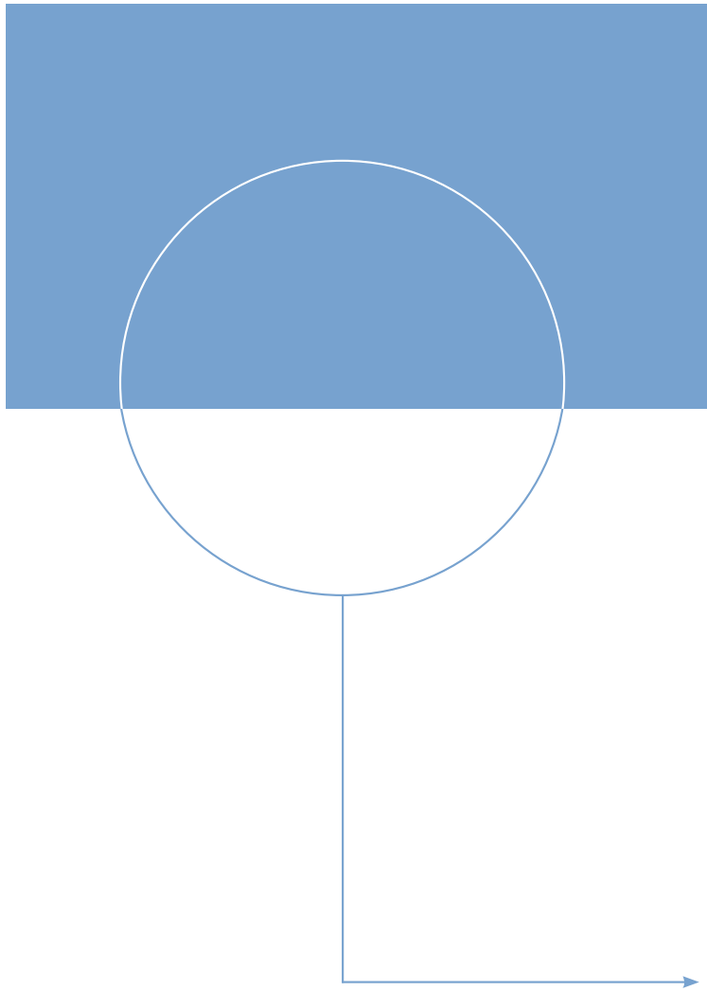


Master's thesis

2020

Jens Nikolai Alfsen

**NTNU**
Norwegian University of
Science and Technology
Faculty of Engineering
Department of Marine Technology

Master's thesis

Jens Nikolai Alfsen

# IMU-based sea state estimation using convolutional neural networks for DP vessels

June 2020

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

# IMU-based sea state estimation using convolutional neural networks for DP vessels

Jens Nikolai Alfsen

Engineering and ICT
Submission date: June 2020
Supervisor: Roger Skjetne
Co-supervisor: Zhengru Ren and Håvard Snefjellå Løvås

Norwegian University of Science and Technology
Department of Marine Technology

**NTNU Trondheim**
**Norwegian University of Science and Technology**
*Department of Marine Technology*

# MASTER OF TECHNOLOGY THESIS DEFINITION (30 SP)

**Name of the candidate:** Jens Nikolai Alfsen

**Field of study:** Marine control engineering

**Thesis title (Norwegian):** IMU-basert sjøtilstandsestimering for DP skip ved bruk av nevrale nettverk

**Thesis title (English):** IMU-based sea state estimation for DP vessels using convolutional neural networks

**Background**

The design of autonomous surface vessels is increasing in speed, and both the academia and maritime industry are doing research on the topic. One important aspect of an autonomous marine vehicle is the situational awareness of the vehicle by increased perception of environmental loads such as wind, waves, and current. Autonomous systems require novel sensor systems that replace or even exceed the capabilities of a human operator, which is of high importance regarding safety and performance in autonomous operations.

There are several approaches that can be used to increase the awareness of the loading situation for the vehicle. A bio-inspired solution is how a fish sense flow and navigate in a current. The fish has a lateral-line sensing organ, which enables it to detect changes in pressure and thus swim graciously in the flow of ocean or river currents. For a marine vehicle, the understanding of surrounding environment can be done through different types of sensors. An array of inertial measurement units mounted along the vessel hull provides detailed information about the forces acting on the vessel. These will measure local accelerations which, if the vessel is rigid, can be fused to a global acceleration vector. The highly non-linear mapping between the wave excitation and ship accelerations, can be modelled based on extensive knowledge of the ship dynamics; however, this is time- and computer demanding. Machine learning models can possibly be able to approximate this complex mapping; if so, they may drastically decrease the time delay.

**Work description**

1. Perform a background and literature review to provide information and relevant references on:
    a. Autonomy in the maritime sector.
    b. Dynamic positioning.
    c. Sea state estimation.
    d. Relevant machine learning methods, especially applications for sea state estimation.

    Write a list with abbreviations and definitions of terms and symbols, relevant to the literature study and project report.

2. Establish a simulation model for a vessel in DP operation, with the ability to set realistic sea states. The simulation model should include realistic IMU measurements from as many IMUs as desired, each located at a specified location in body-fixed frame.

3. Develop relevant preprocessing techniques to create a dataset ready for model training, incl.:
    a. Sensor fusion
    b. One-hot encoding

4. Develop machine learning models for regression/classification of wave height, wave frequency, and wave direction based on the processed IMU data, to be applied to both *regular* and *irregular* long crested sea.

5. ***Tentative (if lab work/data becomes feasible):*** Install IMUs on CSAD in MC-Lab and test the models in the lab (or on other data generated in lab).

**Specifications**

Every weekend throughout the project period, the candidate shall send a status email to the supervisor and co-advisors, providing two brief bulleted lists: 1) work done recent week, and 2) work planned to be done next week.

The scope of work may prove to be larger than initially anticipated. By the approval from the supervisor, described topics may be deleted or reduced in extent without consequences with regard to grading.

The candidate shall present personal contribution to the resolution of problems within the scope of work. Theories and conclusions should be based on mathematical derivations and logic reasoning identifying the various steps in the deduction.

The report shall be organized in a logical structure to give a clear exposition of background, problem, design, results, and critical analysis. The text should be brief and to the point, with a clear language. Rigorous mathematical deductions and illustrating figures are preferred over lengthy textual descriptions. The report shall have font size 11 pts., and it is not expected to be longer than 70 A4-pages, 100 B5-pages, from introduction to conclusion, unless otherwise agreed upon. It shall be written in English (preferably US) and contain the elements: Title page, abstract, preface (incl. description of help, resources, and internal and external factors that have affected the project process), acknowledgement, project definition, list of symbols and acronyms, table of contents, introduction (project background/motivation, objectives, scope and delimitations, and contributions), technical background and literature review, problem formulation, method, results and analysis, conclusions with recommendations for further work, references, and optional appendices. Figures, tables, and equations shall be numerated. The original contribution of the candidate and material taken from other sources shall be clearly identified. Work from other sources shall be properly acknowledged using quotations and a Harvard citation style (e.g. natbib Latex package). The work is expected to be conducted in an honest and ethical manner, without any sort of plagiarism and misconduct, which is taken very seriously by the university and will result in consequences. NTNU can use the results freely in research and teaching by proper referencing, unless otherwise agreed upon.

The thesis shall be submitted with an electronic copy to the main supervisor and department according to NTNU administrative procedures. The final revised version of this thesis definition shall be included after the title page. Computer code, pictures, videos, dataseries, etc., shall be included electronically with the report.

**Start date:** 15 January, 2020 **Due date:** 30. June 2020

**Supervisor:** Roger Skjetne
**Co-advisor(s):** Zhengru Ren, Håvard Snefjellå Løvås

**Trondheim,** June 15th, 2020

Roger Skjetne

Digitally signed by Roger Skjetne
Date: 2020.06.15 13:51:30 +02'00'

**Roger Skjetne**
Supervisor

## Abstract

The thesis explore feasibility of sea state estimation based on measured vessel motion. Spatially distributed IMU sensors installed on a DP vessel enables the accelerations of the vessel to be estimated, which is further used to train the sea state estimation models.

A 6 DOF simulation model for the 1:90 model vessel C/S Arctic Drillship is implemented in Simulink. This includes modeling of long-crested regular and irregular waves and the corresponding vessel response. Realistic measurements are modeled from four spatially distributed IMUs, and a sensor fusion algorithm is applied for estimation of the vessel motion in the center of control. A motion control system is also modeled, including a guidance module, a model-based DP controller, and a model-based nonlinear passive observer. The motion control system is designed based on a simplified 3 DOF control design model, only acting in the horizontal plane.

The sea state estimation model consists of three convolutional neural networks implemented in parallel. Two are built as regression models for estimating specific wave height and peak wave period. The last is a classification model for estimating relative wave direction defined in twelve sectors of 30°. The labeled acceleration data in heave, roll, and pitch from the simulation is firstly used to find the general architecture and secondly the optimal hyperparameters for the neural networks.

Four IMUs are installed on the actual vessel and tested in the Marine Cybernetics Laboratory with long-crested regular waves generated in the basin. The measurements are filtered, and used to estimate the 3 DOF accelerations in the vessel center of control.

The models are trained and tested on labeled data from simulation in long-crested irregular waves, producing very satisfactory results. A mean significant wave height error of 3.133%, a mean peak period error of 0.896%, and 100% accuracy for wave direction is achieved. Similarly satisfying results are found when the models are trained on unfiltered data with noise and bias, demonstrating the robustness of the end-to-end models.

The sea state estimation models are then trained on simulation data in long-crested regular waves before they are tested on the experimental data from the lab. The estimation results are good for the wave period model but worse for wave height and direction. This is, however, as expected due to the natural discrepancies between modeled and actual motion. It still shows the potential for using convolutional neural networks for sea state estimation for DP vessels.

## Preface

This thesis concludes my master's degree from Engineering and ICT at the Norwegian University of Science and Technology, with specializations within marine cybernetics and artificial intelligence. The work presented is partly a continuation of my project thesis, which focused on auto-tuning of DP algorithms using derivative-free optimization.

The experimental data presented in this thesis is produced in the Marine Cybernetics Laboratory at NTNU, and the set-up consumed many hours during the first two months. The laboratory, unfortunately, closed in March due to the COVID-19 outbreak. As a result, the test data retrieved from the lab prior to the closing is used, despite not being sufficient for fully evaluating the scope of the sea state estimation models. The motion control system was implemented on the vessel, however the controller was not tuned. The initial data was therefore generated from the vessel being strapped in the basin for position keeping, although the vessel was planned to be in actual DP operation during the experiment.

Much time went into reading up on, and brief testing of different machine learning methods and their ability to interpret time series data. Boosted trees, LSTMs, and hybrid CNN-LSTM are some of the models initially explored.

The thesis process has enriched my knowledge and experience with hardware and real-world sensor data, which I believe is valuable hands-on experience for future work in the industry. The thesis has also made me expand my knowledge on topics like marine hydrodynamics, control theory, and deep neural networks.

## Acknowledgement

I would like to thank my supervisor Professor Roger Skjetne for giving me the idea of the master thesis, combining my two specializations marine cybernetics and artificial intelligence. I was welcome to his office for several helpful discussions even on short notice. He also provided me with my two co-advisors Zhengru Ren and Håvard Snefjellå Løvås. Zhengru invested a lot of time helping me with the direction of the thesis, as well as proofreading. Håvard has also helped me a great deal through several academic discussions. In the laboratory, I received crucial assistance dealing with the IMUs and other hardware challenges from Torgeir Wahl, who answered my many calls and messages at all hours of the day. Lastly, I would like to extend my gratitude and appreciation to my friend Ola Scheele Moe for his serious knowledge on the subject of autonomy, resulting in many helpful discussions.

## Abbreviations

- ANFIS: Adaptive neuro-fuzzy inference system
- ANN: Artificial neural network
- CDM: Control design model
- CG: Center of gravity
- CLF: Control Lyapunov function
- CNN: Convolutional neural network
- CO: Center of control
- CSAD: C/S Arctic Drillship
- DNN: Deep neural networks
- DOF: Degrees of freedom
- DP: Dynamic positioning
- DR: Dead reckoning
- DSV: Diving support vessel
- ECEF: Earth-centered earth-fixed reference frame
- ECI: Earth-centered inertial reference frame
- EMSA: European Maritime Safety Agency
- FFNN: Feed-forward neural network
- FFT: Fast Fourier transform
- FRF: Frequency response functions
- GES: Globally exponentially stable
- GNSS: Global navigation satellite system
- GPS: Global positioning system
- GRU: Gated recurrent unit
- HHT: Hilbert-Huang transform
- HIL: Human in-the-loop
- IMU: Inertial measurement unit
- ITTC: International towing tank conference

- LSTM: Long short-term memory
- MIMO: Multiple-input multiple-output
- MSS: Marine systems simulator
- NCA: Norwegian coastal authority
- NED: North-east-down reference frame
- NLP: Natural language processing
- NN: Neural network
- OSV: Offshore support vessel
- PLSR: Partial least squares regression
- PSO: Particle swarm optimization
- PSV: Platform supply vessel
- QDA: Quadratic discriminant analysis
- QTM: Qualisys track manager
- RAO: Response amplitude operator
- ReLU: Rectified linear unit
- RF: Random forest
- RNN: Recurrent neural network
- ROV: Remotely operated vehicle
- RPM: Rotations per minute
- SGD: Stochastic gradient descent
- SISO: Single-input single-output
- SVM: Simulation verification model

## Nomenclature

- $\eta$: NED frame position
- $\psi$: Heading angle of vessel x-axis relative to North
- $\chi$: Course angle of vessel velocity vector relative to North
- $\beta$: Crab angle of vessel velocity vector relative to body x-axis

- $\nu$: velocity vector in body reference frame
- $\tau$: forces and moments in body reference frame
- $\alpha$: thruster orientation
- $T_p$: Peak wave period
- $H_s$: Significant wave height

# Table of Contents

# List of Figures

## List of Tables

# 1 Introduction

## 1.1 Motivation

The desire for reducing operational expenses, fuel consumption, and emissions in conjunction with increasing reliability, efficiency, and safety has incentivized the development of autonomous vessels. An extensive area of The Trondheims Fjord has been designated for testing by the Norwegian Coastal Authority (NCA). Yara Birkeland, the world's first commercially available autonomous ship, will be delivered in the fall of 2020 (Stensvold). The two milestones represent huge progress towards autonomy at sea.

Perhaps the most challenging difference between autonomous cars and vessels is the dynamic environment in which a vessel operates. Accurate estimation of sea state- the height, frequency, and direction of the waves - is crucial and fundamental for the safety of an autonomous vessel. It can also increase the performance of vessels with a lower level of autonomy, such as dynamic positioning (DP) vessels. Optimal tuning of the control system of a DP vessel is dependent on the sea state, which is typically assumed to be stationary for 20-minute intervals. This leads to a desire for at least equally frequent updates of the tuning parameters. The precision of the sea state estimate is effectively working as an upper limit for the performance of all higher-level algorithms and systems - they are only as good as the underlying estimate.

The vessel motion is partly influenced by the waves, implying that information about the sea state is contained in measured vessel motion. The importance of accurate, real-time knowledge of the environment in which the vessel operate motivates research into the possibility of sea state estimation solely based on vessel motion measurements.

## 1.2 Objectives

The vessel motion can be measured using an IMU, containing an accelerometer and a gyroscope. For generation of realistic IMU measurements, the first objective is to design a simulation model for the C/S Arctic Drillship (CSAD), described in section 4. The control algorithms for the dynamic positioning system is further presented in section 5.

To create data-sets ready for model training, the relevant pre-processing is presented in section 7. This section also includes the development and tuning of the sea state estimation models based on the data-sets from simulation.

Section 8 presents the instrumentation of the IMUs and the experimental set-up in lab. Finally the sea state estimation results are presented in section 9.

## 1.3 Scope and delimitations

The thesis is focused on the C/S Arctic Drillship, a 1:90 model vessel. This vessel is chosen as it is the largest vessel in the MC-lab fleet, enabling the installation of four spatially distributed IMUs. The sea state estimation models are trained on simulated motion data from the vessel and are therefore only applicable for sea state estimation for CSAD.

Current and wind are neglected for simplicity, resulting in vessel motion induced only by actuators and waves. The data-sets used to train the models are also generated under the assumption of perfect measurements.

While the simulated data thoroughly represent all sea states from moderate to high sea, the experimental data only represent a few different sea states. The reason is the unexpected closing of the lab in March and the inability to conduct further experiments.

## 1.4 Contributions

The main contributions of the thesis are

- A robust, efficient sea state estimation model solely dependent on measured vessel motion. The model can handle noisy measurements, and performs accurate estimation based on only 40 seconds of measured vessel motion.
- A 6 DOF simulation model for the C/S Arctic Drillship with dynamic positioning system, ability to set realistic sea states, and modelling of related IMU measurements.
- DP system for CSAD implemented in Veristand with configuration for logging of IMU measurements.

# 2 Technical background

## 2.1 Autonomous marine operations

There are several levels of autonomy, with each level having increased system independency and hence decreasing level of human interaction. According to NIST (2016) and Ludvigsen and Sørensen (2016), the levels can be categorized as follows

- *Automatic operation (remote control) means that even though the system operates automatically. The human operator directs and controls all high-level mission planning functions, often preprogrammed (human-in-the-loop/human operated).*
- *Management by consent (teleoperation) means that the system automatically makes recommendations for mission actions related to specific functions, and the system prompts the human operator at important points in time for information or decisions. At this level the system may have limited communication bandwidth including time delay, due to i.e. distance. The system can perform many functions independently of human control when delegated to do so (human-delegated).*
- *Semi-autonomous or management by exception means that the system automatically executes mission-related functions when response times are too short for human intervention. The human may override or change parameters and cancel or redirect actions within defined time lines. The operators attention is only brought to exceptions for certain decisions (human-supervisory control).*
- *Highly autonomous, which means that the system automatically executes mission- related functions in an unstructured environment with ability to plan and re-plan the mission. The human may be informed about the progress. The system is independent and "intelligent" (human-out-of-the loop).*

Ludvigsen and Sørensen (2016) also proposes a "bottom-up" architecture for autonomous vessels, as an attempt to explain the critical parts of an autonomous system

- *Mission planner level: The mission objective is defined and the mission is planned. Subject to contingency handling, any input from payload sensor data analysis and any other input from the autonomy layer, the mission may be re-planned.*
- *Guidance and optimization level: Handles waypoints and references commands to the controller.*
- *Control execution level: The plant control and actuator control takes place.*

A fully autonomous vessel will be able to optimize the mission planning, by continuously analysing mission-critical data. Being unmanned enables lower sailing speed since it no longer implies a longer crew shift. Mission planning can be more cargo dependent, not limited by human crew

consideration. This leads to a reduction in operational cost, fuel consumption, and emissions. An increased operational window will also come as a result of the system handling more extreme weather. On the guidance and optimization level, an autonomous vessel will better optimize the waypoints with regards to the measured environmental loads, which further reduce emissions and fuel consumption. The advanced systems for collision avoidance already in place today will also eliminate the human-error caused accidents. These accidents accounted for 65.8% of all accidents at sea in 2019, according to the European Maritime Safety Agency, EMSA (2019). Control execution level autonomy will enable real-time tuning of control algorithms, depending on updated knowledge of the environment - further increasing efficiency and safety.

There are nevertheless many challenges associated with autonomous vessels, including cyber-attacks and the moral questions regarding hardcoded decision-making in life-threatening situations. Another challenge is the dynamic and unpredictable environment it operates, at least in comparison to autonomous cars. Waves strongly influence the performance of such a vessel, demanding highly advanced systems for sea state estimation.

## 2.2 Dynamic positioning

Dynamic positioning (DP) control system enables a vessel to automatically maintain position and heading, or low speed tracking. The system is based on a mathematical model describing the relationship between forces acting on the vessel and its motion. From the GNSS and often supplementary sensors, the system calculates the summed environmental forces affecting the vessel position. The necessary low-level actuator set-points to counteract the environmental forces can be calculated. This also enables the system to keep operating for some time in case of loss of positional signal, called dead reckoning.

According to Sørensen et al. (1996), the first DP system became commercially available in the 1960s. This system was a simple analog PID controller which did not estimate environmental forces or model errors, and was therefore useless in most weather situations. Kalman filtering, advanced digital data transmission, and big improvements within control theory have led to the advanced, robust systems in place today. In 2013 there were 1800 registered DP vessels in the world, according to The Nautical Institute (2013). The most common are Platform supply vessels (PSV), Offshore Support-(OSV), Diving Support- (DSV), ROV Support-, Drill-, and Pipe Laying Vessels.

IMCA (2010) lists the alternatives to dynamic positioning as mainly jack-up barge and anchoring. The advantages for these are that they are not dependent on complex thruster systems with controllers, and blackouts or system failures are not critical with regards to the position keeping. The disadvantages are limited or no maneuverability once the position is set, and dependency on relatively shallow water. Additional disadvantages for anchoring is limitation by obstructed seabed and the dependency on anchor handling tugs. Advantages of DP are thus the maneuverability, water depth independent, unrestricted by obstructed seabed, and quick set up. There is, however,

a high initial cost for the system.

Dynamic positioning is under the control execution level of vessel autonomy, see section 2.1. For the algorithms in the DP control system to function optimally, extensive parameter tuning is required. Today this is done manually by an operator with sea trials. The parameters are dependent on the vessel design but also on the sea state. The challenge in practice becomes the underlying dependency on real-time, accurate estimation of the sea state.

## 2.3 Sea state estimation

Weather forecast and statistics can contribute with much information for a vessel in operation, however, it does not contain the local precision needed for optimal functioning of a DP control system. Regardless of the vessel's level of autonomy, a high-precision sea state estimate is useful, even if only for decision support systems with an operator present. Online sea state estimates can be used directly for manipulating parameters in the control algorithms, or as a trigger for switching between several pre-tuned control laws. The sea state is considered the combination of three parameters; the mean wave height of the one third highest waves ($H_s$), the period of the most energetic waves ($T_p$), and the relative wave direction ($\beta$). Many efficient methods for estimating the wave period exist, but dependable algorithms for the two others are less common.

Wave buoys have been frequently used for sea state estimation, but are limited to a fixed location of operation. Wave radar is an alternative according to Stredulinsky and Thornhill (2011), Clauss et al. (2009a), and Clauss et al. (2009b), however the precision quickly decreases with large vessel movements. The current most promising method is using the vessel itself as a wave buoy, often called *wave buoy analogy*. The theoretical foundation of the *wave buoy analogy* is the existence of a mathematical relationship between the motion of the vessel and the current sea state. Many approaches to identify this relation, enabling mapping of measured vessel motion to a sea state, exist. They are typically divided into model-based and data-driven methods, and further into parametric and non-parametric.

The model-based approaches combine wave-induced measurement-spectrum on board with a mathematical model of the vessel and wave spectrum. These approaches are mostly based on the frequency domain, where the relationship between the vessel cross response spectra and the directional wave spectrum are calculated offline. The calculations are done in hydrodynamic simulation software using the vessel hull geometry, and the mathematically expressed relation is called the response amplitude operator (RAO). The RAO hence express the vessel motion in response to first order waves given the wave parameters. The relation is further utilized to map the motion experienced by the sensors onboard to a specific wave spectrum. A parametric approach means that a certain form of the wave spectra is initially assumed, while the opposite is true for non-parametric approaches. The drawback of model-based approaches is the dependency on accurate vessel models which are expensive, and impossible for more complex sea.

Data-driven approaches, on the other hand, are not dependent on prior knowledge of the vessel and are therefore more easily generalized. These approaches apply machine learning techniques to identify temporal and frequency features. Non-parametric models include, for example, k-nearest neighbors where the similarity in measurements indicate similar output. The model does not assume a form of the underlying function it is trying to approximate. Neural networks are examples of parametric models, where the relationship between sea state and measurements are approximated by optimizing model parameters to produce the desired output. The number of parameters is set, and hence the assumption of the form/complexity of the underlying mapping function.

One of the initial studies on the model-based *wave buoy analogy* was Takekuma and Takahashi (1972). The response amplitude characteristics of the vessel were used to calculate the power spectral density of the waves, under assumption of linear superposition. This was, however, limited to stationary vessels and only based on pitch motion. The method was taken further by Hirayama and Hagino (1985), where accelerometers and vertical gyros were utilized also to consider forward speed of the vessel. The wave direction was obtained by using the ship's radar. The problem caused by the Doppler effect in following sea was not approached until Iseki and Ohtsu (2000).

A number of studies have been published in later years focusing on model-based sea state estimation for DP applications. Load variations, operational trim, and other factors change the dynamic behavior of a vessel and are often not considered in the RAO. Tannuri et al. (2003) focuses on the errors in the estimated wave spectrum induced by these unmodelled changes. In Brodtkorb et al. (2018b) a non-parametric approach is tested for directional wave spectrum estimation. The RAO is simplified into a set of closed-form expressions, which introduces a computationally efficient algorithm that operates under limited knowledge of vessel hull geometry. This method was further extended to applications with forward speed and for short-crested sea states in Nielsen et al. (2018). Both approaches use pre-sampled data from simulation or full-scale sea trials and are hence per definition offline estimation. In Brodtkorb et al. (2018a) the previously mentioned approaches are implemented for online sea state estimation on a DP vessel.

