Gjermund Prestegårdshus

Route modelling in coastal navigation for digital twins and digital platforms

Master's thesis in Marine Technology
Supervisor: Professor Stein Ove Erikstad
August 2020

Master's thesis

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

NTNU
Norwegian University of
Science and Technology

Gjermund Prestegårdshus

Route modelling in coastal navigation for digital twins and digital platforms

Master's thesis in Marine Technology
Supervisor: Professor Stein Ove Erikstad
August 2020

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

NTNU
Norwegian University of
Science and Technology

MSc thesis in Marine Systems Design

Stud. techn. Gjermund Prestegårdshus

Route modelling in coastal navigation for digital twins and digital platforms

Spring 2020

Background

Coastal navigation is navigation where there generally is open sea in one direction and shore in the other. Coastal navigation also includes navigation through limited sea space, e.g. fairway navigation. This type of navigation causes a variety of route options, as well as hazards to avoid, such as banks, reefs, skerries and traffic. In addition, coastal waters have environmental hazards as fog, light, tidal variation of sea elevation, tide, current, wind and waves. These factors raise challenges to navigators, where analysis of coastal bathymetry and metocean conditions in close-to real time is necessary.

Another factor in coastal navigation is the possibility to go inshore or offshore. Routing inshore is sheltered, thus reducing the risk of encountering large waves and wind. Sailing offshore is more exposed to oceanic conditions and -loads, hence larger ship responses. When sailing along the coast of e.g. Norway, sailing offshore is often shorter than sailing inshore. This is a trade-off between longer sailing distance, higher chance of grounding, stranding and collisions by inshore routing, and larger ship motions and loads by offshore routing.

In modern navigation, better weather forecasts and more electronic navigational aids are available. However, there are currently no on-bridge navigation tool that gives clear decision-support of the better route alternatives in real time. A possible way forward is to introduce a digital twin framework to simulate predicted behaviour along a planned route, to give insight and decision support of route alternatives. A digital twin is a virtual model that captures the state and behaviour of a real asset in close to real time, based on sensor input.

Given that the digital twin can estimate predictions of the state of the vessel during the alternative routes, a set of key performance indicators can be derived to give decision support to what routes to choose. Key performance indicators in coastal navigation are e.g. chance of seasickness, fuel consumption and sailing time.

Overall aim and focus

The overall aim of this thesis is to develop a coastal navigation model using a digital twin framework to assess key performance indicators. The model aims to study how to manage fidelity in coastal navigation. Moreover, a simplified digital twin should be formulated to investigate how a digital twin can be applied to gain insight and decision support to engineering tasks. Relevant data are ocean maps and historical data of weather measurements. The digital twin should derive a simplified response in 1 DOF (heave) based on simulated waves.

The thesis will discuss the quality of the navigation model. Additionally, recommendations for improvements and further development should be stated.

Scope and main activities

The candidate should presumably cover the following main points:

1. *Provide a literature review of the state-of-the-art and state-of-the-technology of digital twins and digital services. The literature should be relevant and clarify the state, application and possibilities of the technology.*

2. *Develop a coastal navigation model based on open-source maps and historical weather measurements. The model should be applicable for multiple coastal areas and with various fidelity requirements.*

3. *Develop a ship-in-operation model to estimate probable sea states and vessel response due to external loads. The model should assign a simplified response in heave (1 DOF) of all alternative sailing routes.*

4. *Extend the analysis to include optimisation.*

5. *Discuss, conclude and recommend further research within the topic.*

Modus operandi

Professor Stein Ove Erikstad at NTNU Trondheim will be the responsible advisor.

The work shall follow the guidelines given by NTNU for the MSc project work.

Stein Ove Erikstad

Professor/Responsible Advisor

Abstract

This thesis seeks to investigate how digital twin technology can be used to make smarter decisions and operate more efficiently in coastal navigation. The research questions were: (1) *What are limiting factors of simulation-based route modelling?* (2) *How do we manage fidelity?* (3) *How can we utilise digital twin technology in coastal navigation?* The research was successful in answering these questions.

A proposed digital framework was constructed for a route modelling approach based on set partitioning and simulated responses of a vessel in operation. The route model was created with high versatility, where altering a set of input variables could give a wide range of route alternatives. The network variables are origin, destination, spatial extent, and number of splits in direction 1 and 2, relative to the linear path between origin and destination. The simulated responses are dependent on the weather update variables: start of simulation, number of weather updates and time between each weather update.

