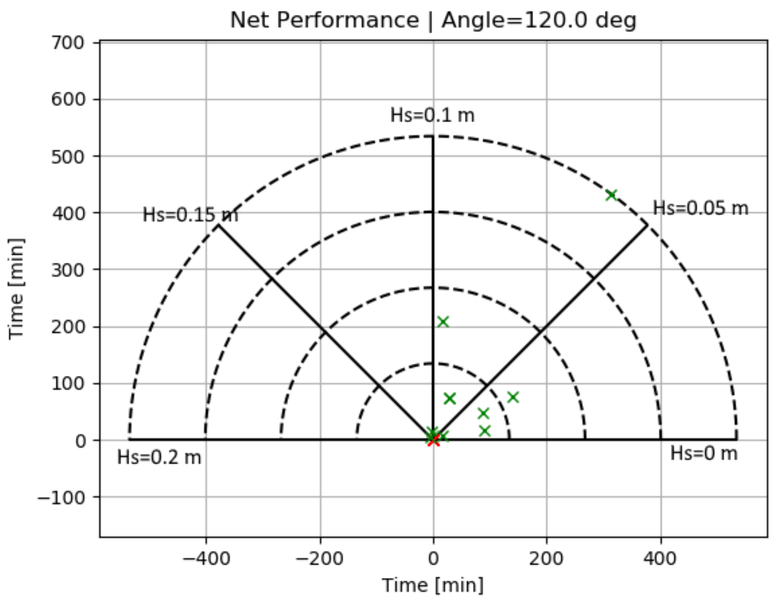


FIGURE 7.19: Neural Net Performance (wave angle=120 deg)

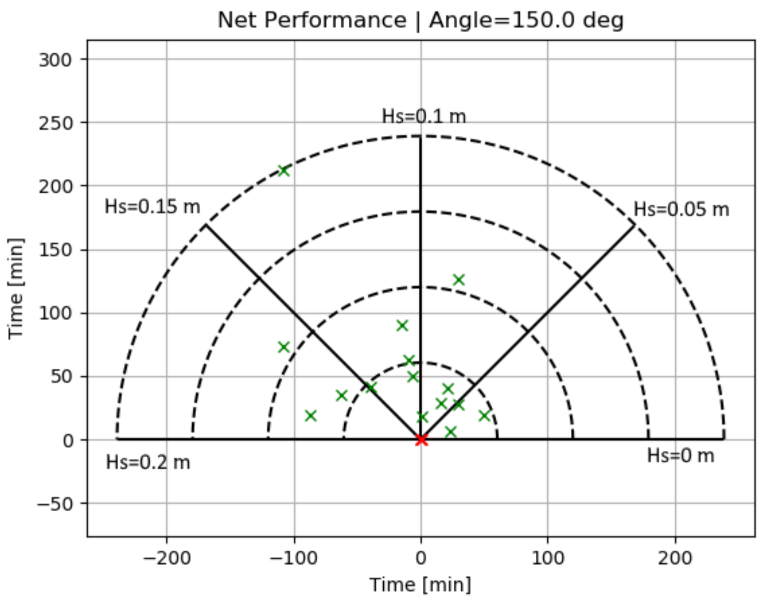


FIGURE 7.20: Neural Net Performance (wave angle=150 deg)

As can be seen from the above Figures, the performance of the initially offline trained network can be considered acceptable for conducting the first sea trials. However, the performance seems to drop between 60 and 120 degrees and thus further training of the network in this area would most definitely improve the end results. It must be noted that although the network was not trained to cope with a wave height above 0.15 m, DP is achieved for sway and yaw in higher wave heights for wave angles of 30 and 150 deg. This could be attributed to the smaller sway component of the acting force and the relatively larger resultant surge force. Thus, in these cases, the problem is tranferred to the surge degree of freedom which afterall is not actuated by the network. Finally, as expected, the network achieves higher performance for lower wave heights as the resultant forces are easier to be counteracted.

The subsequent online training is expected to improve the overall results gradually but, as seen from the above diagrams, a solid initial training seems to have been accomplished.