A combination of parametric and non-parametric models has been attempted to utilize the advantages of both models. In Pascoal and Guedes Soares (2008), a non-parametric formulation is applied initially to get an a-priori estimation of swell and wind sea spectral properties, a parametric fitting procedure is further applied based on the estimated properties. It is an improvement of Pascoal et al. (2008), where the minimization procedure is refined and it presents a scheme of estimating spectral parameters by fitting a parametric form.

Nielsen (2007) implements two different methods for directional wave spectrum estimation, namely Bayesian modeling and parametric modeling. Both methods are based on complex-valued frequency response functions (FRF) and are therefore favorable for evaluating how the filtering aspect influences the final estimation.

The aforementioned model-based approaches are formulated in the frequency domain. Depending

on the accuracy of the RAOs, and hence on the spectral analysis like Fast Fourier Transform (FFT). The vessel's ability to remain stationary highly influences the results due to the time needed to perform the analysis, limiting these approaches. In Nielsen et al. (2015) a hybrid model consisting of a model-based approach and a data-driven approach is implemented. The estimation of wave frequency is done in time domain solely based on the measured vessel motion, hence the stationarity dependency is removed. The wave amplitude and phase are estimated using a model-based approach.

Due to the limitations of approaches formulated in the frequency domain, some research has been done on sea state estimation directly in time domain. Pascoal and Guedes Soares (2009) proposes a Kalman filter-based method with waves in-phase and quadrature components as states. Wave height and direction can then be accurately estimated. Nielsen et al. (2016) is another example of the wave buoy analogy formulated directly in time domain, only partly dependent on RAO. The wave frequency estimation is independent of the RAO, while wave amplitude and phase are estimated using nonlinear least squares fitting. The paper is a continuation of the previously mentioned theoretical paper Nielsen et al. (2015), now applied to model-scale experiments.

Roll, pitch, and heave are movements that are expensive to control, considering the benefit to the objective of dynamic positioning. The DP system is hence limited to controlling motion in the horizontal plane. As a result, the motion in the initially mentioned degrees of freedom is predominantly induced by the waves, and therefore, the most suitable for sea state estimation.

## 2.4 Machine learning applications

The previously mentioned methods are based on knowledge of some version of the vessel transfer function and are hence model-based. The transfer function of a vessel can be complex to calculate, and if nonlinear effects are not correctly accounted for, larger errors will be introduced in more extreme sea states. Transforming the time series samples into frequency domain loses information about the phase differences between the signals, which is crucial information for sea state estimation. Data-driven approaches have therefore become increasingly popular, and the current available computation power has given the earlier disregarded methodologies of neural networks and other ML techniques its renaissance.

An artificial neural network is a function approximator that needs training on labeled data. It consists of several layers of nodes, where each node in a layer takes the weighted sum of outputs from the previous layer as input. An activation function in each node then decides the output, based on the mentioned input. A key element to the activation function is its non-negative derivative - an increased input imply the same or increased output. The number of layers between the input and output layer varies depending on the complexity of the underlying target function, but networks with more than one *hidden-layer* is often referred to as deep neural networks. The network is trained, and hence *fitted*, to the target function by a scheme called backpropagation. The error between the network's estimation and the target is backpropagated through the network

by adjusting each node's weight in the direction of decreased error - gradient descent. The two most popular architectures of deep neural networks (DNN) used in time series analysis are convolutional and recurrent.

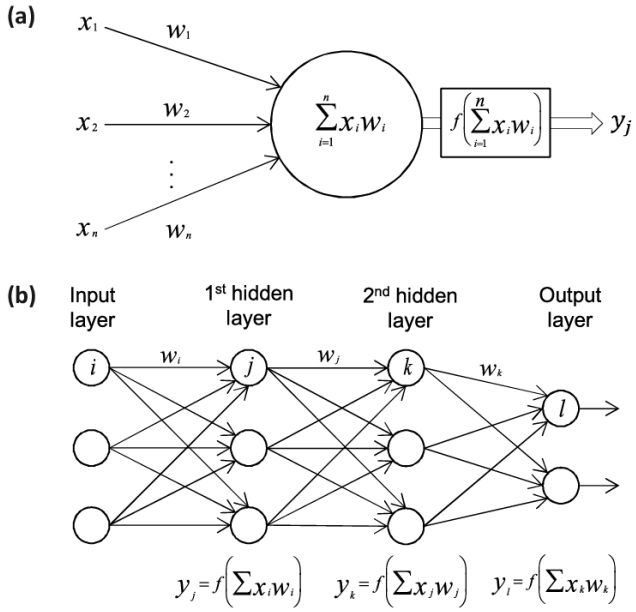



**Figure 1:** Layout of one type of deep neural network (Vieira et al. (2017))

Instead of only densely connected hidden layers, a convolutional neural network typically consists of several convolutional layers, pooling layers, fully connected layers, and normalization layers. The convolutional layers work as a sliding dot product, applying several filters over the time series. This method is based on multiple sets of shared weights, learning to respond to patterns in the time series while maintaining the spatial or temporal relationship between the patterns. The pooling layers further reduce the dimensions of the data by extracting only a single statistic from a predetermined number of neighbouring elements. It works as a sample-based discretization process, where the most common types are max- and min pooling - extracting the maximum or minimum value from a selected region. Several such convolutional stages can be implemented in series, where each stage hierarchically combines the local features into more global features. Lastly, fully connected layers are applied to interpret the encoding. The convolutional neural network architecture is therefore very good for recognizing patterns in image or time series data, indiscriminate to spatial or temporal location of the pattern.

The recurrent neural network (RNN) architecture facilitates input with no limit on size. Through the use of loops it has *hidden states* that work as memory, enabling the input to be interpreted in light of the previous values of the signal. RNNs are hence often used for sequential data such as natural language processing (NLP), where a word or sentence can be ambiguous without knowledge of the context. A problem with the most basic RNNs, often called vanilla RNNs, is the vanishing gradient problem. The backpropagation of the error signal becomes vanishingly small over a long

term temporal interval. The solution was introduced by Hochreiter and Schmidhuber (1997) called Long Short Term Memory (LSTM). The main improvement is introduced through using *weighted gates* to decide what information to keep in each *cell*. This way, dependencies in the signal over long temporal or spatial distance can be remembered. Another simpler version of the LSTM with fewer gates is the gated recurrent unit (GRU), Cho et al. (2014), which has also grown popular in recent years.

When performing dead reckoning (DR) navigation for DP vessels, it often involves integrating several IMU measurements before applying for example Bayesian filtering. The challenge is the delay induced by such methods, as the filtering needs measurements over several waves. Diamant and Jin (2014) uses an Expectation-Maximization algorithm to map short time period acceleration measurements to different pitch states. The acceleration measurements are then combined within each state and integrated to get the pitch compensated distance estimate.

In Ferrandis et al. (2019), machine learning is applied for approximating the complex nonlinear mapping between the stochastic wave elevation and vessel motion in heave, roll, and pitch. Different recurrent neural network models, namely standard, GRU, and LSTM, are trained on expensive CFD simulations offline. The result is a model-independent, more computationally efficient simulation model for 3 DOF vessel motion in extreme sea states.

In Arneson et al. (2019), data-driven sea state estimation is also approached with machine learning. The raw vessel motion data is initially transformed to frequency domain before the frequency domain response is integrated over the frequency range. The result is training data with a single response value for each DOF. Quadratic Discriminant Analysis (QDA) is further applied to the training data for classification of wave direction, as the primary differentiating factor for vessel motion is the wave direction. Partial Least Squares Regression (PLSR), a multivariate regression method, is lastly used to estimate $H_s$ and $T_p$ based on the wave direction.

Mak and Düz (2019) intends to estimate relative wave direction based on 6 DOF vessel motion. The data was produced by a frigate installed with wave radar over two years. Three different deep neural networks are trained on the data, and the results compared. The three networks are CNN for regression, a multivariate LSTM-CNN, and a Sliding Puzzle network. For each network, a hyperparameter tuning study is carried out to optimize the performance and efficiency, and hence limiting the influence of hyperparameter choice. The results show feasibility, however, a lack of data for all sea states is emphasized.

One of the latest contributions to rapid online sea state classification for DP application is Tu et al. (2018). The approach utilizes 4 DOF vessel motion, namely surge, sway, roll, and yaw. K-means clustering and filtering are performed, before the processed data is categorized via Hilbert-Huang transform (HHT). Further feature extraction is performed, before a three-layered classification structure with Adaptive Neuro-Fuzzy Inference System (ANFIS), Random Forest (RF) and Particle Swarm Optimization (PSO) based combination classifiers are implemented for the final sea state

estimation.

The dependency on many hand-crafted features could influence the result to a large degree. Cheng et al. (2019) presented an end-to-end deep neural network that is hence not sensitive to human interventioned pre-processing. An LSTM recurrent neural network detects the long dependencies, a CNN extracts the time-invariant features, and finally, an FFT is employed for the extraction of frequency features. All the features are combined and weighted in a feature fusion layer.

A densely connected CNN is proposed in Cheng et al. (2020) extended also to consider wave direction and generalized for ship motion with forward speed. The network consists of stacked CNN blocks with dense connections. Channel attention modules extract the features form each block, while two feature attention mechanisms intend to combine the hierarchical features.

# 3 Problem Formulation

The 6 DOF motion of a DP vessel, namely the CS Arctic Drillship model vessel, is simulated in Simulink based on the seakeeping model with zero-speed potential coefficients Fossen (2011), due to the assumption of stationkeeping and low-speed maneuvering.

The wave-frequency excitation forces $\boldsymbol{\tau}_{wave}$ are calculated using the vessel low-frequency motion and the vessel RAO. The thesis will cover long-crested regular and irregular sea, assuming deep water and hence applicability of the dispersion function for deepwater. A 3 DOF backstepping control law for $\boldsymbol{\tau}$ is designed based on the objective of following a path described by the variable $s$

$$\left.\begin{aligned} \boldsymbol{\eta}(t) &\to \boldsymbol{\eta}_d(s(t)) \\ \dot{s}(t) &\to v_s(t,s) \end{aligned}\right\} \text{as } t \to \infty.$$

The vessel actuators highly influence the vessel motion in the horizontal plane, however, the vessel motion in heave, roll, and pitch is mostly caused by environmental excitation. Assuming negligible wind and current, the motion therefore contains information about the sea state that the vessel is currently operating in.

An inertial measurement unit (IMU) is a device that is able to measure the aforementioned motion using accelerometers and gyroscopes. Fixed to a body, the accelerometer measures the specific force, while the gyroscope measures the angular velocity. In combination with GPS data, the linear and angular accelerations of the body can be estimated.

Accurate estimation of sea state is crucial and fundamental for the safety of an autonomous vessel, but can also increase the performance of vessels with a lower level of autonomy, such as dynamic positioning. The optimal gains in the control system of a DP vessel is dependent on the sea state, which is typically assumed to be stationary for 20-minute intervals - leading to a desire for at least equally frequent updates of the gains. The precision of the sea state estimate is effectively working as an upper limit for the performance of all higher-level algorithms and systems - they are only as good as the underlying estimate.

As seen in chapter 2, several factors limit the model-based sea state estimation approaches. Examples are the accuracy of the transfer function, the assumption of linearity between the wave spectrum and vessel response, and the time delay induced by the optimization problem accompanying the spectral analysis. There are, however, challenges related to data-driven approaches as well. They include but are not limited to DP ship motion data containing not only wave-induced, but also ship actuator induced motion, the high dimensionality of the sensor data, and noisy measurements. The most critical challenge is still gathering enough accurate data from real-world operations, alternatively building a simulation model that accurately captures most of the non-linear and stochastic dynamics of a vessel in waves.

The first objective of the thesis is to build the simulation model similar to real-world dynamics, including IMU measurements. The second objective is to determine the feasibility of data-driven approaches to sea state estimation. More specifically whether a convolutional neural network (CNN) can be trained to estimate the sea state solely based on 3 DOF accelerations from a DP vessel in simulation. Subsequently, four IMUs are to be installed on the actual model vessel, enabling real motion data to be gathered from the vessel in the basin in MC-lab. The third objective is to determine the precision of the CNN model for the experimental data, alternatively determining how generalized the models have to be also to capture the features of this data.

# 4 Modeling

Following are the mathematical expressions used to build the simulation model derived and presented. This includes the relevant reference frames, and how vectors are transformed between them. Also included are the dynamics relating forces and vessel motion, the simplified vessel dynamics used for the DP control system, and how the IMU measurements are modelled. Lastly the modeling of the sea state is presented, and how the waves are translated into forces acting on the vessel. The 6 DOF model parameters for the C/S Arctic Drillship are used, from Bjørnø (2016).

## 4.1 Kinematics

The kinematics describe the geometrical aspects of motion. It is used to describe the pose; position and attitude, of an object or reference frame with regards to another frame. An example is the relation between the earth fixed NED reference frame and the moving body reference frame visualised in 3 DOF in figure 2.

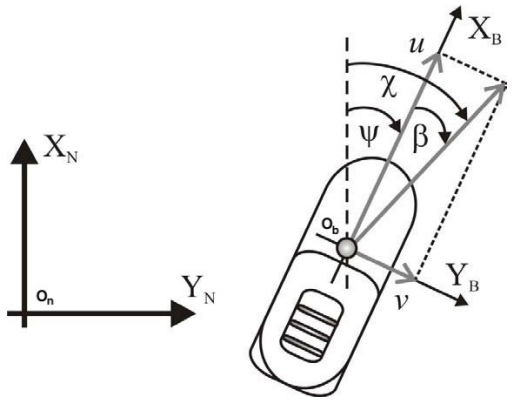



**Figure 2:** Orientation of the moving body fixed reference frame with regards to the fixed NED reference frame (Breivik). Heading angle $\psi$, course angle $\chi$ and crab angle $\beta$ are also indicated.

### 4.1.1 Reference Frames

- Earth-centered inertial (ECI); $\{i\} = (x_i, y_i, z_i)$. Reference frame fixed in space, with origin $o_i$ in the center of the earth. Axis $z_i$ pointed along the earths axis of rotation, with positive direction north. Newton's laws of motion applies to this non-accelerating frame. The IMU measurements are expressed in $\{i\}$.
- Earth-centered Earth-fixed (ECEF); $\{e\} = (x_e, y_e, z_e)$. $z_e$ and origin $o_e$ equal to ECI, however $x_e$ and $y_e$ follows the earths rotation $\omega = 7.2921 \times 10^{-5}$ rad/s. GPS position is given in this reference frame.
- North-East-Down (NED); $\{n\} = (x_n, y_n, z_n)$. Usually defined as the tangent plane, spanned by $x_n$ pointing north and $y_n$ pointing east, on the earth surface. Axis $z_n$ points down, and the reference frame usually moves with the vessel. For local navigation the NED frame can

be assumed inertial and fixed to a point on the surface of the earth. This implies applicability of Newton's laws. The location of $\{n\}$ with regards to $\{e\}$ is uniquely described by longitude and latitude.

- Body; $\{b\} = (x_b, y_b, z_b)$. Fixed to the vessel, usually with $o_b$ at the center of gravity (CG) or at some specified center of control (CO). Axis $x_b$ pointing towards the bow, $y_b$ pointing towards starboard and $z_b$ pointing down normal to the plane spanned by the two others. The vessel velocities in six degrees of freedom are usually expressed in this reference frame, while the pose of the vessel, and hence the frame $\{b\}$, is expressed in $\{n\}$.

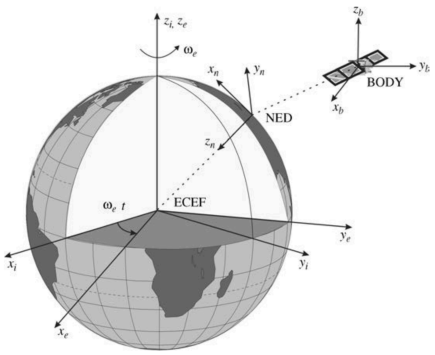



**Figure 3:** Relations between the aforementioned reference frames (Fossen (2011))

- Sensor; $\{s\} = (x_s, y_s, z_s)$. Local reference frame for the inertial measurement units. The rotational velocities are expressed according to the right hand rule, while the linear accelerations are expressed according to the left hand rule (figure 4). The linear acceleration measurements are multiplied with $-1$ henceforth to simplify transformations between reference frames.

- Basin; $\{f\} = (x_f, y_f, z_f)$. The reference frame used in the basin at the MC-lab, where $x_f$ points in the longitudinal direction towards the wave generator, $z_f$ points down and further $y_f$ according to the right hand rule. All positional and orientational Qualisys Oqus measurements are expressed in this reference frame.

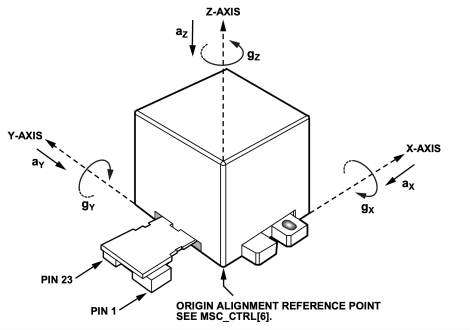



**Figure 4:** IMU sensor reference frame (Devices)

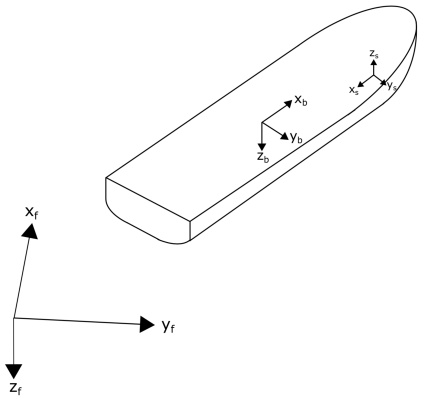



**Figure 5:** Basin $\{f\}$, body $\{b\}$ and sensor $\{s\}$ reference frame (Udjus (2017))

### 4.1.2 Vectorial notation

The vectorial notation in this thesis is adopted from Fossen (2011), and uses sub- and super scripts describing which reference frames the vector is described in and related to.

$\boldsymbol{p}^n_{b/n} = [N, E, D]^\top$: position of $o_b$ with respect to $\{n\}$ expressed in $\{n\}$

$\boldsymbol{\Theta}_{nb} = [\phi, \theta, \psi]^\top$: Euler angles between $\{n\}$ and $\{b\}$

$\boldsymbol{v}^b_{b/n} = [u, v, w]^\top$: linear velocity of $o_b$ with respect to $\{n\}$ expressed in $\{b\}$

$\boldsymbol{\omega}^b_{b/n} = [p, q, r]^\top$: angular velocity of $\{b\}$ with respect to $\{n\}$ expressed in $\{b\}$

$\boldsymbol{a}^b_{b/n} = [a_x, a_y, a_z]^\top$: linear acceleration of $o_b$ with respect to $\{n\}$ expressed in $\{b\}$

$\boldsymbol{\alpha}^b_{b/n} = [\alpha_x, \alpha_y, \alpha_z]^\top$: angular acceleration of $\{b\}$ with respect to $\{n\}$ expressed in $\{b\}$

$\boldsymbol{f}^b_b = [X, Y, Z]^\top$: force with line of action through $o_b$ expressed in $\{b\}$

$\boldsymbol{m}^b_b = [K, M, N]^\top$: moment about $o_b$ expressed in $\{b\}$

This enables the description of the motion of a vessel in 6 DOF by

$$\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{p}^n_{b/n} \\ \boldsymbol{\Theta}_{nb} \end{bmatrix}, \quad \boldsymbol{\nu} = \begin{bmatrix} \boldsymbol{v}^b_{b/n} \\ \boldsymbol{\omega}^b_{b/n} \end{bmatrix}, \quad \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{f}^b_b \\ \boldsymbol{m}^b_b \end{bmatrix}. \tag{1}$$

### 4.1.3 Transformations

An important part of kinematic equations are the transformation matrices

$$\boldsymbol{J}_\Theta(\boldsymbol{\eta}) = \begin{bmatrix} \boldsymbol{R}^n_b(\boldsymbol{\Theta}_{nb}) & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb}) \end{bmatrix}. \tag{2}$$

They vary depending on which reference frames the transformation is between. The linear velocities and accelerations are transformed through the rotation matrix $\boldsymbol{R}(\boldsymbol{\Theta})$ according to the zyx-convention - the axis rotations are done in the order of zyx. The matrix has the property $\boldsymbol{R}_b^n(\boldsymbol{\Theta})^{-1} = \boldsymbol{R}_b^n(\boldsymbol{\Theta})^\top = \boldsymbol{R}_n^b(\boldsymbol{\Theta})$.

$$\begin{aligned}\dot{\boldsymbol{p}}_{b/n}^n &= \boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb})\boldsymbol{v}_{b/n}^b = \boldsymbol{R}_{x,\phi}\boldsymbol{R}_{y,\theta}\boldsymbol{R}_{z,\psi}\boldsymbol{v}_{b/n}^b \\ &= \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \boldsymbol{v}_{b/n}^b. \end{aligned} \tag{3}$$

Angular velocities and accelerations are transformed between reference frames using the transformation matrix $\boldsymbol{T}_\Theta(\boldsymbol{\Theta})$. This matrix, however, does not have the same property as the rotation matrix $\boldsymbol{T}_\Theta(\boldsymbol{\Theta})^{-1} \neq \boldsymbol{T}_\Theta(\boldsymbol{\Theta})^\top$.

$$\dot{\boldsymbol{\Theta}}_{nb} = \boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb})\boldsymbol{\omega}_{b/n}^b = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix} \boldsymbol{\omega}_{b/n}^b. \tag{4}$$

Worth noticing is that the derivative of a rotation matrix is $\dot{\boldsymbol{R}}_n^b = \boldsymbol{R}_n^b \boldsymbol{S}(\boldsymbol{\omega}_{bn}^n)$, where $\boldsymbol{S}(\boldsymbol{\omega})$ is the skew-symmetric matrix for some $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^\top$, i.e.,

$$\boldsymbol{S}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{5}$$

### 4.1.4 Kinematic equations

The vectorial notation and the transformation matrices are used to express the 6 DOF kinematic equations for a vessel. Velocities are transformed from $\{b\}$ to $\{n\}$ as

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{J}_\Theta(\boldsymbol{\eta})\boldsymbol{\nu} \\ &\Updownarrow \\ \begin{bmatrix} \dot{\boldsymbol{p}}_{b/n}^n \\ \dot{\boldsymbol{\Theta}}_{nb} \end{bmatrix} &= \begin{bmatrix} \boldsymbol{R}_b^n(\boldsymbol{\Theta}_{nb}) & \boldsymbol{0}_{3x3} \\ \boldsymbol{0}_{3x3} & \boldsymbol{T}_\Theta(\boldsymbol{\Theta}_{nb}) \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_{b/n}^b \\ \boldsymbol{\omega}_{b/n}^b \end{bmatrix}. \end{aligned} \tag{6}$$

## 4.2 Kinetics

The kinetic equations describe the relationship between forces acting on the vessel and the motion of the vessel. According to Fossen (2011)

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{G}\boldsymbol{\eta} = \boldsymbol{\tau}_{wave} + \boldsymbol{\tau} \tag{7}$$

or in presence of 2D irrotational current $\boldsymbol{\nu}_r = \boldsymbol{\nu} - \boldsymbol{\nu}_c$, where $\dot{\boldsymbol{\nu}} \approx 0$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}}_r + \boldsymbol{C}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{G}\boldsymbol{\eta} = \boldsymbol{\tau}_{wave} + \boldsymbol{\tau}. \tag{8}$$

The matrices can be decomposed accordingly; the mass constant $\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_A$, representing rigid body and added-mass contributions. Coriolis and centripetal term $\boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C}_{RB}(\boldsymbol{\nu}) + \boldsymbol{C}_A(\boldsymbol{\nu})$, representing rigid-body and added mass contributions. The damping term $\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_L + \boldsymbol{D}_{NL}(\boldsymbol{\nu})$, representing the linear and non-linear contributions. Lastly, $\boldsymbol{G}$ is the restoring matrix.

The equation can be categorized into rigid-body, hydrodynamic, and hydrostatic forces as

$$\underbrace{\boldsymbol{M}_{RB}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu}}_{\text{rigid-body forces}} + \underbrace{\boldsymbol{M}_A\dot{\boldsymbol{\nu}}_r + \boldsymbol{C}_A(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r}_{\text{hydrodynamic forces}} + \underbrace{\boldsymbol{G}\boldsymbol{\eta}}_{\text{hydrostatic forces}} = \boldsymbol{\tau} + \boldsymbol{\tau}_{wave}. \tag{9}$$

## 4.3 Simulation verification model

The Simulation verification model (SVM) takes the low level actuator set-points as input and outputs the updated pose of the ship $\boldsymbol{\eta}$. The 6 DOF SVM has been adopted from Perez and Fossen (2007) and is modelled using the kinematic equation presented in section 4.1.4. The kinetic equation (9) is modified under the assumption of zero-speed potential coefficients. In addition speed-dependent fluid memory effects are included

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{R}_b^n(\psi)\boldsymbol{\nu} \\ \boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{RB}\boldsymbol{\nu} + \boldsymbol{C}_A\boldsymbol{\nu}_r + \boldsymbol{D}\boldsymbol{\nu}_r + \boldsymbol{\mu} + \boldsymbol{G}\boldsymbol{\eta} &= \boldsymbol{\tau}_{wave} + \boldsymbol{\tau} \end{aligned} \tag{10}$$

where the speed-dependent fluid memory effects are defined as

$$\boldsymbol{\mu} = \int_0^t \boldsymbol{K}(t-\tau)[\boldsymbol{\nu}(\tau) - U\boldsymbol{e}_1]d\tau. \tag{11}$$

The wave forces $\boldsymbol{\tau}_{wave}$ are modeled using a wave block created by Oyvind Smogeli and a response amplitude operator block made by aforementioned and Thor I. Fossen. Both blocks are from

the MSS toolbox. The wave block calculates harmonic wave component parameters from a wave spectrum based on the inputs of significant wave height, peak frequency, direction, and number of frequencies. The wave component parameters in addition to the vessel low-frequency motion are, through the RAO block, used to calculate the wave-frequency excitation forces.