A case study of RV Gunnerus was conducted. The main results show that a 3-section network with 3 network fidelity variations gave satisfying results. Furthermore, it was concluded that for large-scale networks requiring high-fidelity routing, a multiple-section network should be used, rather than a complete network from origin to destination. The model was successful in analysing seasonal variations, presenting optimal path variations, however with no clear seasonal trend. The value of a digital twin for RV Gunnerus is also discussed, where it was concluded that a baseline digital twin would be beneficial for real-time decision-support. With increasing technology maturity, the proposed simulation-based route model of this project could be included in a load-based digital twin.

For further research, the main efforts should be to further develop the route model to include subset-partitioning. This would be beneficial to computational time and end-use of the service. Additionally, a more accurate optimisation model should be constructed with normalised decision variables. Finally, the optimisation model should be algorithm-based, where the main goal is to find global optima, rather than local optima which was the method of this project.

Sammendrag

Denne oppgaven undersøker hvordan digital tvilling teknologi kan brukes til å ta smartere beslutninger og operere mer effektivt i kystnavigering. Forskningsspørsmålene for denne oppgaven er: (1) *Hva er begrensende faktorer ved simuleringsbasert rutemodellering?* (2) *Hvordan håndterer vi oppløsning og presisjon?* (3) *Hvordan kan digital tvilling teknologi brukes i kystnavigasjon?* Oppgaven har lyktes med å svare på disse spørsmålene.

Et foreslått digitalt rammeverk ble konstruert for en rutemodelleringstilnærming basert på nettverkspartisjon og simulert respons fra et skip i drift. Rutemodellen ble opprettet med høy allsidighet, der relativt små endringer i variablene kan gi et bredt spekter av rutealternativer. Nettverksvariablene er startpunkt, sluttpunkt, romlig utstrekning og antall partisjoner i retning 1 og 2, relativt til den lineære banen mellom start- og sluttpunkt. Den simulerte responsen er avhengig av væroppdateringsvariablene: simuleringsstart, antall væroppdateringer og tid mellom hver væroppdatering.

En casestudie av RV Gunnerus ble gjennomført. Hovedresultatene viser at et 3-seksjonsnettverk med 3 nettverksvariasjoner ga tilfredsstillende resultater. Videre ble det konkludert med at for storskala nettverk som krever ruting med høy oppløsning, bør et flerseksjonsnettverk brukes, i stedet for ett nettverk fra start- til sluttpunkt. Modellen var vellykket med å analysere sesongvariasjoner, og presenterte varierende optimale ruter, men ga ingen tydelige sesongtrender. Verdien av en digital tvilling for RV Gunnerus blir også diskutert, der det ble konkludert med at en sensorbasert digital tvilling ville være gunstig for sanntids beslutningsstøtte. Med økende teknologimodenhet kan den foreslåtte simuleringsbaserte rutemodellen for dette prosjektet inkluderes i en lastbasert digital tvilling.

For videre arbeid, bør hovedinnsatsen være å videreutvikle rutemodellen til å omfatte del-partisjon. Dette vil være fordelaktig for programmets kjøretid og sluttbruk av tjenesten. I tillegg bør en mer nøyaktig optimaliseringsmodell konstrueres med normaliserte beslutningsvariabler. Til slutt bør optimaliseringsmodellen være algoritmebasert, der hovedmålet er å finne globalt optima, i stedet for lokalt optima som var metoden for dette prosjektet.

Preface

This thesis concludes a Master of Science degree in Marine Technology with specialisation in Marine Systems Design at the Norwegian University of Science and Technology. It is written in the spring of 2020 with a workload equal to 30 ECTS. The wish to challenge my own thoughts to think of something new has driven me throughout this project. I hope that my results can be of inspiration to develop digitalisation further within the maritime sector.

I would like to thank my supervisor Stein Ove Erikstad for pushing me in the right direction with this topic, and for good discussions of how to keep on progress. I would also like to thank Astrid H. Brodtkorb for giving me access to some of the MATLAB files in the project of making dynamic positioning- and manoeuvring simulation models for RV Gunnerus. Even though I did not use it in this project, I gained insight to how a simulation model for digital twin applications is constructed and the use cases for it. Lastly, thank you to Endre Sandvik for sharing a copy of his Doctoral thesis, it has been helpful.