7.3 Remarks

As seen from the results presented in the previous Sections, the pre-training of the neural network can be considered satisfactory. The model is expected to require less online training on-board the vessel but sea trials can not be avoided under no circumstances. The performance diagrams presented in the previous paragraph are only indicatory of the network's success but they can not guarantee it. After all, the mathematical modelling of the system itself has its own integrated errors and the sea environment is generally stohastic and not deterministic as the one chosen for the training. However, the burden of training the network from scratch on-board the vessel using the RPI has been avoided for sure and the simulation purposes have been well achieved. Finally, the use of Reinforcement learning for DP presented in this Thesis has been one of the first but surely looks promising for the future as witnessed by the extracted results.

8 Conclusions and Future Work

8.1 Conclusions

This thesis examines the feasibility of designing a control system capable of dynamically positioning an EER Vessel at sea. The study had mainly two parts, the first one being the deployment of a Deep-Q Network capable of controlling the vessel and the second one being the creation of a plausible and complex simulation environment that would be used to pre-train the network, thus relieving the onboard electronics from doing all the training online. The reinforcement learning approach was preferred over a Recurrent Neural Network scheme mainly due to the lack of enough data to conduct the training and the parallel need to fine-tune and calibrate the on-board sensors.

The dynamic positioning solution was verified at the simulation level using three distinct wave heights belonging to Sea states 1 and 2 and waves of three different directions. As for the disturbance, only slowly vaying drift regular sea forces were implemented due to the fact that these alongside the wind forces affect the positioning and that the lateral area of the hull is relatively small resulting in wind forces of negligible magnitude (the hull is approximately on the boundary layer). The goal was to maintain the desired position over an interval of 5 minutes and with the specified accuracy. The position accuracy was set to 2.5 m mainly due to the capabilities of the onboard GPS using the external antenna and the angle accuracy was set to $\pm 15 deg$. The positioning was chosen to be performed only over the sway and yaw degrees of freedom and surge was neglected due to its underactuation that would require maneuvering the vessel which goes out of the scope of the present Thesis.

In addition, as the system exists physically as well, delays in actuation were incorporated to the study by calling the controller once every second. The memory needed for the network in the pre-training was chosen a lot bigger than would be in the subsequent online training using the RPi to fully utilise the capabilities of the larger computational system to give the network as good of a tuning as possible. The network architecture follows in general that of other agents in a multitude of control system environments already existing open-source such as the inverted pendulum environment.

As no similar graphics existed beforehand, learning the use of OpenGL was required to correctly display the actions performed as well as info about the disturbance's wave height and angle of attack, the current sway position and heading angle at each episode and the thrust allocation scheme. Despite using the Nomoto equations to translate the total force and moment from both the actuators and the disturbances upon the vessel to the y-position and heading at each episode, in sea trials this would be substituted by actual sensor measurements. These sensors all have their own respective errors which need to be fixed and using sensor fusion techniques like

Kalman filtering or complementary filters, the values could be stabilized and get closer to the truth.

Finally, the reinforcement learning DQN Architecture allows for the further development of the system by adding for example the surge actuation or by including more sensors such as a dedicated anemometer which would initially position the vessel in a favourable angle to maximize the time of seakeeping. Changes to the algorithm such as a more sophisticated reward system could be beneficial. As for the final purposes of the whole project, dynamic positioning could be expanded on track keeping for both berthing and maneuvering applications.

8.2 Future Work

Even if the LINCOLN Project has officially ended by the time of submition of this Thesis, the work done could be further pushed to explore the boundaries of Ship Autonomous Motion Control. As aforementioned the DQN Architecture could be expanded by introducing other goals and/or hardware additions to the system. Some of them, for instance, would be:

- Introducing more on-board sensors such as an anemometer to enhance system monitoring
- Using cameras and image processing instead of terminal switches to accurately position the valves even in intermediate positions, thus transforming the action space from discrete to continuous
- Coming up with a more sophisticated reward system or use an optimizer to minimize for example the controller effort or the position errors
- Expanding to track keeping which would also allow for surge actuation even without incorporating more thrusters
- Performing a more accurate sensor fusion to reduce the measurement errors

It is the firm belief of the author that this project could be the start of a multitude of enjoyable and timeworthy research papers and/or Theses which would have great impact for both the students and the University Community.