The thruster dynamics transforms the control outputs $\boldsymbol{u}$ and $\boldsymbol{\alpha}$ to the actuator forces acting on the vessel through

$$\boldsymbol{\tau}(\boldsymbol{u}, \boldsymbol{\alpha}) = \boldsymbol{T}(\boldsymbol{\alpha})\boldsymbol{K}_T\boldsymbol{u}. \tag{12}$$

The angles off the thrusters are all fixed to $\boldsymbol{\alpha} = [\pi, \frac{\pi}{4}, \frac{-\pi}{4}, 0, \frac{5\pi}{4}, \frac{3\pi}{4}]$.

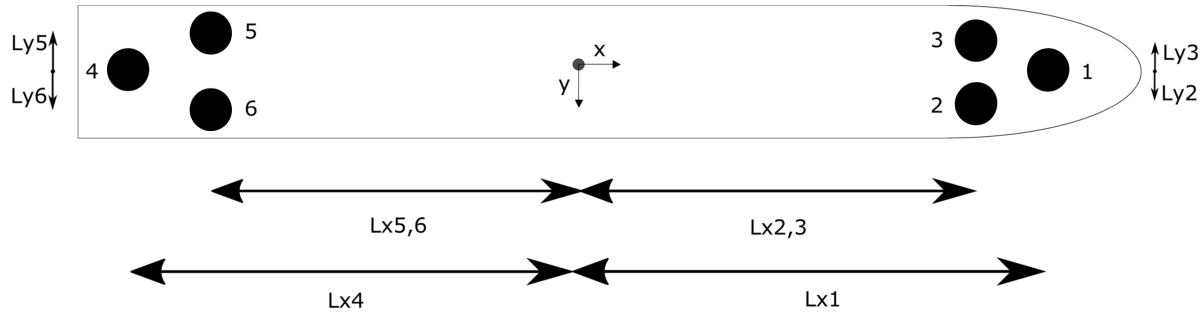



**Figure 6:** Thruster configuration CSAD (Frederich (2016))

**Table 1:** Thruster locations CSAD model (Frederich (2016))

| Thruster | Position X[m] | Position Y[m] |
|---|---|---|
| 1 | 1.0678 | 0.0 |
| 2 | 0.9344 | 0.11 |
| 3 | 0.9344 | -0.11 |
| 4 | -1.1644 | 0.0 |
| 5 | -0.9911 | -0.1644 |
| 6 | -0.9911 | 0.1644 |

## 4.4 Control design model

Due to assumption of low speed manoeuvring, the simplified, low-frequency vessel dynamics are used for the control design model (CDM). The control system only acts in the horizontal plane, and is therefore formulated in 3 DOF - surge, sway, and yaw. The kinematic and kinetic equations are given by

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{13}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{D}_L\boldsymbol{\nu} = \boldsymbol{R}(\psi)\boldsymbol{b} + \boldsymbol{\tau}. \tag{14}$$

The bias $\boldsymbol{b}$ approximates the sum of the low frequency environmental forces and moments, as well as model inaccuracies. The first-order Markov model is used

$$\dot{\boldsymbol{b}} = -\boldsymbol{T}_b^{-1}\boldsymbol{b} + \boldsymbol{E}_b\boldsymbol{w}_b, \tag{15}$$

where $\boldsymbol{T}_b$ is a diagonal matrix consisting of the time constants in 3 DOF, while $\boldsymbol{E}_b$ scales the zero mean white noise $\boldsymbol{w}_b$. The state-space representations of the wave spectra is adopted from Fossen (2011). The motion RAO is formulated as a state-space model, where a second-order system of relative degree one approximates the true wave spectrum

$$h(s) = \frac{K_w s}{s^2 + 2\lambda\omega_0 s + \omega_0^2}. \tag{16}$$

The the gain is defined according to $K_w = 2\lambda\omega_0\sigma$, $\sigma$ describe the wave intensity, $\lambda$ is a damping coefficient, and $\omega_0$ is the dominating wave frequency. By transforming to time domain, the state space model with zero mean white noise $w$ becomes

$$\dot{\boldsymbol{x}}_w = \underbrace{\begin{bmatrix} \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \\ -\boldsymbol{\Omega}_0^2 & -2\boldsymbol{\Lambda}\boldsymbol{\Omega}_0 \end{bmatrix}}_{\boldsymbol{A}_w} \boldsymbol{x}_w + \underbrace{\begin{bmatrix} \boldsymbol{0}_{3\times3} \\ \boldsymbol{K}_w \end{bmatrix}}_{\boldsymbol{E}_w} \boldsymbol{w} \tag{17}$$

$$\boldsymbol{y}_w = \underbrace{\begin{bmatrix} \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \end{bmatrix}}_{\boldsymbol{C}_w} \boldsymbol{x}_w \tag{18}$$

where $\boldsymbol{\Omega}_0 = diag([\omega_1, \omega_2, \omega_3])$ and $\boldsymbol{\Lambda} = diag([\lambda_1, \lambda_2, \lambda_3])$. The resulting CDM that will serve as basis for the model-based observer and controller, i.e,

$$\dot{\boldsymbol{x}}_w = \boldsymbol{A}_w\boldsymbol{x}_w + \boldsymbol{E}_w\boldsymbol{w} \tag{19a}$$

$$\dot{\boldsymbol{\eta}} = \boldsymbol{R}(\psi)\boldsymbol{\nu} \tag{19b}$$

$$\dot{\boldsymbol{b}} = -\boldsymbol{T}_b^{-1}\boldsymbol{b} + \boldsymbol{E}_b\boldsymbol{w}_b \tag{19c}$$

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{D}_L\boldsymbol{\nu} = \boldsymbol{R}(\psi)\boldsymbol{b} + \boldsymbol{\tau} \tag{19d}$$

$$\boldsymbol{y} = \boldsymbol{\eta} + \boldsymbol{C}_\omega\boldsymbol{x}_w \tag{19e}$$

## 4.5 Measurement modeling

### 4.5.1 IMU measurement modeling

An inertial measurement unit (IMU) measures linear specific force and angular velicities in the sensor frame $\{s\}$ with respect to an inertial frame. The sensor frame has to be aligned with the body frame $\{b\}$. If not the sensor data first has to be rotated into alignment with $\{b\}$. The output of the IMU is the linear specific force in the sensor frame, including the acceleration of gravity, bias, and noise. Inspired by Skjetne (2018), let $\{a\}$ be an inertial frame

$$\boldsymbol{f}^s_{s/a} = \boldsymbol{R}^s_a(\ddot{\boldsymbol{p}}^a_{s/a} - \bar{\boldsymbol{g}}^a) + \boldsymbol{b}^s_{a_s} + \boldsymbol{w}^s_{a_s}, \tag{20}$$

where $\boldsymbol{b}^s_{a_s}$ is the bias, $\bar{\boldsymbol{g}}^a$ is the gravitational vector and $\boldsymbol{w}^s_{a_s}$ is the noise. Expressed in $\{b\}$, the measurement equation is formulated as

$$\begin{aligned}
\boldsymbol{f}^b_{s/a} &= \boldsymbol{a}^b_{b/a} - \boldsymbol{R}^b_a\bar{\boldsymbol{g}}^a + \boldsymbol{S}\left(\boldsymbol{\omega}^b_{ab}\right)^2\boldsymbol{l}^b_s + \boldsymbol{S}\left(\boldsymbol{\alpha}^b_{ab}\right)\boldsymbol{l}^b_s + \boldsymbol{b}_{a_s} + \boldsymbol{w}_{a_s} && (21)\\
&= \boldsymbol{a}^b_{b/a} - \bar{\boldsymbol{g}}^b + \boldsymbol{\omega}^b_{b/a} \times \left(\boldsymbol{\omega}^b_{b/a} \times \boldsymbol{l}^b_s\right) + \boldsymbol{\alpha}^b_{b/a} \times \boldsymbol{l}^b_s + \boldsymbol{b}_{a_s} + \boldsymbol{w}_{a_s}. && (22)
\end{aligned}$$

Under the assumption of inertial NED frame $\{n\}$ the vessel motion equations can be stated as

$$\begin{aligned}
\boldsymbol{p}^n_{s/n} &= \boldsymbol{p}^n_{b/n} + \boldsymbol{R}^n_b\boldsymbol{l}^b_s && (23)\\
\dot{\boldsymbol{p}}^n_{s/n} &= \dot{\boldsymbol{p}}^n_{b/n} + \dot{\boldsymbol{R}}^n_b\boldsymbol{l}^b_s\\
&= \dot{\boldsymbol{p}}^n_{b/n} + \boldsymbol{R}^n_b\boldsymbol{S}(\boldsymbol{\omega}^b_{b/n})\boldsymbol{l}^b_s && (24)\\
\ddot{\boldsymbol{p}}^n_{s/n} &= \ddot{\boldsymbol{p}}^n_{b/n} + \dot{\boldsymbol{R}}^n_b\boldsymbol{S}(\boldsymbol{\omega}^b_{b/n})\boldsymbol{l}^b_s + \boldsymbol{R}^n_b\boldsymbol{S}(\dot{\omega}^b_{b/n})\boldsymbol{l}^b_s\\
&= \ddot{\boldsymbol{p}}^n_{b/n} + \boldsymbol{R}^n_b\boldsymbol{S}(\boldsymbol{\omega}^b_{b/n})^2\boldsymbol{l}^b_s + \boldsymbol{R}^n_b\boldsymbol{S}(\dot{\boldsymbol{\omega}}^b_{b/n})\boldsymbol{l}^b_s. && (25)\\
& && (26)
\end{aligned}$$

Relating to the inertial ECI frame $\{i\}$ gives

$$
\begin{aligned}
\boldsymbol{p}^i_{s/i} &= \boldsymbol{p}^i_{e/i} + \boldsymbol{p}^i_{n/e} + \boldsymbol{p}^i_{s/n} \\
&= \boldsymbol{R}^i_e \left( \boldsymbol{p}^e_{n/e} + \boldsymbol{R}^e_n \boldsymbol{p}^n_{s/n} \right) && (27) \\
\dot{\boldsymbol{p}}^i_{s/i} &= \dot{\boldsymbol{R}}^i_e \left( \boldsymbol{p}^e_{n/e} + \boldsymbol{R}^e_n \boldsymbol{p}^n_{s/n} \right) + \boldsymbol{R}^i_e \left( \dot{\boldsymbol{p}}^e_{n/e} + \dot{\boldsymbol{R}}^e_n \boldsymbol{p}^n_{s/n} + \boldsymbol{R}^e_n \dot{\boldsymbol{p}}^n_{s/n} \right) \\
&= \boldsymbol{R}^i_e \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right) \left( \boldsymbol{p}^e_{n/e} + \boldsymbol{R}^e_n \boldsymbol{p}^n_{s/n} \right) + \boldsymbol{R}^i_e \boldsymbol{R}^e_n \dot{\boldsymbol{p}}^n_{s/n} && (28) \\
\ddot{\boldsymbol{p}}^i_{s/i} &= \dot{\boldsymbol{R}}^i_e \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right) \left( \boldsymbol{p}^e_{n/e} + \boldsymbol{R}^e_n \boldsymbol{p}^n_{s/n} \right) + \boldsymbol{R}^i_e \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right) \left( \dot{\boldsymbol{p}}^e_{n/e} + \dot{\boldsymbol{R}}^e_n \boldsymbol{p}^n_{s/n} + \boldsymbol{R}^e_n \dot{\boldsymbol{p}}^n_{s/n} \right) \\
&\quad + \dot{\boldsymbol{R}}^i_e \boldsymbol{R}^e_n \dot{\boldsymbol{p}}^n_{s/n} + \boldsymbol{R}^i_e \dot{\boldsymbol{R}}^e_n \dot{\boldsymbol{p}}^n_{s/n} + \boldsymbol{R}^i_e \boldsymbol{R}^e_n \ddot{\boldsymbol{p}}^n_{s/n} \\
&= \boldsymbol{R}^i_e \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right)^2 \boldsymbol{p}^e_{s/e} + 2 \boldsymbol{R}^i_e \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right) \boldsymbol{R}^e_n \left( \dot{\boldsymbol{p}}^n_{b/n} + \boldsymbol{R}^n_b \boldsymbol{S} \left( \boldsymbol{\omega}^b_{b/n} \right) \boldsymbol{l}^b_s \right) \\
&\quad + \boldsymbol{R}^i_e \boldsymbol{R}^e_n \left[ \ddot{\boldsymbol{p}}^n_{b/n} + \boldsymbol{R}^n_b \boldsymbol{S} \left( \boldsymbol{\omega}^b_{b/n} \right)^2 \boldsymbol{l}^b_s + \boldsymbol{R}^n_b \boldsymbol{S} \left( \dot{\boldsymbol{\omega}}^b_{b/n} \right) \boldsymbol{l}^b_s \right]. && (29)
\end{aligned}
$$

Utilize the fact that $\boldsymbol{p}^i_{e/i} = 0$, $\dot{\boldsymbol{R}}^e_n = 0$, $\boldsymbol{p}^e_{n/e} = 0$ and $\dot{\boldsymbol{\omega}}^e_{e/i} = 0$.

$$
\begin{aligned}
\boldsymbol{v}^b_{b/n} &:= \boldsymbol{R}^b_n \dot{\boldsymbol{p}}^n_{b/n}, \quad \dot{\boldsymbol{p}}^n_{b/n} = \boldsymbol{R}^n_b \boldsymbol{v}^b_{b/n} && (30) \\
\boldsymbol{a}^b_{b/n} &:= \boldsymbol{R}^b_n \dot{\boldsymbol{v}}^n_{b/n} = \boldsymbol{R}^b_n \ddot{\boldsymbol{p}}^n_{b/n}, \quad \ddot{\boldsymbol{p}}^n_{b/n} = \boldsymbol{R}^n_b \boldsymbol{a}^b_{b/n} && (31) \\
\boldsymbol{a}^b_{b/n} &= \dot{\boldsymbol{v}}^b_{b/n} + \boldsymbol{S} \left( \boldsymbol{\omega}^b_{b/n} \right) \boldsymbol{v}^b_{b/n} && (32) \\
\boldsymbol{\alpha}^b_{b/n} &= \dot{\boldsymbol{\omega}}^b_{b/n} && (33)
\end{aligned}
$$

Substituting (30)-(33) into (29), introducing the 3 DOF linear velocities and accelerations as well as angular acceleration, yields

$$
\begin{aligned}
\ddot{\boldsymbol{p}}^i_{s/i} &= \boldsymbol{R}^i_b \left[ \boldsymbol{a}^b_{b/n} + \boldsymbol{S} \left( \boldsymbol{\omega}^b_{b/n} \right)^2 \boldsymbol{l}^b_s + \boldsymbol{S} \left( \boldsymbol{\alpha}^b_{b/n} \right) \boldsymbol{l}^b_s \right] \\
&\quad + \boldsymbol{R}^i_e \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right)^2 \boldsymbol{p}^e_{s/e} + 2 \boldsymbol{R}^i_e \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right) \boldsymbol{R}^e_n \boldsymbol{R}^n_b \left( \boldsymbol{v}^b_{b/n} + \boldsymbol{S} \left( \boldsymbol{\omega}^b_{b/n} \right) \boldsymbol{l}^b_s \right). && (34)
\end{aligned}
$$

Inserting into the measurement equation, with $\boldsymbol{g}^e = \bar{\boldsymbol{g}}^e - \boldsymbol{S} \left( \boldsymbol{\omega}^e_{e/i} \right)^2 \boldsymbol{p}^e_{s/e}$

$$
\begin{aligned}
\boldsymbol{f}_s^b &= \boldsymbol{R}_i^b\left(\ddot{\boldsymbol{p}}_{s/i}^i - \boldsymbol{R}_e^i\bar{\boldsymbol{g}}^e\right) + \boldsymbol{b}_{a_s}^b + \boldsymbol{w}_{a_s}^b \\
&= \boldsymbol{a}_{b/n}^b - \boldsymbol{R}_n^b\boldsymbol{g}^n + \boldsymbol{S}\left(\boldsymbol{\omega}_{b/n}^b\right)^2\boldsymbol{l}_s^b + \boldsymbol{S}\left(\boldsymbol{\alpha}_{b/n}^b\right)\boldsymbol{l}_s^b + 2\boldsymbol{S}\left(\boldsymbol{\omega}_{e/i}^b\right)\left(\boldsymbol{v}_{b/n}^b + \boldsymbol{S}\left(\boldsymbol{\omega}_{b/n}^b\right)\boldsymbol{l}_s^b\right) + \boldsymbol{b}_{a_s}^b + \boldsymbol{w}_{a_s}^b \\
&= \dot{\boldsymbol{v}}_{b/n}^b + \boldsymbol{S}\left(\boldsymbol{\omega}_{b/n}^b\right)\boldsymbol{v}_{b/n}^b + \boldsymbol{S}\left(\boldsymbol{\omega}_{b/n}^b\right)^2\boldsymbol{l}_s^b + \boldsymbol{S}\left(\boldsymbol{\alpha}_{b/n}^b\right)\boldsymbol{l}_s^b + 2\boldsymbol{S}\left(\boldsymbol{\omega}_{e/i}^b\right)\left(\boldsymbol{v}_{b/n}^b + \boldsymbol{S}\left(\boldsymbol{\omega}_{b/n}^b\right)\boldsymbol{l}_s^b\right) \\
&- \boldsymbol{R}_n^b\boldsymbol{g}^n + \boldsymbol{b}_{a_s}^b + \boldsymbol{w}_{a_s}^b
\end{aligned} \tag{35}
$$

which constitutes as the IMU simulation equation for a single IMU placed at arm $\boldsymbol{l}_s^b$ from $o_b$, generating specific force measurements expressed in $\{b\}$.

### 4.5.2 IMU sensor fusion

For redundancy it is often desirable combining the measurements of several IMUs to accurately identify the linear and angular accelerations of the body. Inspired by Skjetne (2018), using the knowledge of the location of each sensor in $\{b\}$ and the Euler accelerations, we first define

$$
\boldsymbol{z} := \operatorname{col}\left(\boldsymbol{a}_{b/i}^b, \boldsymbol{\alpha}_{b/i}^b, \bar{\boldsymbol{\omega}}\right) \in \mathbb{R}^{12} \tag{36}
$$

$$
\boldsymbol{\xi}_{l_s} := \boldsymbol{a}_{s/i}^b = \boldsymbol{a}_{b/i}^b - \boldsymbol{S}\left(\boldsymbol{l}_s\right)\boldsymbol{\alpha}_{b/i}^b + \boldsymbol{H}\left(\boldsymbol{l}_s\right)\bar{\boldsymbol{\omega}} \in \mathbb{R}^3 \tag{37}
$$

where $\boldsymbol{l}_s = \boldsymbol{l}_s^b = col\left(l_x, l_y, l_z\right)$ and

$$
\bar{\boldsymbol{\omega}} := \operatorname{col}\left(\omega_x^2, \omega_y^2, \omega_z^2, \omega_x\omega_y, \omega_x\omega_z, \omega_y\omega_z\right) \in \mathbb{R}^6 \tag{38}
$$

$$
\boldsymbol{H}\left(\boldsymbol{l}_s\right) := \begin{bmatrix} 0 & -l_x & -l_x & l_y & l_z & 0 \\ -l_y & 0 & -l_y & l_x & 0 & l_z \\ -l_z & -l_z & 0 & 0 & l_x & l_y. \end{bmatrix}. \tag{39}
$$

By defining

$$
\boldsymbol{W}_{l_s} := \begin{bmatrix} \boldsymbol{I}_{3\times3} & -\boldsymbol{S}\left(\boldsymbol{l}_s^b\right) & \boldsymbol{H}\left(\boldsymbol{l}_s^b\right) \end{bmatrix} \in \mathbb{R}^{3\times12} \tag{40}
$$

we get

$$
\boldsymbol{\xi}_{l_s} = \boldsymbol{a}_{s/i}^b = \boldsymbol{W}_{l_s}\boldsymbol{z}. \tag{41}
$$

Inserting into the specific force measurement equation $\boldsymbol{f}_s^b = \boldsymbol{a}_{s/i}^b - \bar{\boldsymbol{g}}^b + \boldsymbol{b}_{a_s} + \boldsymbol{w}_{a_s}$ for an IMU $s$, we get

$$\boldsymbol{f}_s^b = \boldsymbol{W}_{l_s}\boldsymbol{z} - \bar{\boldsymbol{g}}^b + \boldsymbol{b}_{a_s} + \boldsymbol{w}_{a_s}. \tag{42}$$

In this case with four IMUs, and $\boldsymbol{l} := \operatorname{col}(l_1, l_2, l_3, l_4) \in \mathbb{R}^{12}$ we define

$$\boldsymbol{\xi} := \begin{bmatrix} \boldsymbol{\xi}_{l_1} \\ \boldsymbol{\xi}_{l_2} \\ \boldsymbol{\xi}_{l_3} \\ \boldsymbol{\xi}_{l_4} \end{bmatrix} = \begin{bmatrix} \boldsymbol{a}_{s_1/i}^b \\ \boldsymbol{a}_{s_2/i}^b \\ \boldsymbol{a}_{s_3/i}^b \\ \boldsymbol{a}_{s_4/i}^b \end{bmatrix} = \begin{bmatrix} \boldsymbol{W}_{l_1} \\ \boldsymbol{W}_{l_2} \\ \boldsymbol{W}_{l_3} \\ \boldsymbol{W}_{l_4} \end{bmatrix} \boldsymbol{z} =: \boldsymbol{W}(\boldsymbol{l})\boldsymbol{z}, \quad \boldsymbol{W}(\boldsymbol{l}) \in \mathbb{R}^{12\times 12}. \tag{43}$$

According to Zappa et al. (2001) the matrix $W(l)$ is invertible as long as the IMUs are not coplanar - at least one IMU is not in the plane spanned by the three others. The Moore-Penrose pseudoinverse is hence applicable

$$\boldsymbol{z} = \boldsymbol{W}(\boldsymbol{l})^\dagger \boldsymbol{\xi}. \tag{44}$$

This yields the combined measurement equation

$$\boldsymbol{f}_c^b = \boldsymbol{W}(\boldsymbol{l})\boldsymbol{z} - \boldsymbol{1}_N \otimes \bar{\boldsymbol{g}}^b + \boldsymbol{b}_c + \boldsymbol{w}_c \tag{45}$$

$$\boldsymbol{z} = \boldsymbol{W}(\boldsymbol{l})^\dagger \left[\boldsymbol{f}_c^b + \boldsymbol{1}_N \otimes \bar{\boldsymbol{g}}^b\right] - \boldsymbol{W}(l)^\dagger \left[\boldsymbol{b}_c + \boldsymbol{w}_c\right] \tag{46}$$

where $\boldsymbol{A} \otimes \boldsymbol{B}$ is the Kronecher Product and the combined measurement, bias and noise defined as

$$\boldsymbol{f}_c^b := \operatorname{col}\left(f_1^b, f_2^b, f_3^b, f_4^b\right) \in \mathbb{R}^{12} \tag{47}$$

$$\boldsymbol{b}_c := \operatorname{col}\left(b_{a_1}, b_{a_2}, b_{a_3}, b_{a_4}\right) \in \mathbb{R}^{12} \tag{48}$$

$$\boldsymbol{w}_c := \operatorname{col}\left(w_{a_1}, w_{a_2}, w_{a_3}, w_{a_4}\right) \in \mathbb{R}^{12}. \tag{49}$$

Using

$$\begin{aligned} \boldsymbol{H}_a &:= \begin{bmatrix} \boldsymbol{I}_{3\times 3} & \boldsymbol{0}_{3\times 9} \end{bmatrix} \in \mathbb{R}^{3\times 12} \\ \boldsymbol{H}_\alpha &:= \begin{bmatrix} \boldsymbol{0}_{3\times 3} & \boldsymbol{I}_{3\times 3} & \boldsymbol{0}_{3\times 6} \end{bmatrix} \in \mathbb{R}^{3\times 12} \end{aligned} \tag{50}$$

and the assumption of accurate signals for $\boldsymbol{\omega}^b_{b/n}(t)$, $\boldsymbol{f}^b_c(t)$, and $\boldsymbol{p}^i_{b/i}(t) = \boldsymbol{p}^i_{b/e}(t) = \boldsymbol{R}^i_e(t)\boldsymbol{p}^e_{b/e}(t)$, the linear and angular acceleration of the vessel center of control with respect to $\{i\}$ is

$$\begin{aligned} \boldsymbol{a}^b_{b/i} &= \boldsymbol{H}_a \boldsymbol{z} \\ \boldsymbol{\alpha}^b_{b/i} &= \boldsymbol{H}_\alpha \boldsymbol{z}. \end{aligned} \tag{51}$$

## 4.6 Waves and vessel response

Long-crested irregular sea is modeled as a sum of waves with different frequencies and amplitudes all going in the same direction - the pattern is constant along the breadth of the wave. Long-crested regular sea is simply modelled as a single wave. The elevation of the surface can be described according to

$$\zeta(t) = \sum_{i=1}^{n} a_i \cos\left(\omega_i t + \epsilon_i\right) \tag{52}$$

where $a_i$ is the amplitude, $\omega_i$ the frequency, and $\epsilon_i$ the phase randomly sampled from a uniform distribution.