Gjermund Prestegårdshus

Gjermund Prestegårdshus

Trondheim, August 2020

Contents

List of Tables

List of Figures

Abbreviations

AI	Artificial intelligence
AIS	Automatic Identification System
API	Application Programming Interface
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CFD	Computational Fluid Dynamics
CM	Configuration Management
DAS	Deterministic Amplitude Scheme
DOF	Degree of Freedom
DP	Design Parameter
DT	Digital Twin
ECDIS	Electronic Chart Display and Information System
ECMWF	European Centre for Medium-Range Weather Forecasts
ENC	Electronic Navigational Chart
FEA	Finite Element Analysis
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
FR	Functional Requirement
HIL	Hardware-In-the-Loop
IoT	Internet-Of-Things
JONSWAP	Joint North Sea Wave Observation Project
KPI	Key Performance Indicator
MBD	Multibody Dynamics
ML	Machine Learning
OSP	Open Simulation Platform

PDM	Product Data Models
PM	Pierson-Moskovitz
POB	People-On-Board
RAS	Random Amplitude Scheme
RV	Research Vessel
SDF	Spectral Density Function
SP	Set Partitioning
UI	User Interface
WP	Waypoint

Chapter 1

Introduction

The overall aim of this thesis is to develop a coastal navigation model using a digital twin framework to assess key performance indicators. This chapter will firstly present the motivation for choosing this topic to see how the results can be applied. Secondly, a brief problem description is presented. Lastly, the structure of the report is described.

1.1 Motivation

The growing threat of climate change is calling for joint action to limit our own damage to the environment we live in and depend on. Polar sea ice is melting, biodiversity is shrinking, and we experience more frequent extreme weather than ever before. These are consequences of our own ignorance to the world we live in, which the global community now starts to acknowledge. To change the pace and course of this development, we first need to understand how the world works, and as 71% of the earth's surface is covered with water, most of this information is stored within the oceans.

Studying the marine biosphere requires planned operations, and vessels capable of executing them. The vessel designs range from modified fishing- and naval vessels to purpose-built research vessels and ice breakers. Common for all types of research vessels is that they can carry equipment needed for specific operations and surveys. This offers versatility, however demanding areas, volumes and additional equipment at the cost of hull optimisation and cost-efficiency.

The lack of optimised seakeeping capabilities could influence the vessel crew and researchers on board. A vessel experiencing motions above comfort limits over time may have a negative effect of the research being done. To mitigate the risk of seasickness and reduced workability, we can control vessel heading, -speed and time of operation. This does however require information of the operational context, and response models estimating vessel motions based on external loads. Information of the operational context is obtained by weather forecasts, which in many areas are predictions with low certainty. In addition, coastal areas have local variations in bathymetry and weather, which could give large differences of seastates within a forecast area. Moreover, response models describing vessel behaviour are in many cases insufficient to predict true behaviour during operation.

Alternatively, and complementary, to vessel behaviour estimations, we can measure vessel motions directly. Connecting sensors to a vessel allows continuous measurements of velocities and accelerations in 6 degrees of motion (DOF). This forms the basics of a digital twin, allowing a virtual model to represent and predict real behaviour- and state of a vessel, based on a stream of sensor data. Such a digital model could give insight to states of our vessel that are otherwise hidden or not possible to predict by traditional methods.

1.1.1 Digital twins

A digital model that reflects a real asset's behaviour and state is referred to as a digital twin (DT). A digital twin is a virtual model that replicates the state and behaviour of a real asset, such as a ship, a semisubmersible, an offshore wind turbine or a fish farm, in close to real time based on sensor input. This technology could give insight and decision support of physical phenomena that are obscure to the naked eye, offering a high level of control of behaviour of the real asset.

More control of the real asset and information enables smarter systems and autonomy. To commission such systems, they need to operate within safety limits, which requires full control of the system in real-time. To achieve this, digital twin technology is sought to be a key enabler (Danielsen-Haces, 2018), and considering the scalability of the Internet-of-Things (IoT), the development could go rapidly and be cost-efficient.

Installation of sensors and sensor calibration can in some cases be practically difficult or cost inefficient. As an alternative, a digital twin could be designed with virtual sensors, consequently not reliant on sensor data from installed sensors. This is a fully virtual service, where retrofitting sensors and updating configurations are more easily managed. This alternative is viable where the digital representation of the real asset is accurate enough, which, for complex systems, demands extensive modelling and knowledge of the real asset.