A Wind Moment Algorithm

```
function Nw=wind_dist(heading)
    % Wind yaw moment calculation based on Isherwood's paper
    S=0;
    L=4.1;% ship length
    B=2;% ship beam
    VR=5*0.5144; %air velocity
    C=(1-0.15)/2+0.15;
    AL=3.98; %m^2 Calculated from Fusion 360
    AT=1.34; %m^2
    lookup_table_angles=10:10:170
    %Tables of constants
    C0
        =[0.0596,0.1106,0.2258,0.2017,0.1759,0.1925,0.2133,0.1827,0.2627,0.2102,

         0.1567,0.0801,0.0189,0.0256,0.0552,0.0881,0.0851];
    C1=[0.061,0.204,0.245,0.457,0.573,0.48,0.315,0.254,zeros(1,size(C0
        ,2)-8)];
    C2=[zeros(1,size(C0,2)-8)
        ,-0.0195,-0.0258,-0.0311,-0.0488,-0.0422,-0.0381,-0.0306,-0.0122];

    C3=[zeros(1,3),0.0067,0.0118,0.0115,0.0081,0.0053,zeros(1,4)
        ,0.0101,0.01,0.0109,0.0091,0.0025];
    C4=[zeros(1,9),0.0335,0.0497,0.074,0.1128,0.0889,0.0689,0.0366,0];
    C5
        =[-0.074,-0.17,-0.38,-0.472,-0.523,-0.546,-0.526,-0.443,-0.508,-0.492,

         -0.457,-0.396,-0.42,-0.463,-0.476,-0.415,-0.22];
    SE
        =[0.0048,0.0074,0.0105,0.0137,0.0149,0.0133,0.0125,0.0123,0.0141,0.0146,

         0.0163,0.0179,0.0166,0.0162,0.0141,0.0101,0.0057];
    CN=C0+C1*(2*AL/L^2)+C2*(2*AT/B^2)+C3*L/B+C4*S/L+C5*C/L+1.96*SE;
    Nws=CN*1.225*AL*L*VR^2/2;
    Nw = interp1q(lookup_table_angles,Nws,abs(heading))
    if (heading<0)
        Nw=-Nw
    end
```

Bibliography

[1] Alexander Amini. “Introduction to Deep Learning”. 6.SI91 Introduction to Deep Learning. 2020.

[2] M.A Unar Arbab Nighat Khizer Dai Yapping. “Design of Heading Controller for Cargo Ship using Feed Forward Artificial Neural Network”. In: *International Journal of Advancements in Computing Technology* (2013).

[3] A. Tiano A.Zirilli G.N Roberts and R.Sutton. “Adaptive Steering of a containership based on neural networks”. In: *International Journal of Adaptive Control and Signal Processing* 14 (2000), pp. 849–873.

[4] Mao Xiang Gu Ding Li. “Dynamic Positioning of Ships using a planned neural network controller”. In: *Journal of Ship Research* 40 (1996), pp. 164–171.

[5] Stanley W. Doroff. *High-Perfomance Ships*. Office of Naval Research-Department of the Navy, Washington D.C, 1960.

[6] Thor I. Fossen. *Guidance and Control of Ocean Vehicles*. John Wiley and Sons, 1999.

[7] Thor I. Fossen. *Handbook of marine craft hydrodynamics and motion control*. John Wiley and Sons, 2011.

[8] Thor I. Fossen. *Marine Control Systems;Guidance, Navigation, and Control of Ships, Rigs and Underwater Vehicles*. Marine Cybernetics, Trondheim, Norway, 2002.

[9] K.A Belibasakis G.A Athanassoulis. *Ship Dynamics*. School of Naval Architecture and Marine Engineering, National Technical University of Athens, 2012.

[10] Hado van Hasselt, Arthur Guez, and David Silver. “Deep Reinforcement Learning with Double Q-Learning”. In: *AAAI*. 2016.

[11] B.E. Kiun Maung OO. *The Design of Semi-Submersibles for minimum vertical motion*. PhD thesis, University of Glasgow, 1974.

[12] Manon Kok, Jeroen D. Hol, and Thomas B. Schön. “Using Inertial Sensors for Position and Orientation Estimation”. In: *CoRR* abs/1704.06053 (2017). arXiv: 1704.06053. URL: http://arxiv.org/abs/1704.06053.