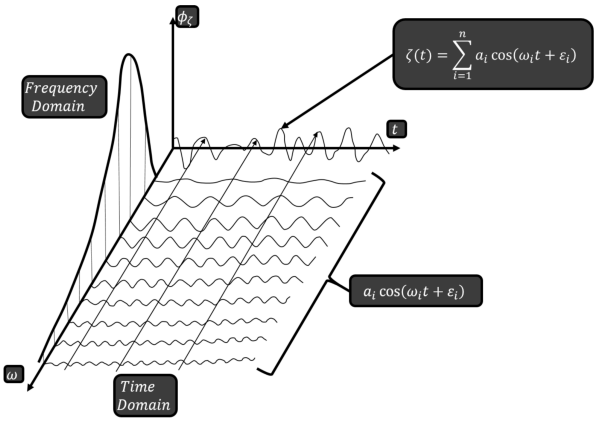



**Figure 7:** Visualization of the structure of a wave spectrum (Ferrandis et al. (2019))

The energy distribution over the wave frequencies in long-crested sea is described by the wave power spectrum $S(\omega)$. The statistical properties used to set the different sea states are significant wave height $H_s$ and peak period $T_p$. The first describes the mean height of the 1/3 highest waves, while the second describe the period and hence the frequency in the irregular sea with the highest power. They are related to the wave spectrum through

$$H_s := 4\sqrt{m_0}, \quad m_0 := \int_0^\infty S(\omega)d\omega \tag{53}$$

$$T_p := \frac{2\pi}{\omega_p}, \quad \omega_p := \arg\max_j S(\omega_j). \tag{54}$$

The wave direction $\Theta$, or relative wave direction $\beta$, indicate the direction of the long-crested irregular sea, see figure 8. The open sea is divided into 12 sectors, each representing $\pi/6$ rad or 30°.

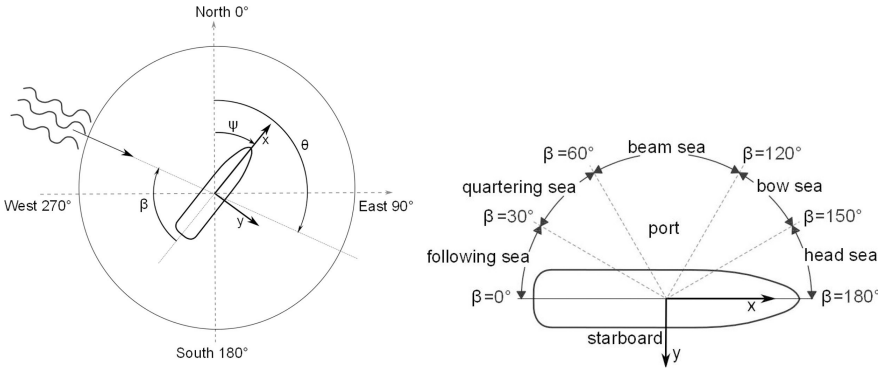



**Figure 8:** Definition of wave direction sectors (Brodtkorb et al. (2018b))

**Table 2:** Combinations of significant wave heights ($H_s$) and peak periods ($T_p$) (Price (1974))

| Sea state | Description | $H_s$ [m] | $T_p$ [s] |
|---|---|---|---|
| 0 | Calm (glossy) | 0 | - |
| 1 | Calm (rippled) | 0 - 0.1 | 4.87 - 5.66 |
| 2 | Smooth (wavelets) | 0.1 - 0.5 | 5.66 - 6.76 |
| 3 | Slight | 0.5 - 1.25 | 6.76 - 7.95 |
| 4 | Moderate | 1.25 - 2.5 | 7.95 - 9.24 |
| 5 | Rough | 2.5 - 4.0 | 9.24 - 10.47 |
| 6 | Very rough | 4.0 - 6.0 | 10.47 - 11.86 |
| 7 | High | 6.0 - 9.0 | 11.86 - 13.66 |
| 8 | Very high | 9.0 - 14.0 | 13.66 - 16.11 |
| >8 | Phenomenal | > 14 | > 16.11 |

Table 2 shows the most commonly used definition for sea states credited Price (1974). The CSAD vessel used in this thesis is a 1:90 model ship, and the wave height is thus scaled down with a factor of 100 for simplicity. Now the dispersion function for deepwater is $\lambda = \frac{g}{2\pi}T^2$. If the vessel is scaled down to $1/k$, then the corresponding wave length is scaled with $1/k$ and the wave period $\sqrt{1/k}$. The wave period has therefore been scaled down by a factor $\sqrt{90}$. The vessel works like a lowpass

filter in the way that small wave lengths compared to the vessel dimensions will not be registered by the sensors. Only the more powerful sea states have been chosen for this thesis, described in table 3.

| Sea state no. | Description | $H_s$ [m] | $T_p$ [s] |
|---|---|---|---|
| 1 | Moderate | 0.0125 - 0.025 | 0.84 - 0.97 |
| 2 | Rough | 0.025 - 0.04 | 0.97 - 1.1 |
| 3 | Very rough | 0.04 - 0.06 | 1.1 - 1.25 |
| 4 | High | 0.06 - 0.09 | 1.25 - 1.44 |

**Table 3:** The four sea states for irregular long crested sea used in the thesis

In the simulation model, the harmonic wave component parameters are calculated using a block in the MSS toolbox made by Oyvind Notland Smogeli. The spectrum type is ITTC with randomly chosen combination of wave direction, specific wave height and peak frequency. The following static parameters are chosen

- Number of frequencies in grid: 40
- Number of waves: 5
- Frequency cutoff factor: 3

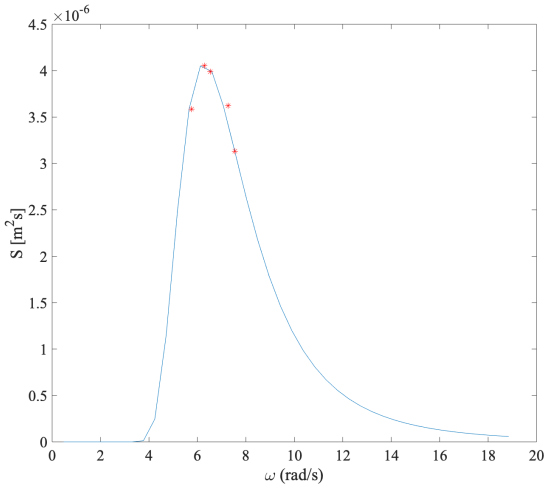



**Figure 9:** Plotted wave spectrum with $H_s = 0.03$ m and $T_p = 1$ s.

# 5 Motion control

The motion control is split into three different subsystems; guidance, control, and navigation. The first subsystem generates the optimal path based on set-points. The second subsystem generates the desired thrust based on sensor signals and desired position from the guidance module. It is further split into a model based motion controller and a fixed thrust allocation. Finally, the navigation module contains a model based observer that filters the sensor signals and recreates unmeasured signals.

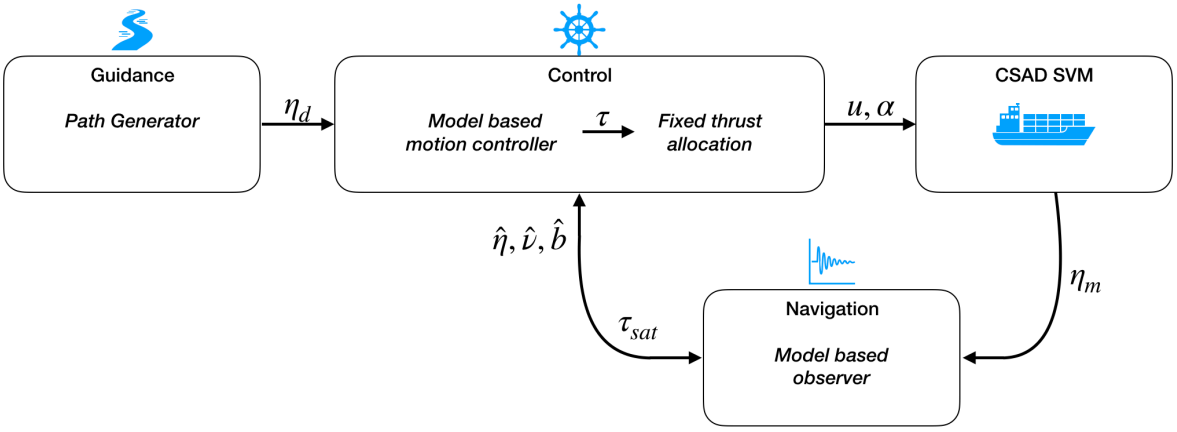



**Figure 10:** The DP control system.

## 5.1 Maneuvering-based guidance design

The guidance algorithm is implemented according to Skjetne (2019), where the objective is to follow a path based on calculated desired positions and velocities along the path. We let $\boldsymbol{p_d} : \mathbb{R} \to \mathbb{R}^2$ be parametrized by the continous path variable $s_1$ and similarly $\psi_d : \mathbb{R} \to \mathbb{R}$ be parametrized by $s_2$. The desired pose then describes a path

$$\boldsymbol{\eta_d}(s) = \begin{bmatrix} \boldsymbol{p}_d(s_1) \\ \psi_d(s_2) \end{bmatrix}, \quad \boldsymbol{s} \in \mathbb{R}^2 \tag{55}$$

Based on the desired path speed $u_d$, the speed assignment can be formulated as

$$v_s(t, s_1) = \frac{u_d(t, s_1)}{|\boldsymbol{p}_d^{s_1}(s_1)|} \tag{56}$$

To accomplish the objective of moving in a straight line from $(\boldsymbol{p}_0, \psi_0)$ to $(\boldsymbol{p}_t, \psi_t)$ the speed along the path shall be governed by $\dot{\boldsymbol{s}}$ which again abide by the limitation of speed and acceleration limitations the CSAD in all three DOFs.

Initially let the path between two points be parametrised by $\boldsymbol{s}$ accordingly

$$\boldsymbol{p}_d(s_1) = \frac{(1+\lambda-s_1)\boldsymbol{p}_0 + s_1\boldsymbol{p}_t}{1+\lambda}, \quad 0 < \lambda \ll 1 \tag{57}$$

$$\psi_d(s_2) = \frac{(1+\lambda-s_2)\psi_0 + s_2\psi_t}{1+\lambda}, \quad 0 < \lambda \ll 1 \tag{58}$$

Then

$$\boldsymbol{p}_d^{s_1}(s_1) = \frac{\boldsymbol{p}_t - \boldsymbol{p}_0}{1+\lambda} = \boldsymbol{p}_d^{s_1} \tag{59}$$

$$\psi_d^{s_2}(s_2) = \frac{\psi_t - \psi_0}{1+\lambda} = \psi_d^{s_2} \tag{60}$$

The resulting position and heading filter becomes

$$\dot{s}_1 = v_s(t, s_1) = \sigma_p(t)\frac{u_s(s_1)}{|\boldsymbol{p}_d^s| + \epsilon} \tag{61}$$

$$\dot{s}_2 = \sigma_\psi(t)\frac{r_s(s_2)}{|\psi_d^s| + \epsilon} \tag{62}$$

The two $\sigma$ parameters are activation signals for the motion in position and heading, $\sigma : \mathbb{R}_{\geq 0} \to \{0, 1\}$

$$\boldsymbol{\eta}_d(s) = \begin{bmatrix} \boldsymbol{p}_d(s_1) \\ \psi_d(s_2) \end{bmatrix}, \quad \boldsymbol{\eta}_d^s = \begin{bmatrix} \boldsymbol{p}_d^{s_1} \\ \psi_d^{s_2} \end{bmatrix}, \quad \boldsymbol{\eta}_d^{s^2} = \begin{bmatrix} \boldsymbol{p}_d^{s_1^2} \\ \psi_d^{s_2^2} \end{bmatrix} \tag{63}$$

## 5.2 Model-based DP controller

The DP controller is implemented from Skjetne (2019), and is intended to accomplish the maneuvering control objective by designing the control law for $\boldsymbol{\tau}$ such that

$$\left.\begin{aligned} \boldsymbol{\eta}(t) &\to \boldsymbol{\eta}_d(s(t)) \\ \dot{s}(t) &\to v_s(t, s) \end{aligned}\right\} \text{as } t \to \infty$$

which implies that with two separate path parameters the control law $\boldsymbol{\tau}$ will ensure

$$\left.\begin{aligned} \boldsymbol{p}(t) &\to \boldsymbol{p}_d(s_1(t)) \\ \psi(t) &\to \psi_d(s_2(t)) \end{aligned}\right\} \text{as } t \to \infty.$$

The LgV backstepping design is chosen, hence let

$$\boldsymbol{z}_1 := \boldsymbol{R}(\psi)^\top \left[\boldsymbol{\eta} - \boldsymbol{\eta}_d(s)\right], \quad \boldsymbol{z}_2 = \boldsymbol{\nu} - \boldsymbol{\alpha}_1, \quad \omega = \dot{s} - v_s(t). \tag{64}$$

**Step 1**

The design follows these steps

$$\dot{\boldsymbol{z}_1} = \dot{\boldsymbol{R}}^\top [\boldsymbol{\eta} - \boldsymbol{\eta}_d] + \boldsymbol{R}(\psi)^\top [\dot{\boldsymbol{\eta}} - \boldsymbol{\eta}_d^s \dot{s}] = -r\boldsymbol{S}\boldsymbol{z}_1 + \boldsymbol{z}_2 + \boldsymbol{\alpha}_1 - \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(\omega + v_s) \tag{65}$$

the skew-symmetric matrix $S$

$$\boldsymbol{S} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{66}$$

The first control Lyapunov function (CLF) is chosen as

$$V_1 = \frac{1}{2}\boldsymbol{z}_1^\top \boldsymbol{z}_1 \tag{67}$$

differentiating

$$\dot{V}_1 = -r\boldsymbol{z}_1^\top \boldsymbol{S}\boldsymbol{z}_1 + \boldsymbol{z}_1^\top \boldsymbol{z}_2 + \boldsymbol{z}_1^\top \left[\boldsymbol{\alpha}_1 - \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(\omega + v_s)\right]. \tag{68}$$

To cancel the unnecessary terms $\boldsymbol{\alpha}_1$ is chosen as

$$\boldsymbol{\alpha}_1 = -\boldsymbol{K}_1\boldsymbol{z}_1 + \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s v_s + \boldsymbol{\alpha}_{10}, \quad \boldsymbol{K}_1 = \boldsymbol{K}_1^\top > 0 \tag{69}$$

and the first tuning function

$$\rho_1 = -\boldsymbol{z}_1^\top \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s. \tag{70}$$

Applying Young's inequality to the derivated CLF gives

$$\dot{V}_1 \leq -\boldsymbol{z}_1^\top \boldsymbol{K}_1\boldsymbol{z}_1 + \rho_1\omega + \kappa_1 \boldsymbol{z}_1^\top \boldsymbol{z}_1 + \frac{1}{4\kappa_1}\boldsymbol{z}_2^\top \boldsymbol{z}_2 + \boldsymbol{z}_1^\top \boldsymbol{\alpha}_{10}, \quad \kappa_1 > 0 \tag{71}$$

with

$$\boldsymbol{\alpha}_{10} = -\kappa_1 \boldsymbol{z}_1 \tag{72}$$

we get

$$\dot{V}_1 \leq -\boldsymbol{z}_1^\top \boldsymbol{K}_1 \boldsymbol{z}_1 + \rho_1 \omega + \frac{1}{4\kappa_1} \boldsymbol{z}_2^\top \boldsymbol{z}_2 \tag{73}$$

$$\boldsymbol{\alpha}_1(t, \boldsymbol{s}, \boldsymbol{\eta}) = -(\boldsymbol{K}_1 + \kappa_1 \boldsymbol{I}) \boldsymbol{R}(\psi)^\top [\boldsymbol{\eta} - \boldsymbol{\eta}_d(s)] + \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(s) v_s(t, s). \tag{74}$$

We now have

$$\rho_1 = -\boldsymbol{z}_1^\top \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s = V_1^s(\boldsymbol{\eta}, s) \tag{75}$$

$$\dot{V}_1 = -\boldsymbol{z}_1^\top \boldsymbol{K}_1 \boldsymbol{z}_1 - \omega \boldsymbol{\eta}_d^s(s)^\top \boldsymbol{R}(\psi) \boldsymbol{z}_1 + \frac{1}{4\kappa_1} \boldsymbol{z}_2^\top \boldsymbol{z}_2. \tag{76}$$

Choosing the tracking update law $\omega = 0$

$$\Rightarrow \dot{s} = v_s(t, s). \tag{77}$$

Finally let $\tilde{\boldsymbol{K}}_1 = \boldsymbol{K}_1 + \kappa_1 \boldsymbol{I}$, then we have

$$\boldsymbol{\alpha}_1(t, \boldsymbol{s}, \boldsymbol{\eta}) = \tilde{\boldsymbol{K}}_1 \boldsymbol{R}(\psi)^\top [\boldsymbol{\eta} - \boldsymbol{\eta}_d(s)] + \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(s) v_s(t, s) \tag{78}$$

$$\dot{\boldsymbol{z}}_1 = -\left( \tilde{\boldsymbol{K}}_1 + r\boldsymbol{S} \right) \boldsymbol{z}_1 + \boldsymbol{z}_2 - \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(s) \omega \tag{79}$$

$$\dot{s} = v_s(t, s) \tag{80}$$

$$\dot{V}_1 \leq -\boldsymbol{z}_1^\top \boldsymbol{K}_1 \boldsymbol{z}_1 + \frac{1}{4\kappa_1} \boldsymbol{z}_2^\top \boldsymbol{z}_2. \tag{81}$$

To cancel $\dot{\boldsymbol{\alpha}}_1$ we get $\dot{\boldsymbol{\alpha}}_1 = \boldsymbol{\sigma}_1 + \boldsymbol{\alpha}_1^s \dot{s}$ where

$$\boldsymbol{\sigma}_1(t, s, \boldsymbol{\eta}, \boldsymbol{\nu}) = r\tilde{\boldsymbol{K}}_1 \boldsymbol{S} \boldsymbol{z}_1 - \tilde{\boldsymbol{K}}_1 \boldsymbol{\nu} - r\boldsymbol{S}\boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(s) v_s(t, s) + \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(s) v_s^t(t, s) \tag{82}$$

$$\boldsymbol{\alpha}_1^s(t, s, \boldsymbol{\eta}) = \tilde{\boldsymbol{K}}_1 \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(s) + \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^{s^2}(s) v_s(t, s) + \boldsymbol{R}(\psi)^\top \boldsymbol{\eta}_d^s(s) v_s^s(t, s). \tag{83}$$

**Step 2**

$$\boldsymbol{M}\dot{\boldsymbol{z}}_2 = \boldsymbol{M}\dot{\boldsymbol{\nu}} - \boldsymbol{M}\dot{\boldsymbol{\alpha}}_1 = -\boldsymbol{D}\boldsymbol{\nu} + \boldsymbol{\tau} + \boldsymbol{R}(\psi)^\top \boldsymbol{b} - \boldsymbol{M}[\boldsymbol{\sigma}_1 + \boldsymbol{\alpha}_1^s \dot{s}]. \tag{84}$$

The new CLF partly consisting of the CLF from part 1

$$V_2 = V_1 + \frac{1}{2} \boldsymbol{z}_2^\top \boldsymbol{M} \boldsymbol{z}_2. \tag{85}$$

Derivating

$$\dot{V}_2 = \dot{V}_1 + \boldsymbol{z}_2 \boldsymbol{M} \dot{\boldsymbol{z}}_2 \tag{86}$$

and inserting previous findings we get

$$\dot{V}_2 \leq -\boldsymbol{z}_1 \boldsymbol{K}_1 \boldsymbol{z}_1 + \frac{1}{4\kappa_1} \boldsymbol{z}_2^\top \boldsymbol{z}_2 + \boldsymbol{z}_2^\top \Big[ -\boldsymbol{D}(\boldsymbol{z}_2 + \boldsymbol{\alpha}_1) + \boldsymbol{\tau} + \boldsymbol{R}(\psi)^\top \boldsymbol{b} - \boldsymbol{M}(\boldsymbol{\sigma}_1 + \boldsymbol{\alpha}_1^s \dot{s}) \Big]. \tag{87}$$

Choosing the control law

$$\boldsymbol{\tau} = -\boldsymbol{K}_2 \boldsymbol{z}_2 + \boldsymbol{D}\boldsymbol{\alpha}_1 - \boldsymbol{R}(\psi)^\top \boldsymbol{b} + \boldsymbol{M}(\boldsymbol{\sigma}_1 + \boldsymbol{\alpha}_1^s \dot{s}), \quad \boldsymbol{K}_2 = \boldsymbol{K}_2^\top > 0 \tag{88}$$

makes the CLF $V_2$ pdf and $\dot{V}_2$ ndf for all $\boldsymbol{z} \neq 0$, under the constraints of $\boldsymbol{K}_1$, $\boldsymbol{K}_2$ and $\kappa_1$

$$\dot{V}_2 \leq -\boldsymbol{z}_1^\top \boldsymbol{K}_1 \boldsymbol{z}_1 - \boldsymbol{z}_2^\top \left( \boldsymbol{K}_2 - \frac{1}{4\kappa_1} \boldsymbol{I} \right) \boldsymbol{z}_2. \tag{89}$$

The final control law and closed-loop system becomes

$$\dot{s} = v_s \tag{90}$$

$$\boldsymbol{\tau} = -\boldsymbol{K}_2 \boldsymbol{z}_2 + \boldsymbol{D}\boldsymbol{\alpha}_1 - \boldsymbol{R}(\psi)^\top \boldsymbol{b} + \boldsymbol{M}(\boldsymbol{\sigma}_1 + \boldsymbol{\alpha}_1^s \dot{s}) \tag{91}$$

with z dynamic

$$\dot{\boldsymbol{z}_1} = -\left( (\boldsymbol{K}_1 + \kappa_1 \boldsymbol{I}) + r\boldsymbol{S} \right) \boldsymbol{z}_1 + \boldsymbol{z}_2 \tag{92}$$

$$\boldsymbol{M}\dot{\boldsymbol{z}}_2 = -(\boldsymbol{D} + \boldsymbol{K}_2)\boldsymbol{z}_2 \tag{93}$$

where $\boldsymbol{K}_1 = \boldsymbol{K}_1^\top > 0$, $\boldsymbol{K}_2 = \boldsymbol{K}_2^\top > 0$ and $\kappa_1 > 0$ are gain matrices tuned using the optimization algorithm *fminsearch*.

## 5.3 Model-based observer

A nonlinear passive observer is implemented mainly due to its tuning simplicity compared to other similar nonlinear observes like the Extended Kalman Filter. It can also be proven globally exponentially stable (GES). The observer takes the position signal $\boldsymbol{\eta}$ as well as the saturated controller calculated desired forces $\boldsymbol{\tau}_{sat}$ and outputs estimates for position $\hat{\boldsymbol{\eta}}$, velocity $\hat{\boldsymbol{\nu}}$, and bias $\hat{\boldsymbol{b}}$.

The first objective of the observer is to filter out the firste order wave forces. It is disadvantageous for the controller to react to these high-frequency signals, as it leads to unnecessary/unachievable energy demand as well as wear and tear on the equipment. The second objective is to reconstruct states that are unmeasurable or expensive to measure, namely the velocity $\hat{\boldsymbol{\nu}}$ and the bias $\hat{\boldsymbol{b}}$. Assuming negligible process and measurement noise $\omega = \upsilon = 0$ and negligible wave-induced yaw rotation $\psi \approx \psi + \psi_w$ the DP observer equations Fossen (2011) can be formulated as follows

$$\dot{\hat{\boldsymbol{x}}}_w = \boldsymbol{A}_\omega \hat{\boldsymbol{x}}_w + \boldsymbol{K}_1 \tilde{\boldsymbol{y}} \tag{94a}$$

$$\dot{\hat{\boldsymbol{\eta}}} = \boldsymbol{R}(\psi)\hat{\boldsymbol{\nu}} + \boldsymbol{K}_2 \tilde{\boldsymbol{y}} \tag{94b}$$

$$\dot{\hat{\boldsymbol{b}}} = -\boldsymbol{T}^{-1}\hat{\boldsymbol{b}} + \boldsymbol{K}_3 \tilde{\boldsymbol{y}} \tag{94c}$$

$$\boldsymbol{M}\dot{\hat{\boldsymbol{\nu}}} = -\boldsymbol{D}\hat{\boldsymbol{\nu}} + \boldsymbol{R}^T(\psi)\hat{\boldsymbol{b}} + \boldsymbol{\tau} + \boldsymbol{R}^T(\psi)\boldsymbol{K}_4 \tilde{\boldsymbol{y}} \tag{94d}$$

$$\hat{\boldsymbol{y}} = \hat{\boldsymbol{\eta}} + \boldsymbol{C}_\omega \hat{\boldsymbol{x}}_w \tag{94e}$$

where the gains are

$$\boldsymbol{K}_1 = [diag([k_1, k_2, k_3]), diag([k_4, k_5, k_6])]^\top \tag{95}$$

$$\boldsymbol{K}_2 = diag([k_7, k_8, k_9]) \tag{96}$$

$$\boldsymbol{K}_3 = diag([k_{10}, k_{11}, k_{12}]) \tag{97}$$

$$\boldsymbol{K}_4 = diag([k_{13}, k_{14}, k_{15}]). \tag{98}$$

The gains are tuned using the optimization algorithm *fminsearch* under the constraints of the general tuning rules described by Fossen (2011)

$$k_i = -2(\zeta_{ni} - \lambda_i)\frac{\omega_{ci}}{\omega_{oi}} \tag{99a}$$

$$k_{3+i} = 2(\zeta_{ni} - \lambda_i)\omega_{oi} \tag{99b}$$

$$k_{6+i} = \omega_{ci} \tag{99c}$$

$$k_{9+i} \gg \frac{k_{12+i}}{T_{b,i}} \tag{99d}$$

where $i = 1, 2, 3$, $\lambda$ is the relative damping of the wave spectrum, $\zeta$ is a damping parameter, $\omega_0$ is the peak frequency of the wave spectrum, $\omega_c$ is the filter cut-off frequency and $T_b$ is the bias time constant.

## 5.4 Thruster allocation

For simplicity, the thruster orientations are fixed and set to $\boldsymbol{\alpha} = [\pi, \frac{\pi}{4}, \frac{-\pi}{4}, 0, \frac{5\pi}{4}, \frac{3\pi}{4}]$. The transformation from desired forces and moments to low level actuator setpoints $\boldsymbol{u}$ is adopted from

Lyngstadaas (2018)

$$\boldsymbol{u} = \boldsymbol{K}_T^{-1}\boldsymbol{T}^{\dagger}(\boldsymbol{\alpha})\boldsymbol{\tau}_{sat} \tag{100}$$

$$\boldsymbol{T}(\boldsymbol{\alpha}) = \begin{bmatrix} cos(\alpha_1) & cos(\alpha_2) & cos(\alpha_3) & cos(\alpha_4) & cos(\alpha_5) & cos(\alpha_6) \\ sin(\alpha_1) & sin(\alpha_2) & sin(\alpha_3) & sin(\alpha_4) & sin(\alpha_5) & sin(\alpha_6) \\ \phi(\alpha_1) & \phi(\alpha_2) & \phi(\alpha_3) & \phi(\alpha_4) & \phi(\alpha_5) & \phi(\alpha_6) \end{bmatrix} \tag{101}$$

where $\phi(\alpha_i) = L_i \cos(\beta_i)\sin(\alpha_i)$, $L_i = \sqrt{L_{i,x}^2 + L_{i,y}^2}$ and $\beta_i = tan(\frac{L_{i,x}}{L_{i,y}})$. $\boldsymbol{T}^{\dagger}(\alpha)$ is the pseudo inverse of $\boldsymbol{T}(\alpha)$. The forces and moments are saturated according to Løvås (2019), which is the recommended adaptation of the saturation limits suggested by Lyngstadaas (2018)

$$\boldsymbol{\tau}_{max} = \begin{bmatrix} 3N \\ 3N \\ 3Nm \end{bmatrix} \dot{\boldsymbol{\tau}}_{max} = \begin{bmatrix} 2.88N/s \\ 1.60N/s \\ 1.36Nm/s \end{bmatrix} \tag{102}$$

# 6 Convolutional neural networks

An artificial neural network (ANN) is, as briefly mentioned in section 2.4, a function approximator inspired by the human brain. It is a parametric model, trained by supervised learning on labeled data sampled from the unknown target function. The layered structure of artificial neurons mimics that of the brain in how each neuron receives an input, processes it, and signals all neurons connected to it. More specifically, a single artificial neuron, seen in figure 11 a), receives weighted outputs from neurons in the previous layer. The weighted outputs are added a bias, summed, and passed through an activation function which has a non-negative derivative - described in more detail in section 6.3. The biases are neglected here for simplicity. Feed-forward neural networks (FFNN) are the most basic type of artificial neural network where the information is passed in only one direction, from input to output. Figure 11 b) show an FFNN with two hidden layers. All layers are *dense* or *fully connected*, meaning each neuron has an incoming edge from all neurons in the previous layer, as well as an outgoing edge to all neurons in the next layer.

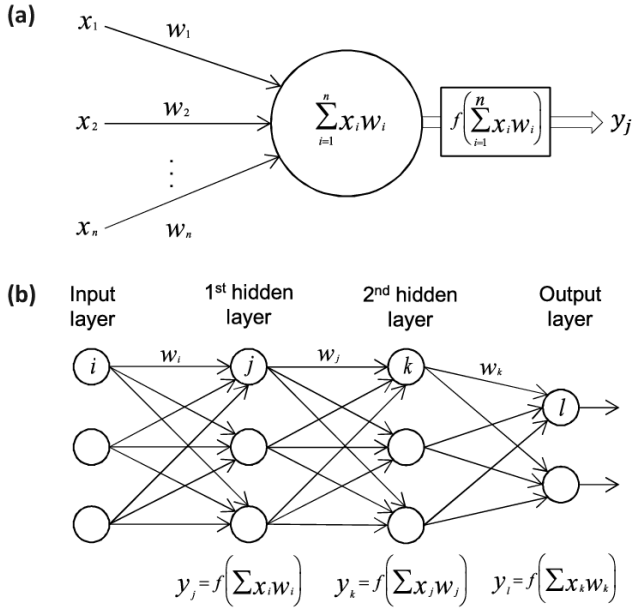



**Figure 11:** Layout of a deep feed-forward neural network (Vieira et al. (2017))

The convolutional neural network (CNN) is a type of ANN, however, in addition to dense layers, the CNN typically consists of blocks of convolutional and pooling layers. This NN architecture was created and is mostly used for image classification and object detection due to the convolutional filter's ability to recognize features regardless of temporal or spatial location. Figure 12 illustrate how a CNN can be applied to the problem of image classification. The input layer is a 2d matrix of pixel values, which is filtered and downsampled several times through convolution and pooling. Finally, the data cube of information is flattened and used as input to a regular feed-forward neural network. This enables the model to learn nonlinear combinations of the features outputted from the last convolution block. The last layer utilizes the softmax activation function that outputs a

probability distribution over all the possible targets.

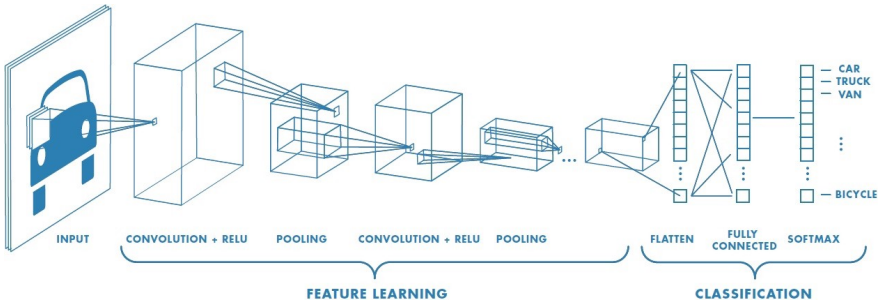



**Figure 12:** Layout of a convolutional neural network (Prabhu (2018))

## 6.1 Convolutional layer

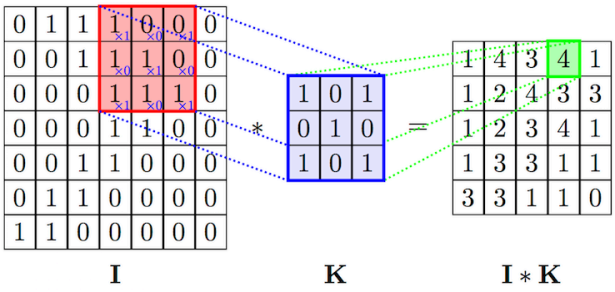
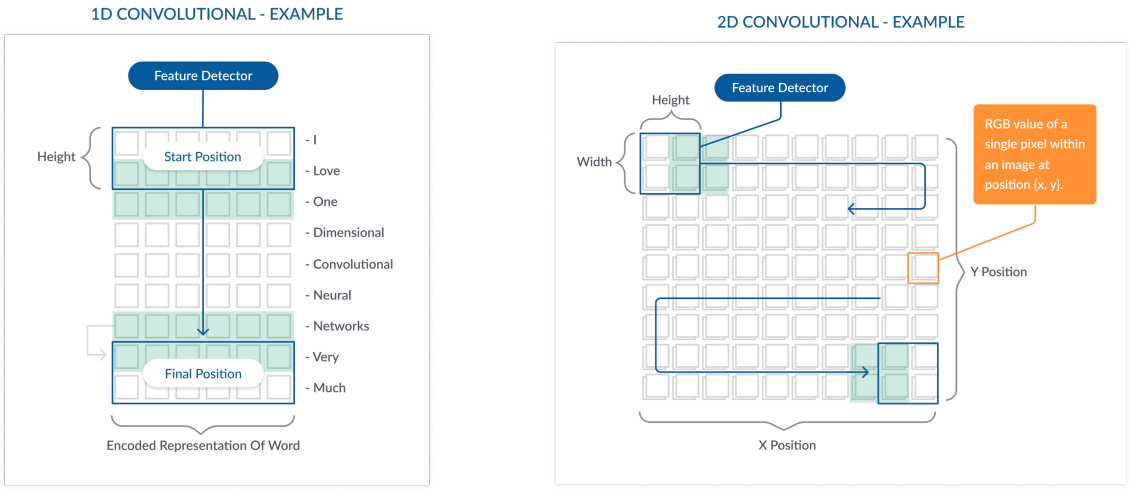



**Figure 13:** The filters in a convolutional layer (Vo (2018))

A number of different filters/kernels are applied to the input matrix, as seen in figure 15. In the example, a filter of size $3 \times 3$ is "slid" across the input matrix from upper left to lower right, row by row, resulting in a size reduction. Different filters extract different features such as edge detection, blurring, and sharpening. In addition to number of filters and the filter dimension, the stride is an important parameter of the convolutional layer. The stride can be explained as the speed of the filter - the step size between each convolutional operation. Convolutional layers can also be applied to one-dimensional data such as text or time signals, as seen in figure 14a. The convolutional layers are often used after each other in blocks to capture low-level details - however, at the cost of computational complexity.




**(a)** 1D convolutional filter

**(b)** 2D convolutional filter

**Figure 14:** The two different convolutional filters with size 2 and $2 \times 2$, and stride 1(Missinglink.ai (2019))

## 6.2 Pooling layer

To decrease the computational power needed and to extract only the most dominant features from a convolution layer or block, a pooling layer is used. A pooling layer traverses the input matrix in a similar manner as the convolutional filters, only extracting max, min, or average value within a set area. The important parameters for the pooling layer is the size of each kernel, as well as whether to extract min, max, or average values. Max pooling intuitively also works as a noise suppressant, as the high-frequency variations will be lost through the layer. This is one of the reasons that a CNN needs less data pre-processing than other ML methods.

## 6.3 Activation functions

The choice of activation function is crucial when building deep neural networks. It is highly influential for the model's ability to learn, convergence speed, and accuracy. The activation function is illustrated in figure 11 a) as a function of the sum of the weighted outputs from the previous layer

$$f\left(\sum_{i=1}^{n} x_i w_i\right). \tag{103}$$

One such activation function is attached to each neuron in the network, and based on the sum of the weighted outputs from the previous layer, it decides to which degree the neuron should be activated. The activation functions also normalize the output values to a range between 0 and 1, or -1 to 1. It can be as simple as a step function, making each neuron either active or inactive. Another simple type is some linear activation function, $f(x) = ax$. The problem with linear activation functions, is that the backpropagation algorithm used to train the model weights depends on the

derivative - which is constant for linear functions. Another problem is the inability of the model to fit a nonlinear function. Regardless of the number of hidden layers, a linear combination of linear functions is still linear.

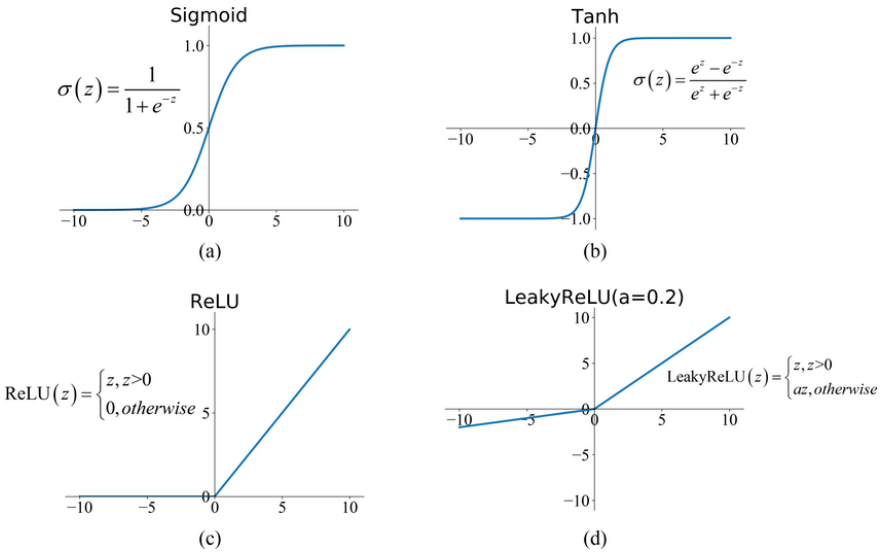



**Figure 15:** Most common activation functions used in deep learning (Feng et al. (2019))

Figure 15 shows the most common nonlinear activation functions. In these cases, the derivative function exists and can be related to the input. This enables the gradient method to use knowledge of the derivative to adjust the weights in the right direction. The other outcome of nonlinear activation functions is that a model can theoretically fit any target function - the more hidden layers and nodes, the more complex function can be fitted. The downside is the increased computational complexity. When training, the model calculates the output of each neuron's activation function for each iteration - possibly millions of calculations. The function therefore has to be as computationally efficient as possible.

The sigmoid and tanh functions both have smooth derivatives and normalizes the output, but are more computationally expensive. Another disadvantage is the *vanishing gradient problem*. When the input is very large or small, the derivative becomes vanishingly small - again causing the weight change during training to be similarly small. This often causes the model to stop learning or learn extremely slow. The largest difference between the two is that tanh is zero centered, making it better suited for inputs with equally positive and negative numbers.

The rectified linear unit (ReLU) activation function has the advantage of being less computationally expensive. On the other hand, negative values result in the derivative being zero and the weights will not change during training. The leaky ReLU function is a solution to this problem, as it outputs $az$ $(0 < a < 1)$ for negative input values. This does, however, come at the cost of consistency for negative input values.

There are two different types of tasks a neural network can perform; classification and regression.

The first outputs the class affiliation of the input. An example is figure 12, where the CNN outputs whether the input image is a car, van, truck, or bicycle. This kind of task calls for the softmax activation function, which outputs a probability distribution over the possible classes. Softmax is only used in output layers. Regression, on the other hand, intends to output a real number close to the target value. Linear activation functions, or ReLU if the output should be positive, are therefore most common in the output layer of a regression model. Classification outputs discrete or categorical variables, while regression outputs continuous variables.

## 6.4 Training

The next challenge is to train the network. More specifically, to tune the weights and biases of the network, henceforth referred to as the parameters. A central part of training is deciding on a loss/cost function that is based on the error between output value and target/true value. The loss function enumerates the current performance of the model, but its derivative with respect to each parameter is also actively utilized for the algorithm to bump the parameters in the direction of decreased loss. The algorithm for training a neural network is hence twofold; calculating the gradients and using them to adjust the parameters in the right direction.

Perhaps the most important principle when training a supervised machine learning model is splitting the data set into training and test set. The goal is for the model to learn an unknown function by seeing labeled samples. There is hence of paramount importance that a part of the data set is put aside to get a post-training, final indication of how the model has actually fitted the unknown function. Limited by computational memory, the model is trained on *batches*, subsamples of the training data, for each parameter update. An ANN often trains faster with batches, due to the more frequent parameter update - one for each batch. The downside is that if the batch size is too small, the cases in the batch are not a good representation of the training set, and the estimated gradients become inaccurate. Another critical factor is that the training set is shuffled so that each *batch*, to some degree, represents the full training set. One parameter update for all the *batches* in the training set is called an *epoch* - one training run on the entire training set.

### 6.4.1 Loss function

As mentioned, training a neural network is basically the task of finding the combinations of parameters that minimize the loss function. The function has to capture as many aspects of the model as possible and map them to a single real number - the lower the number, the better the model. In the case of neural networks, the function is based on the error between the model's output values and the target values. Minimizing the loss function hence implies minimizing the error. The error refers to the mean error of a batch of training cases. For a regression model, where the target is one or more continuous variables, the most common loss function is the mean squared error

$$L(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \frac{1}{n}\sum_{i=1}^{n}(X_i - \hat{X}_i)^2 \tag{104}$$

where $n$ is the batch size, $X_i$ is the target value, and $\hat{X}_i$ is the estimated value. Other loss functions for regression problems are mean absolute error and mean absolute percentage error. The binary cross-entropy and categorical cross-entropy are the most popular for binary and multi-class classification problems, respectively. The output distribution of the model is compared to the target values where only the true class is 1 and the rest is 0 - represented as a one-hot encoded vector. Categorical cross-entropy is mathematically formulated as

$$L(\boldsymbol{X}, \hat{\boldsymbol{X}}) = -\sum_{j=0}^{m}\sum_{i=0}^{n}(X_{ij}\log(\hat{X}_{ij})). \tag{105}$$

### 6.4.2 Backpropagation and gradient methods

The term backpropagation refers to the algorithm that calculates the gradient of the loss function. Gradients are derived with respect to each individual weight $\frac{\delta L}{\delta w_i}$ and each individual bias $\frac{\delta L}{\delta b_i}$. The algorithm was initially published in the 1970s, however, most of the attention is credited Rumelhart et al. (1986). The algorithm's main contribution is its efficiency, enabling the calculation of hundreds of thousands of gradients within reasonable time. The layered structure enables the calculation of the gradients analytically, instead of numerical approximation. Once the gradients in the last layer are found, they can be propagated backwards through the network - hence backpropagation.

Subsequently, some gradient method is needed to update the parameters based on the calculated gradients. A gradient method is simply an optimization algorithm that minimizes some function $f(x)$ by using its gradient for search direction - gradient descent. Many existing algorithms are applicable for training a neural network, the differences are mostly how they balance convergence speed versus robustness. The single most important parameter for the optimizer is the *learning rate* - the step size taken for each parameter update.

Batch gradient descent updates the parameters based on the cost for the entire data set. This can be time-consuming, which resulted in an improved algorithm; Stochastic gradient descent (SGD). SGD does parameter updates for each training example, drastically reducing convergence time. On the other hand, the frequent parameter update based on a small part of the training data causes large fluctuations in the loss. Overshooting and oscillations is also a common problem with SGD. The challenges with both approaches are tuning the learning rate, and that the same learning rate is applied to all parameters, which is often not optimal.

Momentum is a method that takes on some of the challenges with SGD. Inspired by physics and potential energy, the loss functions are viewed as terrain with tops and valleys. The steeper the terrain, the faster the parameter update - not unlike a ball rolling down a hill. The algorithm

considers previous gradients and has a term limiting the size of the parameter change - effectively working the same way as air resistance. This limits the overshoot and decreases oscillations by better directional guidance.

In later years algorithms have been published that take care of the difficult choice of learning rate, as well as using adapted learning rates for each parameter. They still need hyperparameter consideration but are more forgiving towards imprecise tuning. Adagrad, published by Duchi et al. (2011), is an example of such. If a feature occurs often, the associated parameters are given a lower learning rate than if the features occur more seldom. A potential challenge is decreasing learning rates as the model train, leading to diminishing learning rates and stagnation. As opposed to Adagrad, RMSprop only consider a moving average of limited past gradients, avoiding the diminishing learning rates. Adam, an algorithm inspired by both Momentum and RMSprop, is considered to be one of the best gradient descent optimization algorithms in general - fast and good performance on most function surfaces. It uses adapted learning rates as RMSprops, but unlike Momentum, the "ball" has friction against the surface.

### 6.4.3 Overfitting

When fitting a neural network to an underlying function, the most common cause of poor performance is overfitting the training data. Overfitting is when the model learns the training data too well, and lose the ability to generalize towards new, unseen data. The model learns the irrelevant details and noise in the training data, not present in the test data. The end goal in supervised learning is not to learn the outputs of the training data, but to learn the underlying function based on an incomplete sampling.

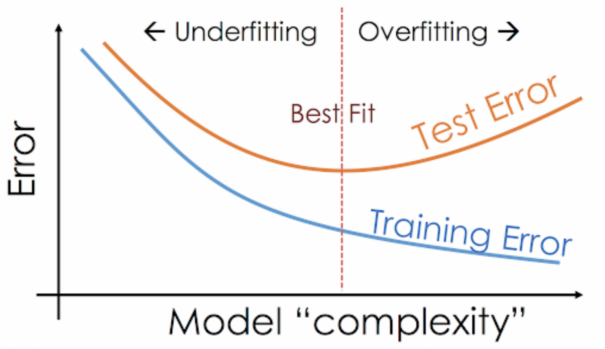



**Figure 16:** Development of training error versus test error during training of a neural network. Initially the error is large on both data sets. As the model parameters are adjusted the error decreases until the error on the test set start increasing. From that point on further fitting to the training data is at the cost of the models ability to generalize (Saxena (2020))

Using what is called a validation set often provides a good indication of when the model is at *best fit*. A subsample of the training set is set aside, often around 10-20 %. The model is trained on the remaining cases and tested on the validation set after each epoch. After the model is fully trained,

the the training set's loss can be plotted against the loss of the validation set for each epoch of the training, see figure 17.

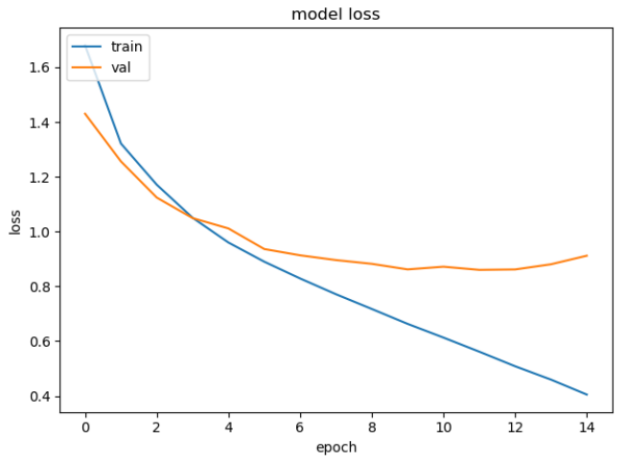



**Figure 17:** Plot of training and validation loss for overfit model (Wang (2018)).

The validation set can be the same for the entire training, or one can do $x$-fold cross-validation - splitting the training set into $x$ parts. The model is trained $x$ times for each epoch, each time on a new combination of $x-1$ parts. The final part is used for validation. The validation loss for the epoch is then the average validation loss for each of the $x$ training iterations.

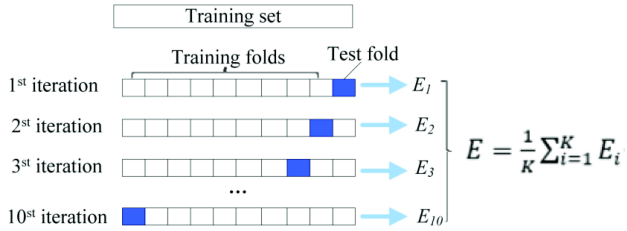



**Figure 18:** 10 fold cross validation (Niu et al. (2018))

Dropout is another technique to avoid overfitting. It was first published in Hinton et al. (2012), introducing *... a regularization method that approximates training a large number of neural networks with different architectures in parallel* (Browlee (2018)). Dropout ignores a certain number of outputs from a layer. Which outputs being ignored changes probabilistically for each iteration. This introduces noise to the model, preventing overfit. Since different nodes are active each time information is passed forward through a layer with dropout, several nodes are forces to take responsibility for the same features - increasing robustness. The most important hyperparameter for a dropout layer is the *rate*, setting the probability of each node being ignored - usually around .5.

A practical way to make sure the best model, meaning the architecture and the parameters, is kept is using validation-based early stopping. Many models start overfitting after some number

of epochs, hence the solution is to always save the model with the smallest validation loss. The *patience* is often used as a hyperparameter indicating how many epochs the model is allowed to train without seeing a reduction in validation loss, preventing a lot of time spent training a model that has surpassed its *best fit*.

# 7 Sea state estimation based on simulated IMU measurements

## 7.1 Data generation

For generating a large enough and realistic data set, the simulation model is run in DP operation for 40 seconds in each sea state - with long-crested irregular sea. The data set is equally distributed over the 12 wave directions. The choice of significant wave height and peak period is chosen first by random choice of sea state, and further by a random choice of the two parameters from within the domains defined by the sea state. 2000 simulations are run for each wave direction, resulting in a data set consisting of 24000 cases. Each case contains linear acceleration in heave and angular velocities in roll and pitch, from four different IMUs.

The specific have height is intuitively positively correlated with the accelerations, and negatively correlated with the peak period. The exceptions are introduced when the peak period of the waves is close to the natural period in either of the three degrees of freedom. In case of induced resonance, the accelerations will diverge from the aforementioned pattern. The roll motions are anti-symmetric about the x-axis, while the heave motions are symmetric. This is the main pattern utilized in differentiating between port and starboard waves. Similarly, the relation between pitch motion and heave motion and their temporal difference contains information for differentiating between head and following sea.