[13] Nikolaos Kouretas. “Development of a Dynamic Positioning System for an Emergency Recovery and Rescue Vessel”. In: *Diploma Thesis, School of Naval Architecture and Marine Engineering, National Technical University of Athens* (2019).

[14] Shrikant Rao K.S. Hemanth Viswanath Talasila. “Calibration of 3-axis Magnetometers”. In: *IFAC Proceedings Volumes* 45 (2012), pp. 175–178.

[15] Long ji Lin. “Reinforcement learning for robots using neural networks”. In: 1992.

[16] R.M.Isherwood M.A. “Wind Resistance of Merchant Ships”. In: *RINA Transactions* 115 (1972), pp. 327–338.

[17] D.J. Murray-Smith M.A Unar. “Automatic Steering of Ships using neural networks”. In: *International Journal of Adaptive Control and Signal Processing* 13 (1999), pp. 203–218.

[18] P.P.C Yip Mao Xiang Gu Yoh-Han Pao. “Neural-net computing for real-time control of a ship’s dynamic positioning at sea”. In: *Control Eng.Practice* 1 (1993), pp. 305–314.

[19] Y.S.Kim M.Hamamoto M. Fujino. “Dynamic Stability of a Ship in Quartering Seas”. In: *Prepared for Stability* (1994).

[20] J.N Newman. “Second Order Slowly Varying Forces on Vessels in Irregular Waves”. In: *Proc. Int. Symp. on the Dynamics of Marine Vehicles and Structures in Waves, London, England* (1974), pp. 182–186.

[21] M.K Ochi. “Turbulent winds anf forces for consideration of stability of marine systems”. In: *Proccedings,4th International Conference on Stability of Ships and Ocean Vehicles, Naples* (1990).

[22] OpenAI. *OpenAI Spinning Up*. URL: https://spinningup.openai.com/.

[23] Holland Marine Parts. *JT-30 Perfect Bow Thruster for Wakeboarding Boats*. URL: https://www.hollandmarineparts.nl/jetthruster/jt-30.

[24] Johannes Pinkster. *Dynamic Positioning of vessels at sea, Course Notes*. International centre for mechanical sciences, Udine, Netherlands.

[25] William Gerraint Price. *Probabilistic theory of ship dynamics*. Chapman and Hall, 1974.

[26] Dr.Eng Professor M. Yamagata. “STANDARD OF STABILITY ADOPTED IN JAPAN”. In: *RINA Transactions* 101 (1959), pp. 417–443.

[27] Sebastian Raschka and Vahid Mirjalili. *Python Machine Learning; Machine Learning and Deep Learning with Python, scikit-learn and TensorFlow*. Packt Publishing, Birmingham-Mumbai, 2017.

[28] Tom Schaul et al. *Prioritized Experience Replay*. 2015. arXiv: 1511.05952 [cs.LG].

[29] Asgeir J. Sorensen. *Marine Control Systems;Propulsion and Motion Control of Ships and Ocean Structures, Lecture Notes*. 2013.

[30] K.J. Spyrou. *Ship Design for Maneuverability and Course Stability*. School of Naval Architecture and Marine Engineering, National Technical University of Athens, 2010.

[31] Grant E. Hearn Yao Zhang and Pratyush Sen. “A Multivariable Neural Controller for Automatic Ship Berthing”. In: *IEEE Control Systems Magazine, Vol.17 Issue 4 , pp.31-45) Year=1997* ().

[32] Grant E. Hearn Yao Zhang and Pratyush Sen. “A Neural Network Approach to Ship Track-Keeping Control”. In: *IEEE Journal of Oceanic Engineering, No.4* 21 (1996)).

[33] Xin Hu Haiquan Chen Yongyi Lin Jialu Du. “Design of Neural Network Observer for Ship Dynamic Positioning System”. In: *Proceedings of the 33rd Chinese Control Conference* (2014).

[34] Zhengquan Zhou Yusong Cao and William S. Vorus. “Application of a Neural Network Predictor/Controller to Dynamic Positioning of Offshore Structures”. In: *Dynamic Positioning Conference; Advances in Technology* (2000).