## 7.2 Pre-processing

Initially, the measurements from the four IMUs are used to find the accelerations in CO, using the derived equations in section 4.5.2. The data set is then split up into training set and test set, with the latter consisting of 10 % of the original data set. From the training set, a validation set of 20% is further extracted.

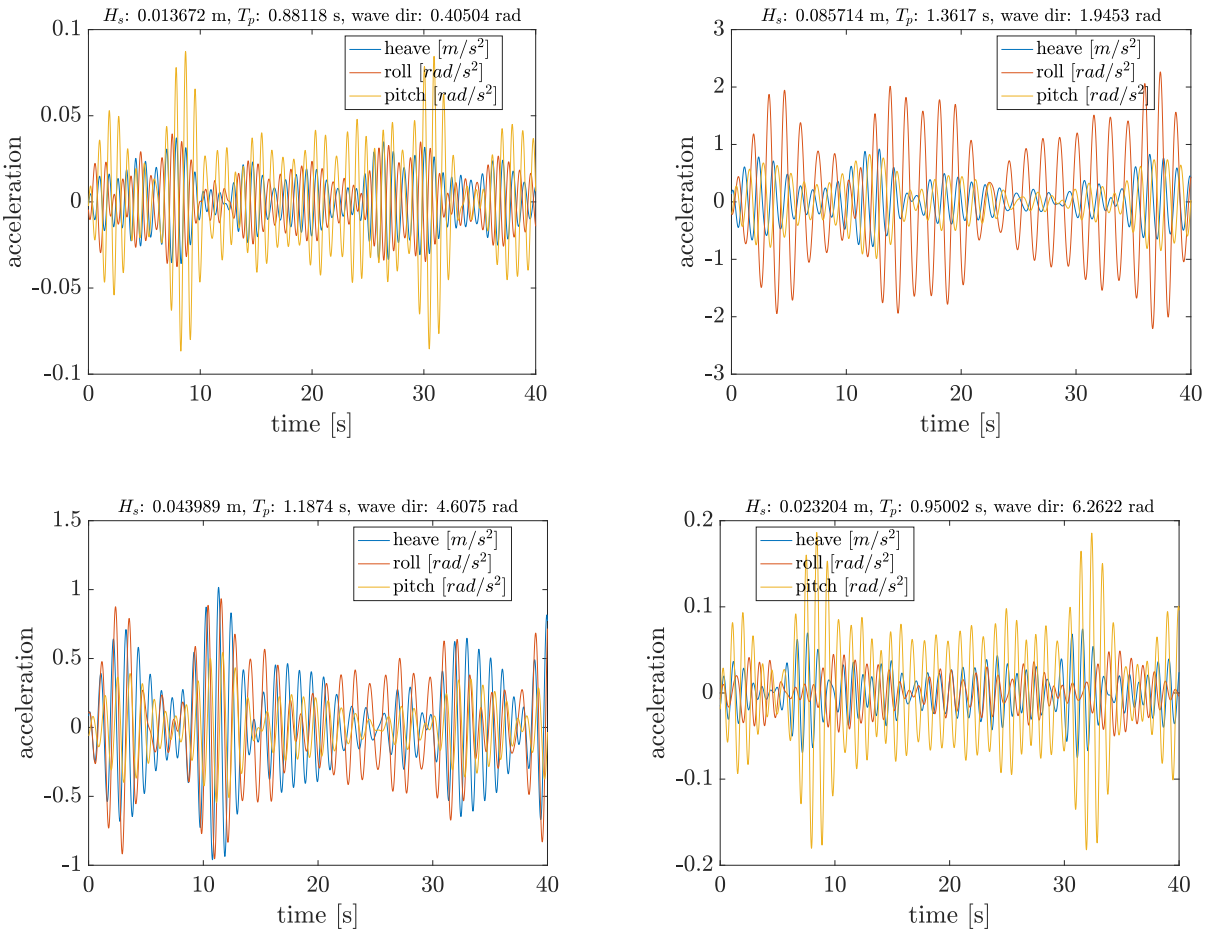



**Figure 19:** 3 DOF acceleration data from different sea states in irregular sea

Since the wave directions are defined in sectors, and hence into 12 classes, one-hot encoding is necessary for the CNN not to be biased towards any one class. One-hot encoding does *binarization* on the wave direction label of each case. The result is that the wave direction is represented by 12 binary values, where only the n'th value is 1 in the case of sector n being the true wave direction. The output layer of the wave direction model therefore need to have 12 nodes. The main reason for one-hot encoding is that a data labeling according to the wave direction sector number, 1-12, would result in bias towards higher sector numbers. In addition, the first sector is equally close to sectors 2 and 12, however the difference in activation in the output node between sector 2 and 12 would be close to the difference between min and max activation. Illustration of the process seen in table 4.

**Table 4:** Illustration of one-hot encoding

| case number | case label |
|---|---|
| 1 | 7 |
| 2 | 3 |
| 3 | 1 |
| 4 | 3 |
| 5 | 2 |

$\Rightarrow$

| case number \ case label | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 7.3 Hyperparameter tuning

Any parameter that is initialized in the CNN before training, and hence can not be estimated from the data, is a hyperparameter. The choice of each hyperparameter highly influences the performance of the model, and needs to be carefully chosen. To determine which hyperparameters bring out the best performance, the loss on the test set after 60 epochs are compared. Grid search is a common approach where each combination of values for two or more hyperparameters is tested and compared with regards to some external data set not used during training. Another important factor is setting the *random seed* when splitting the data set into training and test set. When using a set *random seed*, the same cases are chosen as test set when splitting the data set. Resulting in all models being compared on the same test set.

Early stopping is naturally utilized to avoid overfitting. The *optimizer* has been chosen to be Adam for all three models, due to its speed and adaptive learning rates. ReLU is chosen as *activation function* in all hidden layers, for all three models. The reason is the simplicity of the function, and therefore speed, while still being non-linear enough for the backpropagation algorithm. An increased *stride* can decrease the training time, however the training time is acceptable with a *stride* of 1. This ensures no features or patterns in the data are missed in the convolution layers.

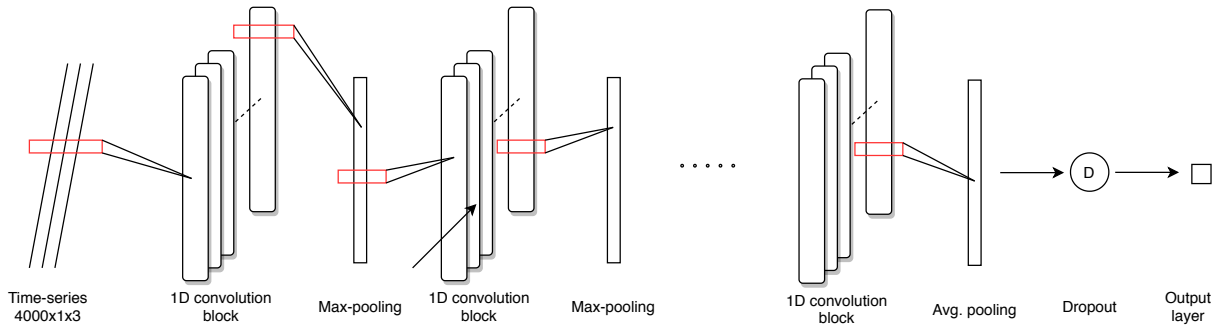



**Figure 20:** General architecture of the regression CNN models before grid search. The red rectangles represent the filter/kernel, and the lines represent how one filter extraction is represented in the next layer.

The following three models are organized in parallel.

### 7.3.1 Specific wave height model

The specific wave height model is a regression model with only positive outputs, so the ReLU activation function is used in the output layer. Manual testing is performed for kernel size and number of kernels in the convolution layers, as well as size of the pooling filters. A grid search is then performed for the number of layers per convolution block, and number of such blocks in the model, see figure 20. The static hyperparameters for each convolutional layer during the search are

- Number of filters/kernels: 254
- Size of filter/kernel: 10
- Stride: 1

Pooling layers are added after each convolution block, max-pooling with *filter/kernel size* 3 after all except the last block, where the global average pooling layer is added. In addition there is a dropout layer with *rate* at 0.5 before the output layer. With the aforementioned static hyperparameters, the result of the grid search is seen in table 5.

**Table 5:** Grid search results for different number of convolutional blocks and layers per block.

| Layers per block | No. blocks | Mean squared error | Mean percentage error | Standard deviation percentage |
|---|---|---|---|---|
| 2 | 2 | $5.159\times10^{-6}$ | 4.171 | 3.963 |
| 3 | 2 | $3.792\times10^{-6}$ | 3.826 | 3.626 |
| 4 | 2 | $3.865\times10^{-6}$ | 3.944 | 4.106 |
| 1 | 3 | $4.775\times10^{-6}$ | 3.990 | 3.613 |
| 2 | 3 | $2.938\times10^{-6}$ | 3.460 | 3.685 |
| 3 | 3 | $3.134\times10^{-6}$ | 3.242 | 3.484 |
| 4 | 3 | $3.387\times10^{-6}$ | 3.535 | 3.799 |
| 1 | 4 | $3.155\times10^{-6}$ | 3.401 | 3.524 |
| 2 | 4 | $2.258\times10^{-6}$ | 3.149 | 3.544 |
| 3 | 4 | $2.189\times10^{-6}$ | 3.133 | 3.177 |
| 4 | 4 | $3.427\times10^{-6}$ | 3.773 | 3.613 |

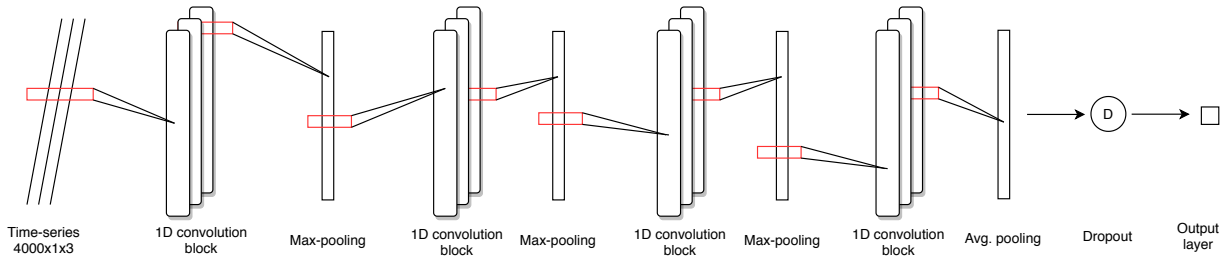



**Figure 21:** Architecture of the wave height model after grid search. Three convolutional layers per block and four blocks achieved the best accuracy.

### 7.3.2 Peak period model

The static hyperparameters for each convolutional layer during the search are

- Number of filters/kernels: 254
- Size of filter/kernel: 10
- Stride: 1

For finding the best number of layers in each convolution block, and the number of blocks, similar grid search is performed to the peak period model. The architecture with five layers per block and three blocks is chosen.

**Table 6:** Grid search results for different number of convolutional blocks and layers per block.

| Layers per block | No. blocks | Mean squared error | Mean percentage error | Standard deviation percentage |
|---|---|---|---|---|
| 1 | 2 | 0.03910 | 2.756 | 1.987 |
| 2 | 2 | 0.00771 | 1.145 | 0.886 |
| 3 | 2 | 0.01620 | 1.680 | 1.346 |
| 4 | 2 | 0.00392 | 0.877 | 0.657 |
| 1 | 3 | 0.00834 | 1.176 | 0.985 |
| 2 | 3 | 0.01377 | 1.564 | 1.199 |
| 3 | 3 | 0.00446 | 0.934 | 0.747 |
| 4 | 3 | 0.01014 | 1.412 | 1.002 |
| 1 | 4 | 0.00484 | 0.955 | 0.679 |
| 2 | 4 | 0.00434 | 0.875 | 0.686 |
| 3 | 4 | 0.00686 | 1.111 | 0.932 |
| 1 | 5 | 0.00618 | 1.052 | 0.787 |
| 2 | 5 | 0.00539 | 1.052 | 0.765 |
| 3 | 5 | 0.00391 | 0.896 | 0.693 |

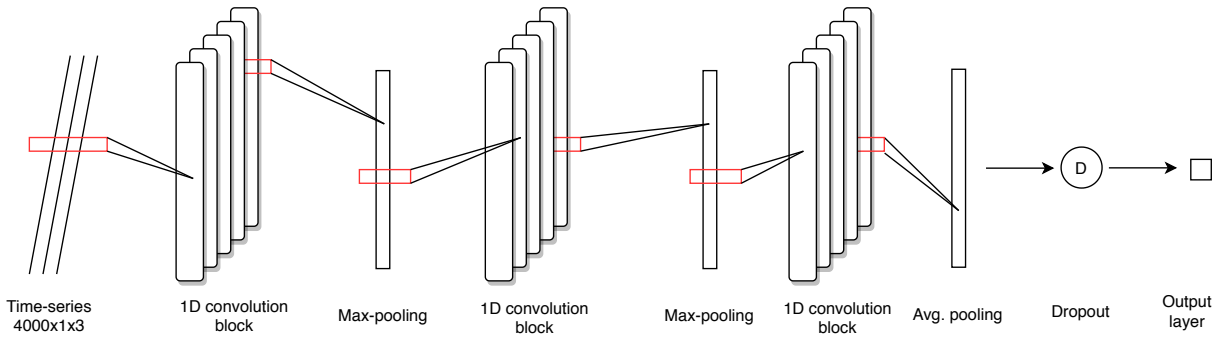



**Figure 22:** Architecture of the peak period model after grid search.

### 7.3.3 Wave direction model

The same base architecture is used for the wave direction model, however since it is a classification model, the softmax activation function is used in the output layer. For finding the best number of layers in each convolution block, and the number of blocks, the same grid search is performed. The architecture with three layers per block and two blocks is chosen.

**Table 7:** Grid search results for different number of convolutional blocks and layers per block.

| Layers per block | No. blocks | Accuracy on test set |
|---|---|---|
| 1 | 2 | 99.87% |
| 2 | 2 | 99.92% |
| 3 | 2 | 100% |

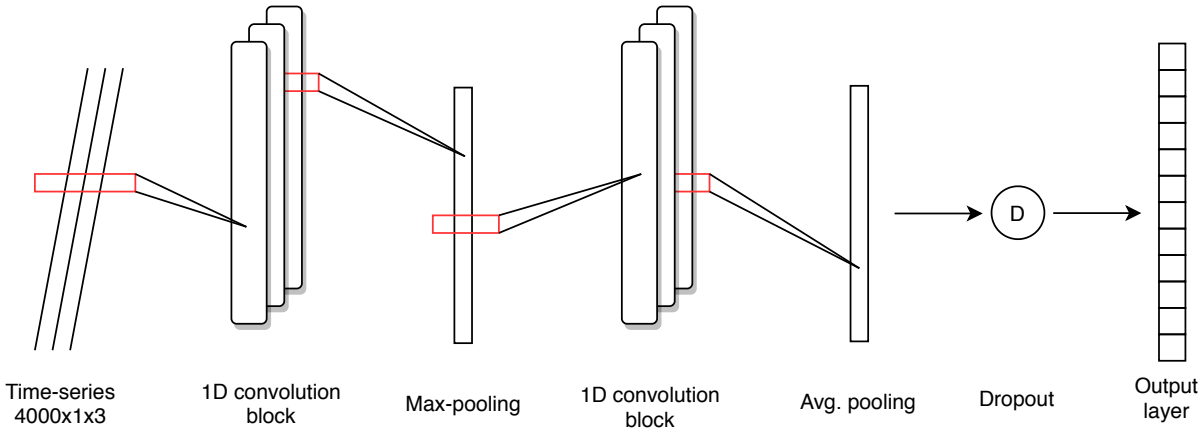



**Figure 23:** Architecture of the wave direction model after grid search.

# 8 Instrumentation and experimental set-up

The Marine Cybernetics Laboratory contains a wave basin with advanced instrumentation package and a towing carriage. The dimensions of the wave basin is $40 \times 6.45$ m with a constant depth of 1.5 m, see figure 24. The lab has a fleet of model ships which are primarily used for experimental testing of marine control systems and hydrodynamic testing. For more information see the Marine Cybernetics Laboratory Handbook (2017).

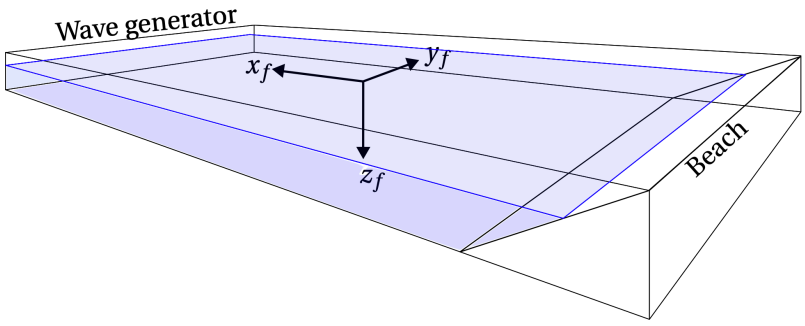



**Figure 24:** Basin in MCLab (Udjus (2017))

## 8.1 CS Arctic Drillship

In 2016, John Bjørnø built a 1:90 model of the Cat I Arctic Drillship, a ship built by Inocean for Equinor in 2013. The model was originally built for his research on thruster-assisted position mooring. It is equipped with six azimuth thrusters, where the angles are controlled through a servo motor, while the RPMs are set by a speed controller. The main dimensions of the vessel is presented in table 8. The large size makes it desirable for mounting additional sensors.

The control system is ran on a National Instruments CompactRIO-9024 (cRIO), which is an embedded real-time controller. It reads the Qualisys Track Manager (QTM) pose data, and sets the low-level set points for the servo motors and the speed controllers. Any Simulink code compiled to C-language can be exported to the cRIO. The playstation controller sends the user commands via bluetooth to the Raspberry Pi, which further sends the signals to the cRIO via ethernet cable. From figure 25 the two ways of vessel control is shown; laptop and playstation controller.

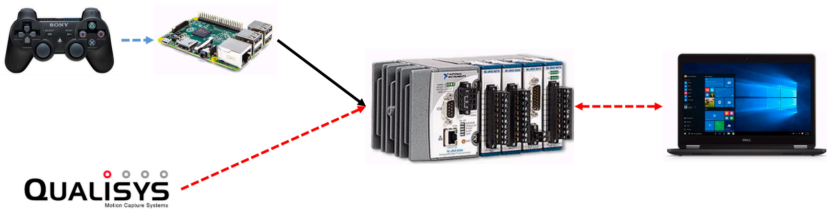



**Figure 25:** Communication diagram Marine Cybernetics Laboratory Handbook (2017)

| Description | Data |
|---|---|
| Length over all ($L_{oa}$) | 2.578 [m] |
| Breadth (B) | 0.440 [m] |
| Depth moulded (D) | 0.211 [m] |
| Draft (T) | 0.133 [m] |

**Table 8:** Scaled data for the CSAD (Bjørnø (2016))

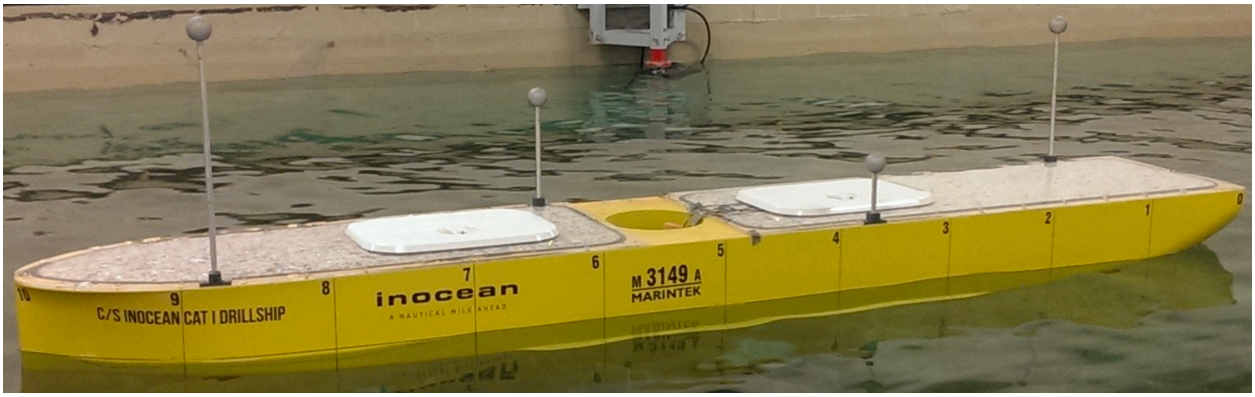



**Figure 26:** the CSAD, a 1:90 model ship of the Inocean Cat I Arctic Drillship (Bjørnø (2016))

## 8.2 Qualisys Motion Capture System

Qualisys Motion Capture System provides pose data of surface vessels for all 6 DOFs. The system consist of three high speed infrared cameras set up along one of the short sides of the lab. The cameras register the reflections of the four antennas, see figure 28. The Qualisys Track Manager (QTM) software installed on one of the lab computers calculates and broadcasts the vessel pose over the wireless network. The signal sampling is at 50Hz, and with millimeter precision. The QTM is meant to emulate a full scale global navigation satellite system (GNSS).

## 8.3 Wave generator

The lab is equipped with a wave generator that is 6 meter wide. An Active Wave Absorption Control System is in place to minimize the reflections of the waves from the basin. Available spectrums are first order Stoke, JONSWAP, Pierson-Moskowitz, Bretschneider, ISSC and ITTC. Due to the single paddle, the generator is naturally limited to long-crested waves. The system has the following capabilities

| | Significant wave height [m] | Peak period [s] |
|---|---|---|
| Regular waves | $H < 0.25$ | 0.3–3.0 |
| Irregular waves | $H_s < 0.15$ | 0.6–1.5 |

**Table 9:** The wave maker capabilities (Marine Cybernetics Laboratory Handbook (2017))

## 8.4 IMU

The four IMUs mounted on the vessel are of the type ADIS16364 by Analog Devices. Each IMU contains a triaxis gyroscope and triaxis accelerometer, with built-in compensation for bias, alignment and sensitivity. Each of the four IMUs are connected to a microprocessor, of the type Arduino Leonardo ETH, which enables sampling of the IMU data and communication through ethernet to the cRIO. The data is sampled at a rate of 20 Hz. Synchronization of the sampling from the four IMUs is handled by one master IMU sending an interrupt signal to the three others- initiating sampling simultaneously.

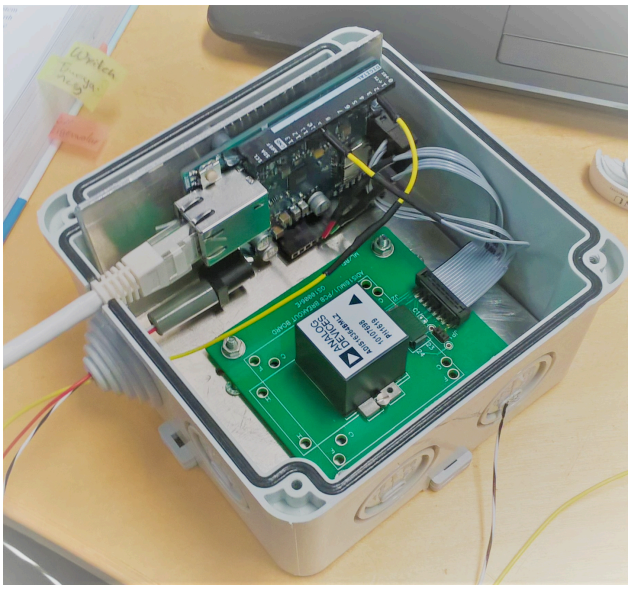


**Figure 27:** The IMU and the Arduino Leonardo ETH microprocessor (Udjus (2017))

For full technical documentation of the IMUs see Analog Devices, and for more description of the sampling, processing, and communication of the signal see Udjus (2017).

## 8.5 Set-up, data generation, and pre-processing

The DP control system was implemented and tested using a human-in-the-loop (HIL) procedure, however not tuned for the actual vessel. The HIL procedure was performed using the same hardware as installed on the vessel, but with a simulation model installed on the cRIO, generating updated positional data based on low-level actuator commands.

Generation of the data from the CSAD in actual DP operation in the basin was initially planned. However, due to the pandemic and closing of the lab, the data from what was originally intended to be an initial test of the sensors had to be used. The set-up in lab is seen in figure 28, where the CSAD is loosely strapped for limited movement in the horizontal plane. This set-up enables motion data generation for a single wave direction with only small displacements in vessel heading. Several different combinations of wave height, frequency, and direction for long-crested regular waves were used in the experiment, however, only some of the motion data is applicable in this thesis due to the choice of sea state domains, described in table 3.

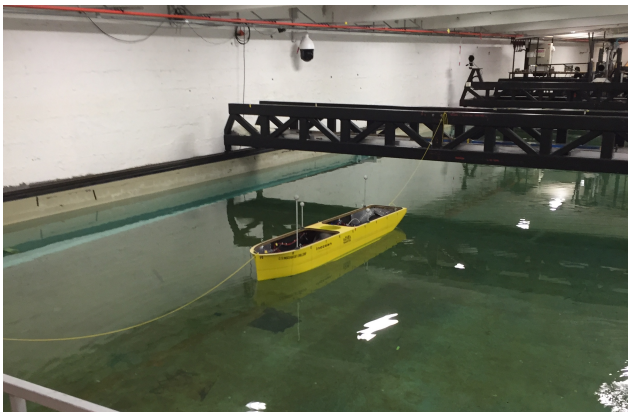


**Figure 28:** CSAD in the basin, strapped for wave direction data

The IMU sensor configuration is visualized in figure 29, and each IMUs pose and location in $\{b\}$ is presented in table 10. For obvious reasons, measuring the precise location of the IMUs was difficult, likely introducing measurement errors. The measurements from the accelerometer had to be rotated in order for positive directions to be according to the right hand rule. In addition the measurements were scaled from $mg$ to $m/s^2$. Subsequently, all IMU measurements were rotated to align with $\{b\}$ using the rotation matrix described in section 4.1.3 and the pose of the IMUs in table 10. Low-pass filtering was applied to the measurements, and the bias removed using the mean measurement values. The angular accelerations were derived from the filtered gyroscope measurements before IMU sensor fusion was performed.

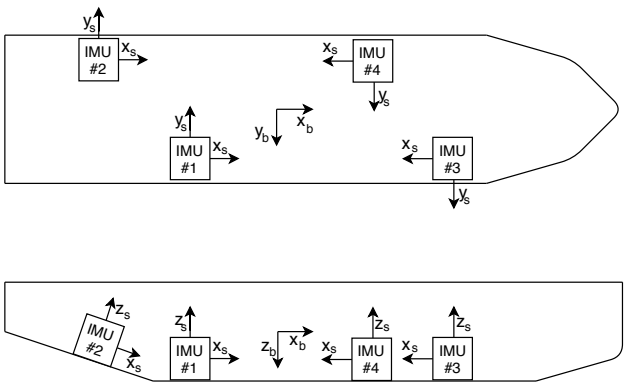



**Figure 29:** The set up of the four IMUs on CSAD

| IMU no. | x [mm] | y [mm] | z [mm] |
|---|---|---|---|
| 1 | -244 | 184 | 82 |
| 2 | -950 | -130 | 0 |
| 3 | 740 | 130 | 82 |
| 4 | 450 | -160 | 82 |

| IMU no. | $\phi$ [deg] | $\theta$ [deg] | $\psi$ [deg] |
|---|---|---|---|
| 1 | 180 | 0 | 0 |
| 2 | 180 | -17.5 | 6.5 |
| 3 | 180 | 0 | 180 |
| 4 | 180 | 0 | 180 |

**Table 10:** Location and pose of each IMU in $\{b\}$

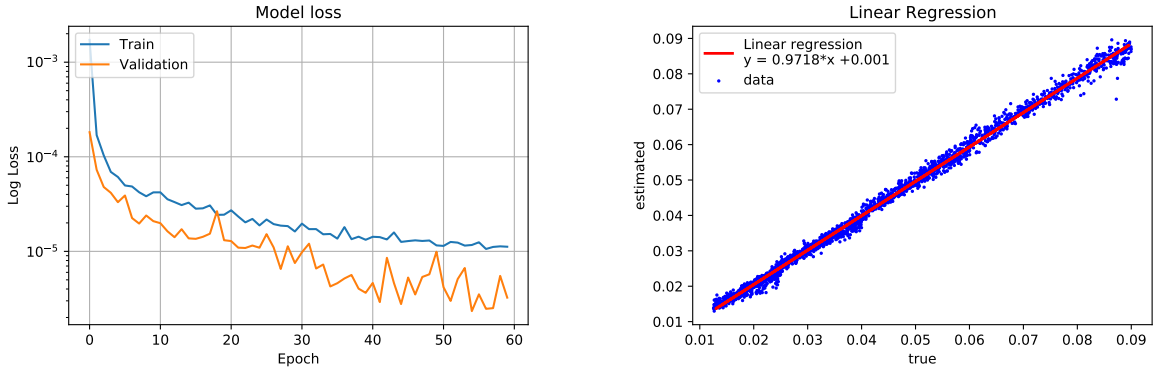
# 9 Results and discussion

The model architectures found after hyper-parameter search and testing are used, described in section 7. The estimation results from the models trained on data from simulation in long-crested irregular sea, under the assumption of perfect measurement, is presented in the first subsection. Subsequently the models are trained and tested on unfiltered noisy data, as a robustness test. Lastly the models are trained on data from simulation in long-crested regular sea. The data is downsampled to 20 Hz, since that is the sampling frequency of the IMUs in lab. Experimental data from lab is used as verification set, ensuring sufficient generalization of the models. The models are then fed the experimental acceleration data set from lab, and the estimation results are presented.

## 9.1 Long crested irregular waves

### 9.1.1 Perfect measurements

#### Specific wave height




**(a)** Plotted loss over number of epochs. **(b)** Linear regression plot for estimated over true $H_s$

**Figure 30:** Result $H_s$ model. Best validation loss registered after episode 54.

The model loss during training is seen in figure 30a, converging around epoch 60. Figure 30b shows the scatter plot of estimated versus true $H_s$, and a linear regression is performed giving an indication of the estimation accuracy. The following numerical results were achieved

- MSE: $2.189 \times 10^{-6} m^2$
- Mean error: 3.133%
- Standard deviation error: 3.177%

The results are very satisfactory in comparison to Arneson et al. (2019), discussed in section 2.4, where a data-driven approach is also tested for sea state estimation. The data in the paper is also generated from simulations in long-crested irregular waves using a single-peaked spectrum, based on very similar sea state domains. The paper presents an average significant wave height estimation

error of 0.7 m, or roughly 10%. The result is also satisfactory in comparison to Brodtkorb et al. (2018b), discussed in section 2.3, where an average significant wave height estimation error of 5.79% with a standard deviation of 3.78% is reported.

In figure 31 the peak frequency is plotted against wave direction for the cases that the $H_s$ model estimated with the largest error. The cases with an estimation error of more than the mean percentage error (3.133%) is shown. They are distributed relatively even over the wave directions, however it indicates that higher peak frequency, or lower peak period, makes it harder to estimate significant wave height.

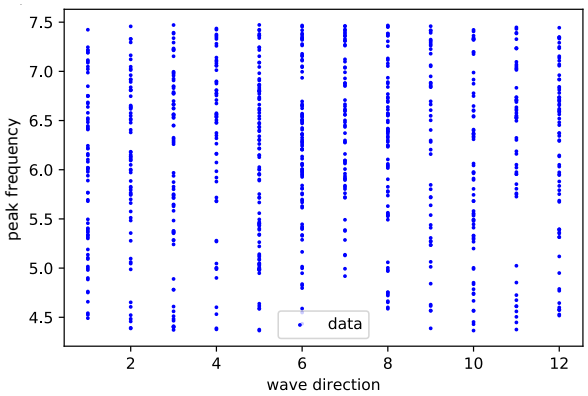
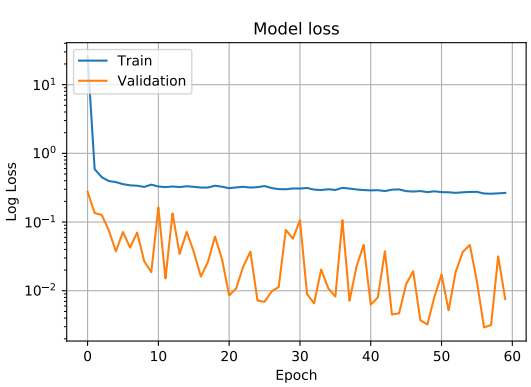
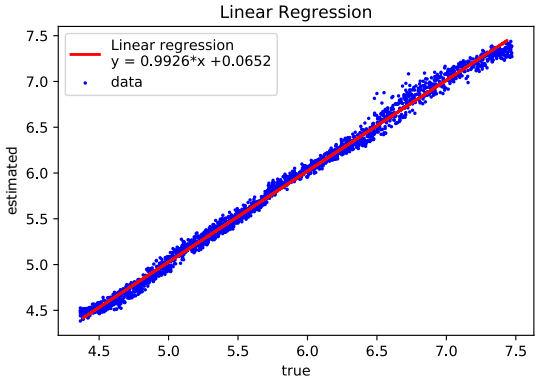



**Figure 31:** Characteristics of worst estimated cases

### Peak wave period/frequency




**(a)** Plotted loss over number of epochs.




**(b)** Scatter plot for estimated over true peak frequency

**Figure 32:** Result $T_p$ model after grid search for finding optimal number of layers per block and number of blocks. Best validation loss registered after episode 56.

The model loss during training is seen in figure 32a. Figure 32b shows the scatter plot of estimated versus true peak frequency, and a linear regression is performed giving an indication of the estimation accuracy. The following numerical results were achieved,

- MSE: $0.00391(rad/s)^2$
- Mean error: 0.896%
- Standard deviation error: 0.693%

These results are also very satisfactory compared to Arneson et al. (2019) and Brodtkorb et al. (2018b). The prior presents an average peak period error of 1.5s, translating to approximately 12%, while the latter presents an average peak period error of 7.59%.

In figure 33 the significant wave height is plotted against wave direction for the cases that the $T_p$ model estimated with the largest error. The cases with an estimation error of more than the mean percentage error (0.896%) is shown. They are distributed relatively even over the wave directions, however it indicates that higher significant wave height makes it harder to estimate peak period/frequency.

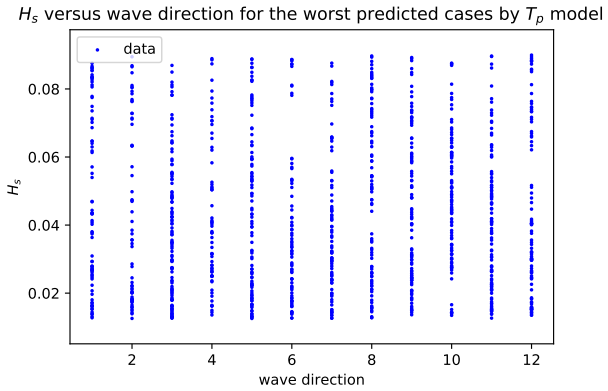



**Figure 33:** $H_s$ plotted agains wave direction for the worst estimated cases by the $T_p$ model. Worst cases defined as the cases with an estimation error of more than mean percentage error (0.896)

### Wave direction

The train and validation loss is plotted over number of epochs of training the wave direction model in figure 34, while the confusion matrix is seen in figure 35. The model achieve 100% accuracy on the test set. Even though each sector is as much as 30°, the often mentioned problems of distinguishing between head and tail sea, or port and starboard sea seems non-existent.

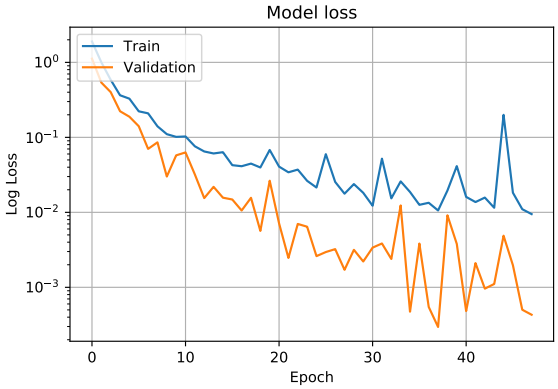



**Figure 34:** Plotted loss over number of epochs for wave direction model

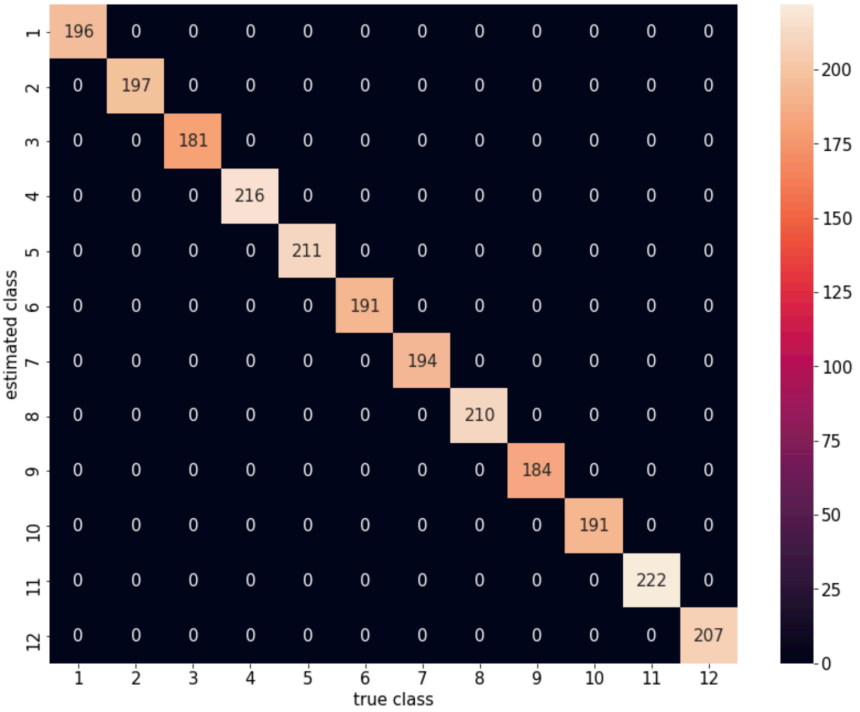



**Figure 35:** Confusion matrix for estimated wave direction sector

### 9.1.2 Measurements with unfiltered noise and bias

Following are the results from the models trained and tested on unfiltered IMU measurements with noise and bias based on the parameters:

- Gyro StD $= [0.0026, 0.0027, 0.0025]\frac{\pi}{180}$
- Accelerometer StD $= [0.001887, 0.0005518, 0.0008167]$

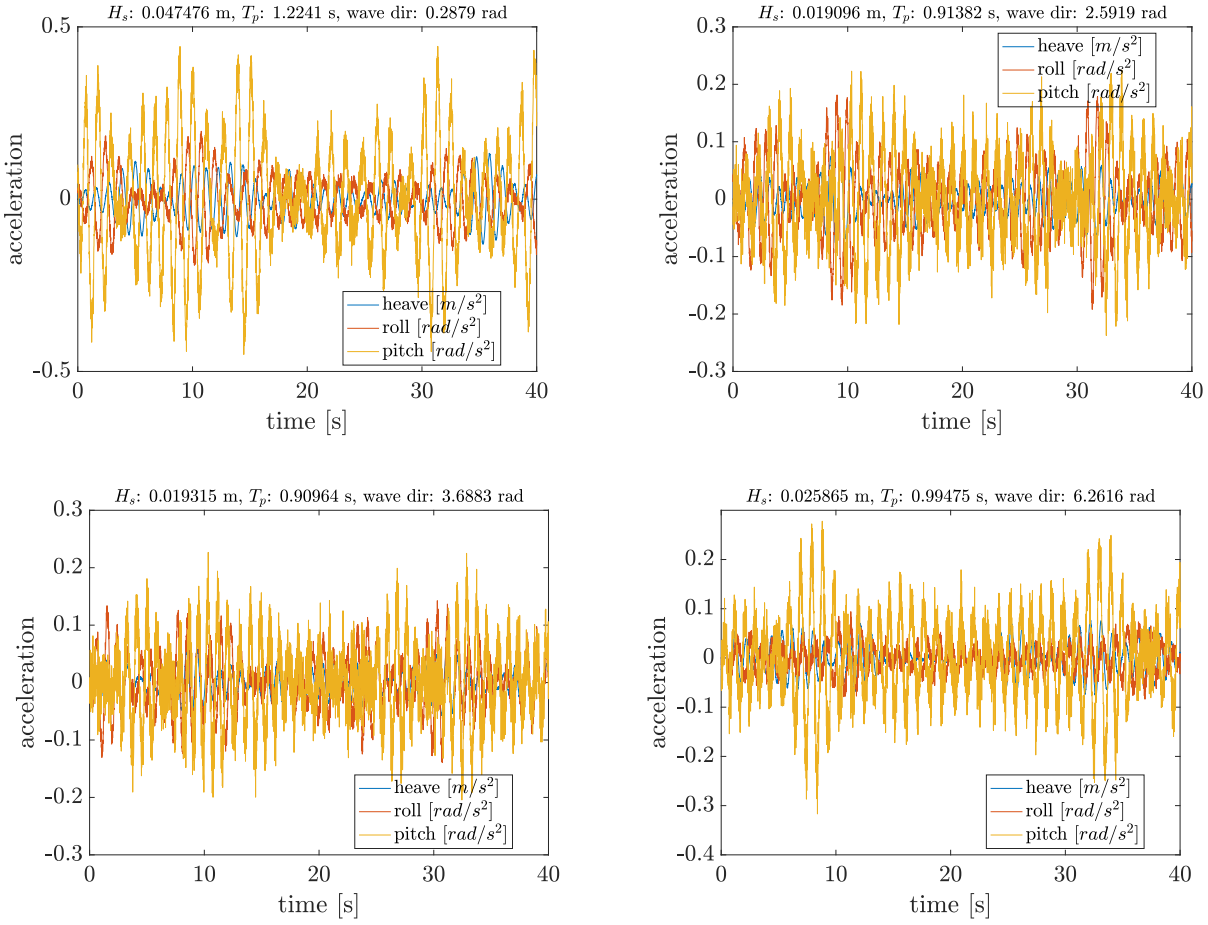
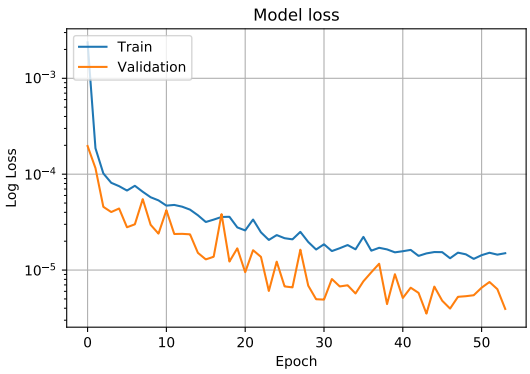
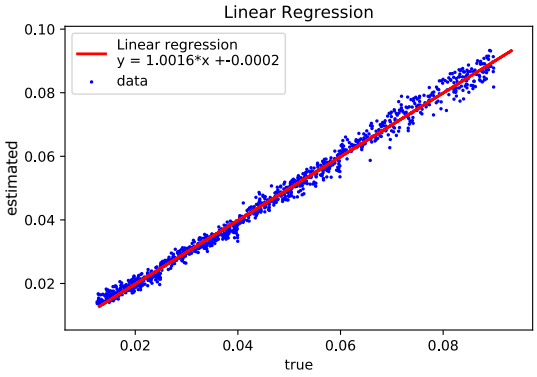



**Figure 36:** 3 DOF noisy acceleration data from different sea states in irregular sea

### Specific wave height




**(a)** Plotted loss over number of epochs.




**(b)** Linear regression plot for estimated over true $H_s$

**Figure 37:** Result $H_s$ model. Best validation loss registered after episode 43.

The linear regression plot in figure 37b shows very similar results as for the data generated under the assumption of perfect measurement. This demonstrates the filtering capability, and robustness, of the convolutional neural networks. Figure 38 also supports the indication that higher peak frequency, or lower peak period, makes it harder to estimate significant wave height. The numerical results for the specific wave height estimation were

- MSE: $3.6307 \times 10^{-6} m^2$
- Mean error: 3.8142%
- Standard deviation error: 3.6405%

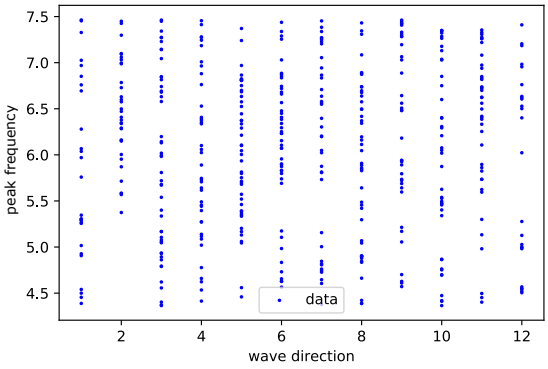
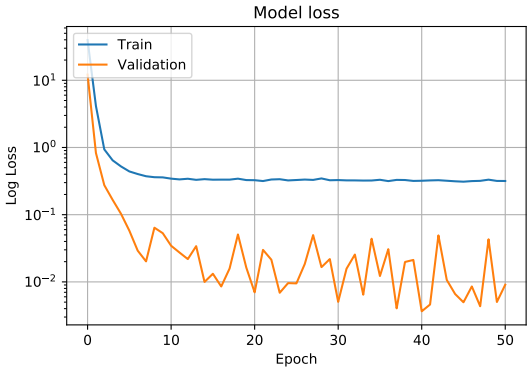
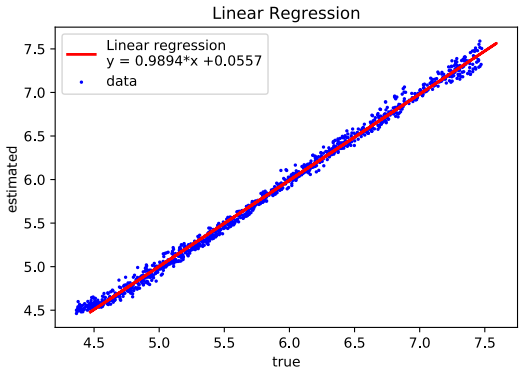



**Figure 38:** Peak frequency plotted agains wave direction for the worst estimated cases by the $H_s$ model. Worst cases defined as the cases with an estimation error of more than mean percentage error (3.814)

### Peak wave period/frequency




**(a)** Plotted loss over number of epochs.




**(b)** Linear regression plot for estimated over true peak frequency

**Figure 39:** Result $T_p$ model. Best validation loss registered after episode 40.

Also for the $T_p$ model, the results are equally good for the noisy data set, further substantiating the filtering capabilities of the CNNs. The numerical results achieved were

- MSE: $0.00361(rad/s)^2$
- Mean error: 0.848%
- Standard deviation error: 0.709%

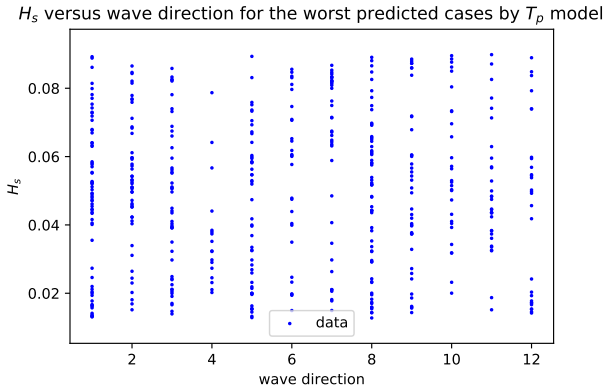



**Figure 40:** $H_s$ plotted agains wave direction for the worst estimated cases by the $T_p$ model. Worst cases defined as the cases with an estimation error of more than mean percentage error (0.848)

### Wave direction

The train and validation loss is plotted over number of epochs of training the wave direction model in figure 41, while the confusion matrix is seen in figure 42. The model achieve 100% accuracy on the test set.

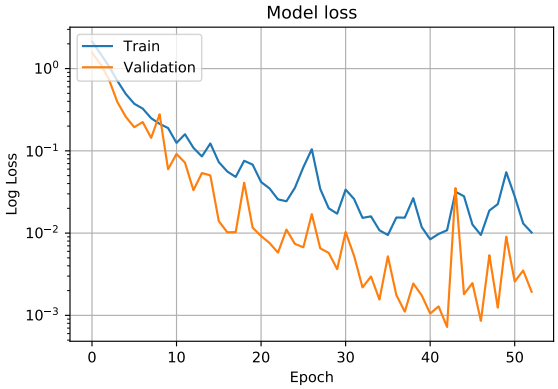



**Figure 41:** Plotted loss over number of epochs for wave direction model

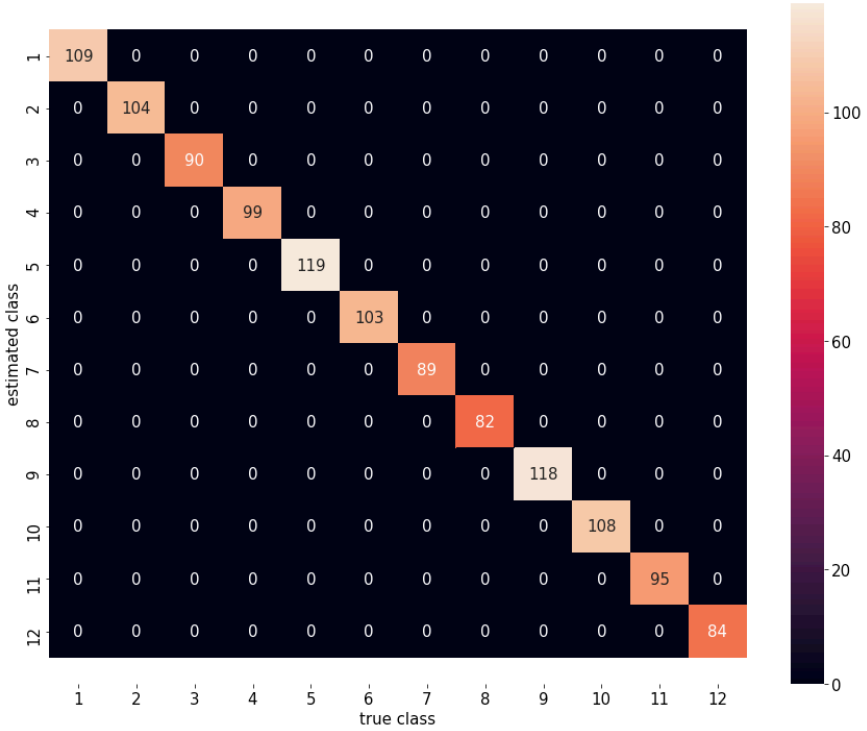
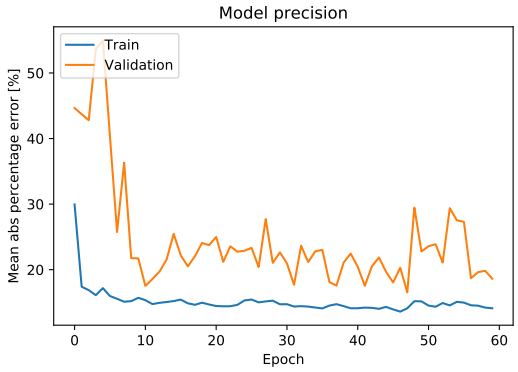



**Figure 42:** Confusion matrix for estimated wave direction sector

## 9.2 Long crested regular waves

The models are trained on IMU data, under the assumption of perfect measurement, from simulation in long-crested regular waves. They are subsequently tested on processed measurements from lab, see section 8.5.

### Wave height




**(a)** Mean absolute percentage error over number of epochs trained


**(b)** Model loss over number of epochs trained

**Figure 43:** Model performance over number of epochs trained.

Figure 43a shows the mean absolute percentage error over number of epochs trained for the training and validation set. The percentage error relatively early starts oscillating around 20%, and the final model achieve approximately 16%. This is a good result considering the discrepancies between modeled and actual motion of the vessel. The wave height estimation is probably sensitive to modeling inaccuracies, since the value of the max accelerations are presumably weighted heavily during estimation. The results achieved were

- MSE: $4.7318 \times 10^{-4} m^2$
- Mean error: 16.5379%
- Standard deviation error: 11.6194%

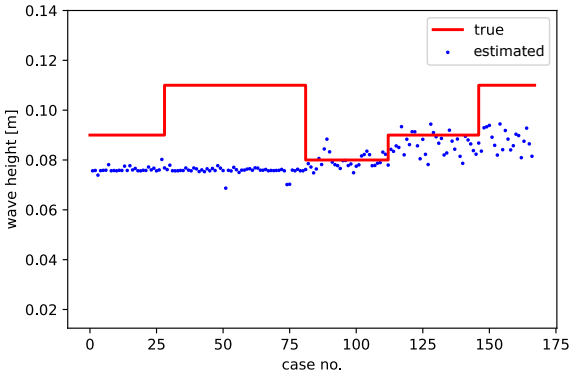
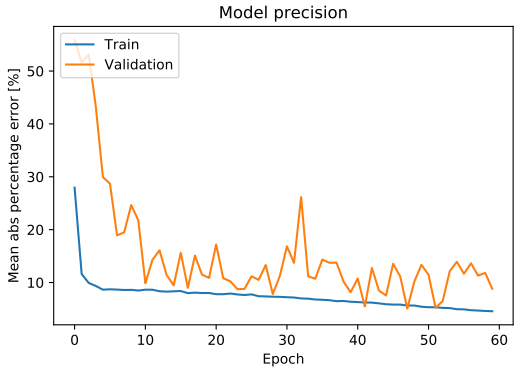
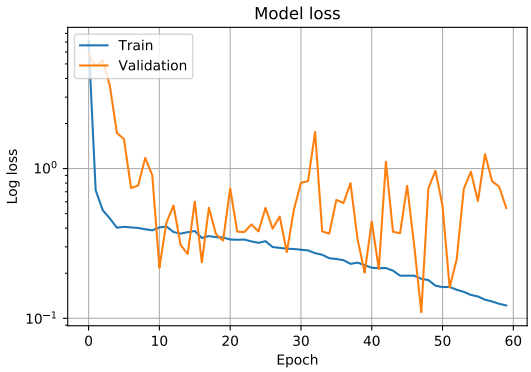



**Figure 44:** Estimated wave height for experimental data

### Wave period/frequency




**(a)** Mean absolute percentage error over number of epochs trained




**(b)** Model loss over number of epochs trained

**Figure 45:** Result wave period model trained on simulation data, with experimental data as validation set.

Figure 45a shows the mean absolute percentage error over number of epochs trained for the training and validation set. The percentage error relatively early starts oscillating around 10%, and the final model achieve approximately 5%. This is a satisfying result in comparison to the model-based approach of Brodtkorb et al. (2018b), which also tests on real data from DP vessel operation. As mentioned, the paper presents a mean error of 7.59%. The results achieved were

- MSE: $0.10987(rad/s)^2$
- Mean error: 5.076%
- Standard deviation error: 6.175%

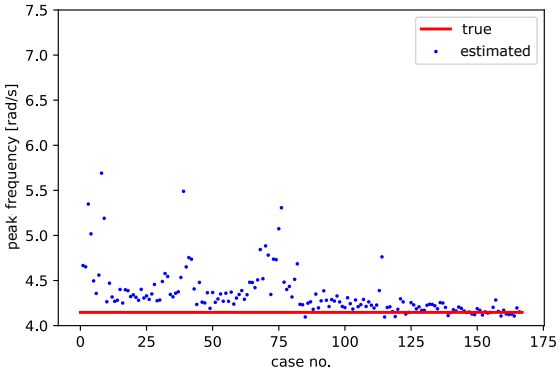



**Figure 46:** Estimated wave frequency for experimental data

### Wave direction

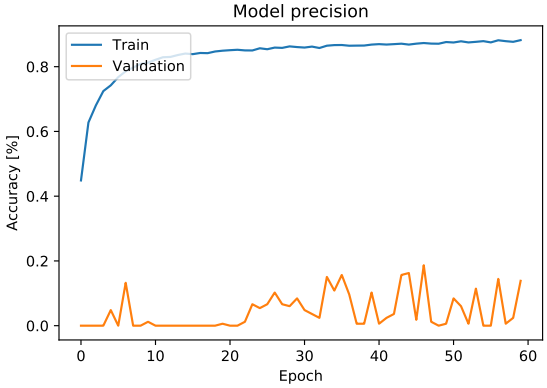



**Figure 47:** Accuracy estimated wave direction for experimental data

The estimation results for wave direction, seen in figure 48, are not very good. It can be partly explained by the sensitivity to modeling inaccuracies. The model is mostly able to distinguish between head and tail sea, but the fine details in the motion that distinguishes between the degrees of head/starboard sea is clearly not modeled accurately.

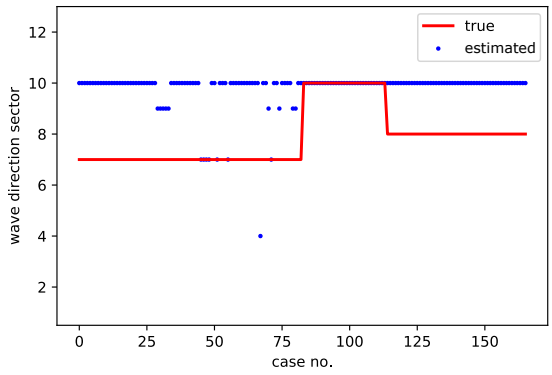



**Figure 48:** Estimated wave direction for experimental data

## 9.3 General discussion

The data-driven sea state estimation models using convolutional neural networks perform very well for long-crested irregular waves, and achieves results that are similar or better than other published model-based and data-driven approaches. The results show that the models have no problem with estimating the higher sea states, which has introduced problems for other sea state estimation approaches. The common problem of differentiating between head and tail sea also seems non-existent. The results for the noisy measurements show the robustness of the models. It is an end-to-end approach, not sensitive to human interventioned pre-processing, nor dependent on hand-crafted features. The results on the experimental data are also relatively good, considering

the simplification and inaccuracies in the simulation model.

Another advantage for these models is their effectiveness, using only 40 seconds of response measurements for the estimations in irregular sea. In comparison, the approach in Brodtkorb et al. (2018b) uses 10 minutes. For regular sea, the models use only 10 seconds of response measurements. This makes the models applicable for online sea state estimation. The approach can also be used for initial estimates for other better tested approaches.

One reason for great results for irregular sea is potentially that the sea is modeled with five waves in the spectrum, see section 4.6, and only for long-crested sea. Another limiting factor is that the sea states tested are moderate to high sea. The capabilities of the models for very small and extreme sea states are hence not known. This is, however, to the author's knowledge, still a similar approach as the other mentioned papers.

A noticeable and important characteristic of all the loss plots from the training of the models for irregular sea data is that the validation loss is lower than the training loss. This is something to be careful with since it can be a sign of data leakage, bad sampling, or the validation set being too small. Data leakage is when parts of the training data are present in the validation data - indicating artificially good generalization of the model. The loss achieved on the test set is, however, for all models, at the same level as the best validation loss. This substantiates that the validation set is in fact a good representation of the test set. The reason for the lower validation loss can therefore be explained by two factors. Firstly the regularization caused by the dropout layer is only applied to the training data, not during validation and testing. As explained in section 6.4.3, the dropout layer ignores the contribution of half the nodes, resulting in less precision. The last factor is that the training loss is calculated during an epoch, and based on all the batches, while the validation loss is calculated after each epoch. The validation loss is therefore based, on average, on half an epoch more of training.

# 10 Conclusion

The simulation model was successfully implemented, with the ability to generate realistic IMU measurements from the DP vessel in long-crested regular and irregular sea. Pre-processing techniques were applied to the measurements, enabling the generation of labeled data set consisting of estimated vessel accelerations in center of control.

A computationally effective data-driven sea state estimation approach for DP vessels was presented. The convolutional networks require very little pre-processing and no hand-engineered feature extraction. For long-crested irregular sea the method produced good results, with a mean significant wave height error of 3.133%, a mean peak period error of 0.896%, and 100% accuracy in classifying the relative wave direction within the right 30° sector. Very similar results were accomplished on the raw, noisy acceleration data, demonstrating the robustness of the model.

Four IMUs were installed on the C/S Arctic Drillship in MC-lab, and tested for the vessel in long-crested regular sea. The results indicated the feasibility of a sea state estimator trained on simulated data, however reflecting the discrepancies between modeled and real vessel dynamics.

The dilemma is that a data-driven approach is desirable as it does not require knowledge about the vessel characteristics, however, it relies on a large amount of collected sensor data from operation. It has proven hard to collect the amount and broad distribution of data needed to train a sufficiently robust model. The alternative is, as attempted in this thesis, to train the model on simulated data. This takes us back to the original problem, where the simulation model has to capture all the real-life dynamics of the vessel as well as waves, which again requires precise knowledge about the vessel characteristics.

This thesis was, however, not meant to solve the aforementioned problem. The objective was rather to demonstrate that, given IMU measurement data from a DP vessel in a sufficient number of different sea states, a data-driven approach using CNNs for sea state estimation is computationally effective, robust, and accurate.

# 11 Further work

Suggestions for further work are

- Test whether a recurrent neural network, or a hybrid model with a convolutional neural network, could improve the sea state estimation capabilities.
- Extend the approach to also consider forward speed of the vessel, perhaps by using the speed and heading/coarse angle as additional input to the neural networks.
- Test the approach for short-crested waves.
- Test the approach for simulation data generated from other wave spectrums, like Jonswap

and Torsethaugen. Perhaps also with other more complex wave spectrum parameters.

- Use the estimated relative wave direction as input to a heading controller for optimal heading control.
- Tune the controller and collect more data in lab from the vessel in actual DP mode.

## References


Arneson, I.B., Brodtkorb, A.H., Sørensen, A.J., 2019. Sea State Estimation Using Quadratic Discriminant Analysis and Partial Least Squares Regression. IFAC-PapersOnLine 52, 72–77. URL: http://www.sciencedirect.com/science/article/pii/S2405896319321706, doi:10.1016/j.ifacol.2019.12.285.

Bjørnø, J., 2016. Thruster-Assisted Position Mooring of C/S Inocean Cat I Drillship. 147 URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2402895.

Brodtkorb, A.H., Nielsen, U.D., Sørensen, A.J., 2018a. Online wave estimation using vessel motion measurementsThis work was supported by the Research Council of Norway through the Centres of Excellence funding scheme, project number 223254 - NTNU AMOS. IFAC-PapersOnLine 51, 244–249. URL: http://www.sciencedirect.com/science/article/pii/S2405896318321992, doi:10.1016/j.ifacol.2018.09.510.

Brodtkorb, A.H., Nielsen, U.D., Sørensen, A.J., 2018b. Sea state estimation using vessel response in dynamic positioning. Applied Ocean Research 70, 76–86. URL: http://www.sciencedirect.com/science/article/pii/S0141118717302481, doi:10.1016/j.apor.2017.09.005.

Browlee, J., 2018. A Gentle Introduction to Dropout for Regularizing Deep Neural Networks URL: https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/.

Cheng, X., Li, G., Ellefsen, A.L., Chen, S., Hildre, H.P., Zhang, H., 2020. A Novel Densely Connected Convolutional Neural Network for Sea State Estimation Using Ship Motion Data. IEEE Transactions on Instrumentation and Measurement , 1–1doi:10.1109/TIM.2020.2967115. conference Name: IEEE Transactions on Instrumentation and Measurement.

Cheng, X., Li, G., Skulstad, R., Chen, S., Hildre, H.P., Zhang, H., 2019. Modeling and Analysis of Motion Data from Dynamically Positioned Vessels for Sea State Estimation, in: 2019 International Conference on Robotics and Automation (ICRA), pp. 6644–6650. doi:10.1109/ICRA.2019.8794069. iSSN: 2577-087X.

Cho, K., van Merrienboer, B., Bahdanau, D., Bengio, Y., 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. arXiv:1409.1259 [cs, stat] URL: http://arxiv.org/abs/1409.1259. arXiv: 1409.1259.

Clauss, G.F., Kosleck, S., Testa, D., Hessner, K., 2009a. Forecast of Critical Situations in Short-Crested Seas, American Society of Mechanical Engineers Digital Collection. pp. 217–226. URL: https://asmedigitalcollection.asme.org/OMAE/proceedings/OMAE2008/48197/217/331158, doi:10.1115/OMAE2008-57188.

Clauss, G.F., Testa, D., Kosleck, S., Stuĺck, R., 2009b. Forecast of Critical Wave Groups From Surface Elevation Snapshots, American Society of Mechanical Engineers Digital Collection. pp. 79–

88. URL: https://asmedigitalcollection.asme.org/OMAE/proceedings/OMAE2007/42681/79/323314, doi:10.1115/OMAE2007-29090.

Devices, A., . IMU technical document. URL: https://www.analog.com/media/en/technical-documentation/data-sheets/ADIS16350_16355.pdf.

Diamant, R., Jin, Y., 2014. A Machine Learning Approach for Dead-Reckoning Navigation at Sea Using a Single Accelerometer. IEEE Journal of Oceanic Engineering 39, 672–684. doi:10.1109/JOE.2013.2279421.

Duchi, J., Hazan, E., Singer, Y., 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research 12, 2121–2159. URL: http://jmlr.org/papers/v12/duchi11a.html.

EMSA, 2019. Annual Overview of Marine Casualties and Incidents 2019. Technical Report. URL: http://www.emsa.europa.eu/accident-investigation-publications/annual-overview.html.

Feng, J., He, X., Teng, Q., Ren, C., Chen, H., Li, Y., 2019. Reconstruction of porous media from extremely limited information using conditional generative adversarial networks. Physical Review E 100. doi:10.1103/PhysRevE.100.033308.

Ferrandis, J.d.., Triantafyllou, M., Chryssostomidis, C., Karniadakis, G., 2019. Learning functionals via LSTM neural networks for predicting vessel dynamics in extreme sea states. arXiv:1912.13382 [cs] URL: http://arxiv.org/abs/1912.13382. arXiv: 1912.13382.

Fossen, T.I., 2011. Handbook of Marine Craft Hydrodynamics and Motion Control. John Wiley & Sons. Google-Books-ID: oR3sBgAAQBAJ.

Frederich, P., 2016. Constrained Optimal Thrust Allocation for C/S Inocean Cat I Drillship URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2415123.

Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R., 2012. Improving neural networks by preventing co-adaptation of feature detectors. arXiv:1207.0580 [cs] URL: http://arxiv.org/abs/1207.0580. arXiv: 1207.0580.

Hirayama, T., Hagino, Y., 1985. REAL-TIME ESTIMATION OF SEA SPECTRA BASED ON MOTIONS OF A RUNNING SHIP - FULL SCALE TRIAL URL: https://trid.trb.org/view/427432.

Hochreiter, S., Schmidhuber, J., 1997. Long Short-Term Memory. Neural Computation 9, 1735–1780. URL: https://doi.org/10.1162/neco.1997.9.8.1735, doi:10.1162/neco.1997.9.8.1735.

IMCA, 2010. Introduction to Dynamic Positioning. URL: https://web.archive.org/web/20100626080452/http://www.imca-int.com/divisions/marine/reference/intro01.html.

Institute, T.N., 2013. What is Dynamic Positioning? URL: https://web.archive.org/web/20130125101320/http://www.nautinst.org/en/dynamic-positioning/what-is-dynamic-positioning/index.cfm.

Iseki, T., Ohtsu, K., 2000. Bayesian estimation of directional wave spectra based on ship motions. Control Engineering Practice 8, 215–219. URL: http://www.sciencedirect.com/science/article/pii/S0967066199001562, doi:10.1016/S0967-0661(99)00156-2.

Ludvigsen, M., Sørensen, A.J., 2016. Towards integrated autonomous underwater operations for ocean mapping and monitoring. Annual Reviews in Control 42, 145–157. URL: http://www.sciencedirect.com/science/article/pii/S1367578816300256, doi:10.1016/j.arcontrol.2016.09.013.

Lyngstadaas, O.N., 2018. Ship Motion Control Concepts Considering Actuator Constraints URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2560385.

Løvås, H., 2019. DP autotuning by use of derivative-free optimization. Ph.D. thesis.

Mak, B., Düz, B., 2019. Ship As a Wave Buoy: Estimating Relative Wave Direction From In-Service Ship Motion Measurements Using Machine Learning, American Society of Mechanical Engineers Digital Collection. URL: https://asmedigitalcollection.asme.org/OMAE/proceedings/OMAE2019/58882/V009T13A043/1068082, doi:10.1115/OMAE2019-96201.

Marine Cybernetics Laboratory Handbook, D.o.M.T., 2017. Marine Cybernetics Laboratory Handbook.

Missinglink.ai, 2019. Keras Conv1D: Working with 1D Convolutional Neural Networks in Keras URL: https://missinglink.ai/guides/keras/keras-conv1d-working-1d-convolutional-neural-networks-keras/.

Nielsen, U.D., 2007. Response-based estimation of sea state parametersinfluence of filtering. Ocean Engineering 34, 1797–1810. URL: http://www.sciencedirect.com/science/article/pii/S0029801807000625, doi:10.1016/j.oceaneng.2007.03.002.

Nielsen, U.D., Bjerregård, M., Galeazzi, R., Fossen, T.I., 2015. New concepts for shipboard sea state estimation, in: OCEANS 2015 - MTS/IEEE Washington, pp. 1–10. doi:10.23919/OCEANS.2015.7404386. iSSN: null.

Nielsen, U.D., Brodtkorb, A.H., Sørensen, A.J., 2018. A brute-force spectral approach for wave estimation using measured vessel motions. Marine Structures 60, 101–121. URL: http://www.sciencedirect.com/science/article/pii/S0951833917303209, doi:10.1016/j.marstruc.2018.03.011.

Nielsen, U.D., Galeazzi, R., Brodtkorb, A.H., 2016. Evaluation of shipboard wave estimation techniques through model-scale experiments, in: OCEANS 2016 - Shanghai, pp. 1–8. doi:10.1109/OCEANSAP.2016.7485701.

NIST, 2016. Autonomy Levels For Unmanned Systems. URL: https://www.nist.gov/el/intelligent-systems-division-73500/cognition-and-collaboration-systems/autonomy-levels-unmanned.

Niu, M., Li, Y., Wang, C., Han, K., 2018. RFAmyloid: A Web Server for Predicting Amyloid Proteins. International journal of molecular sciences 19. doi:10.3390/ijms19072071.

Pascoal, R., Guedes Soares, C., 2008. Non-parametric wave spectral estimation using vessel motions. Applied Ocean Research 30, 46–53. URL: http://www.sciencedirect.com/science/article/pii/S0141118708000163, doi:10.1016/j.apor.2008.03.003.

Pascoal, R., Guedes Soares, C., 2009. Kalman filtering of vessel motions for ocean wave directional spectrum estimation. Ocean Engineering 36, 477–488. URL: http://www.sciencedirect.com/science/article/pii/S0029801809000183, doi:10.1016/j.oceaneng.2009.01.013.

Pascoal, R., Guedes Soares, C., Sorensen, A.J., 2008. Ocean Wave Spectral Estimation Using Vessel Wave Frequency Motions, American Society of Mechanical Engineers Digital Collection. pp. 337–345. URL: https://asmedigitalcollection.asme.org/OMAE/proceedings/OMAE2005/41960/337/303111, doi:10.1115/OMAE2005-67584.

Perez, T., Fossen, T.I., 2007. Kinematic Models for Manoeuvring and Seakeeping of Marine Vessels. Modeling, Identification and Control: A Norwegian Research Bulletin 28, 19–30. URL: http://www.mic-journal.no/ABS/MIC-2007-1-3.asp, doi:10.4173/mic.2007.1.3.

Prabhu, 2018. Understanding of Convolutional Neural Network (CNN) Deep Learning URL: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148.

Price, W.G., 1974. Probabilistic Theory of Ship Dynamics. University College London, Published by: Chapman & Hall Ltd, London, ISBN: 0 412 12430 0, Printed in the United Kingdom URL: https://repository.tudelft.nl/islandora/object/uuid%3A51f2ac13-0639-48ab-b3c4-710a79f969f2.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323, 533–536. URL: https://www.nature.com/articles/323533a0, doi:10.1038/323533a0. number: 6088 Publisher: Nature Publishing Group.

Saxena, S., 2020. Underfitting vs. Overfitting (vs. Best Fitting) in Machine Learning URL: https://www.analyticsvidhya.com/blog/2020/02/underfitting-overfitting-best-fitting-machine-learning/.

Skjetne, R., 2018. Kinematic IMU equations.

Skjetne, R., 2019. Maneuvering-based guidance design for dynamic positioning.

Stensvold, T., . Yara Birkeland: Leveres fra Vard Brattvaag etter sommeren URL: https://www.tu.no/artikler/yara-birkeland-leveres-fra-vard-brattvaag-etter-sommeren/484547.

Stredulinsky, D.C., Thornhill, E.M., 2011. Ship Motion and Wave Radar Data Fusion for Shipboard Wave Measurement. URL: https://www.ingentaconnect.com/content/sname/jsr/2011/00000055/00000002/art00001.

Sørensen, A.J., Sagatun, S.I., Fossen, T.I., 1996. Design of a dynamic positioning system using model-based control. Control Engineering Practice 4, 359–368. URL: http://www.sciencedirect.com/science/article/pii/0967066196000135, doi:10.1016/0967-0661(96)00013-5.

Takekuma, K., Takahashi, T., 1972. On the evaluation of sea spectra based on measured ship motions. Mitsubishi Heavy Industries Ltd, Proceedings of the 13th International Towing Tank Conference, ITTC72, Subject Seakeeping, Paper: P1972-5 Proceedings. URL: https://repository.tudelft.nl/islandora/object/uuid%3Ab2903208-1720-4b6f-b999-7413a30bfb5a.

Tannuri, E.A., Sparano, J.V., Simos, A.N., Da Cruz, J.J., 2003. Estimating directional wave spectrum based on stationary ship motion measurements. Applied Ocean Research 25, 243–261. URL: http://www.sciencedirect.com/science/article/pii/S0141118704000070, doi:10.1016/j.apor.2004.01.003.

Tu, F., Ge, S.S., Choo, Y.S., Hang, C.C., 2018. Sea state identification based on vessel motion response learning via multi-layer classifiers. Ocean Engineering 147, 318–332. URL: http://www.sciencedirect.com/science/article/pii/S0029801817305188, doi:10.1016/j.oceaneng.2017.08.047.

Udjus, G., 2017. Force Field Identification and Positioning Control of an Autonomous Vessel using Inertial Measurement Units. Ph.D. thesis.

Vieira, S., Pinaya, W., Mechelli, A., 2017. Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications. Neuroscience & Biobehavioral Reviews 74. doi:10.1016/j.neubiorev.2017.01.002.

Vo, A., 2018. Deep Learning Computer Vision and Convolutional Neural Networks URL: https://anhvnn.wordpress.com/2018/02/01/deep-learning-computer-vision-and-convolutional-neural-networks/.

Wang, J., 2018. Keras samples:LeNet-5 on cifar10 dataset. URL: https://bywmm.github.io/2018/12/14/keras%20samples-%20LeNet-5%20on%20cifar10%20dataset/.

Zappa, B., Legnani, G., van den Bogert, A.J., Adamini, R., 2001. On the Number and Placement of Accelerometers for Angular Velocity and Acceleration Determination. Journal of Dynamic Systems, Measurement, and Control 123, 552–554. URL: https://asmedigitalcollection.asme.org/dynamicsystems/article/123/3/552/460354/

On-the-Number-and-Placement-of-Accelerometers-for, doi:10.1115/1.1386649. publisher: American Society of Mechanical Engineers Digital Collection.