1.1.2 Coastal navigation

Coastal navigation is navigation with open sea in one direction and shorelines in the other. It also includes navigation through limited sea space, e.g. fairway navigation. This type of navigation causes a variety of route options, as well as hazards to avoid, such as banks, reefs, skerries and traffic. In addition, coastal waters have environmental hazards causing challenges to navigators, where experience and real-time analysis of bathymetry and metocean conditions are necessary.

Another factor in coastal navigation is the possibility to go inshore or offshore. Inshore navigation often means finer navigation through sheltered waters, with a lower risk of experiencing large vessel motions. Sailing offshore is more exposed to oceanic conditions, thus with a higher risk of large motions and loads. Additionally, sailing offshore is often shorter than inshore, possibly reducing sailing time and fuel costs. These factors raise a set of trade-offs which needs considerations in most types of coastal navigation.

1.2 Problem description

This thesis seeks to investigate how to manage and select route alternatives based on key performance indicators (KPIs) derived from simulated responses of a digital twin framework. The digital twin is the problem's analysis tool, where the end goal is to present the application of a digital twin for decision-support and insight to vessel behaviour and state in operation. The case vessel is NTNU's multipurpose research vessel (RV) Gunnerus. The vessel is built to execute different types of marine operations, thus not optimised for transit and seakeeping. The latter is being analysed for various route alternatives.

The route alternatives are generated by a navigation model based on known waypoints and required fidelity. The problem is a transit problem, where we have a constant start point and a constant destination. Between the known points, a network of routes is exposed to dynamic oceanic conditions, where the routes' performance is derived by a digital twin simulating expected behaviour in 1 DOF along each route. To assess the performance of the navigation model, a fidelity analysis is needed. A route network with low fidelity could provide equal results and insight as a high-fidelity network. Hence, the routing model itself needs to be analysed to give insight to how high-quality routing with a digital navigation model can be achieved.

The thesis seeks to give insight to how digital twins could be used in coastal navigation by answering the following research questions:

1. *What are the limiting factors of using simulation-based engineering in coastal navigation?*

2. *How can we manage fidelity in route network generation and what is the required fidelity to include routing options through limited sea space in a large-scale network?*

3. *How can we utilise digital twin technology to give decision-support in coastal navigation?*

1.3 Thesis structure

Chapter 2: Presents a literature review of central concepts, theories and state-of-the-art to describe the topic and present necessary information needed for the analysis. Additionally, a detailed case description is presented.

Chapter 3: Presents the methods used in the study. An overview of the structure of the research is presented firstly. Following, descriptions of central methods are presented.

Chapter 4: Presents the results of the study.

Chapter 5: Discusses and evaluates the research results and methodologies.

Chapter 6: Concludes the report and recommends further research within the research topic.

Chapter 2

Background

This chapter will present background and theory of the thesis topic. The aim is to elaborate the motivation for this topic, and to present necessary background information which the analysis builds on. A detailed case description is presented lastly.

2.1 Computer aided engineering

Technology is rapidly advancing, where the possibilities of modern ingenuity are basing knowledge and problem-solving on a digital format. Computer aided engineering (CAE) has to a large extent replaced traditional design, solving engineering tasks quicker and more precise than possible before. The field of CAE uses computer software to solve tasks, e.g. simulation, numerical assessments and optimisation. In marine systems, CAE is broadly used, e.g. by computational fluid dynamics (CFD), finite element analysis (FEA) and multibody dynamics (MBD).

2.1.1 Mapping between virtual and physical space

The process of CAE can be regarded as a mapping between virtual and physical space, where the virtual space has no relevance without the physical space. The physical space is complex and dynamic, consisting of people, objects, material, rules and regulations (Zheng, Yang and Cheng, 2019). The virtual space is the digital counterpart of the physical space, where 3D models, load models and virtual assets, e.g. digital twins, exists. The interaction between the two spaces can be regarded as direct- and indirect mapping. Direct mapping is e.g. sensor measurements of a real asset used to represent the state of the virtual asset. Indirect mapping represents the obscure interactions between virtual and physical space, e.g. the hull design of a computer-aided design (CAD) ship model in design stages reflected on the real asset's shape upon construction. A framework regarding a digital twin solution is presented in Figure 2.1 (figure adapted from (Zheng, Yang and Cheng, 2019)).

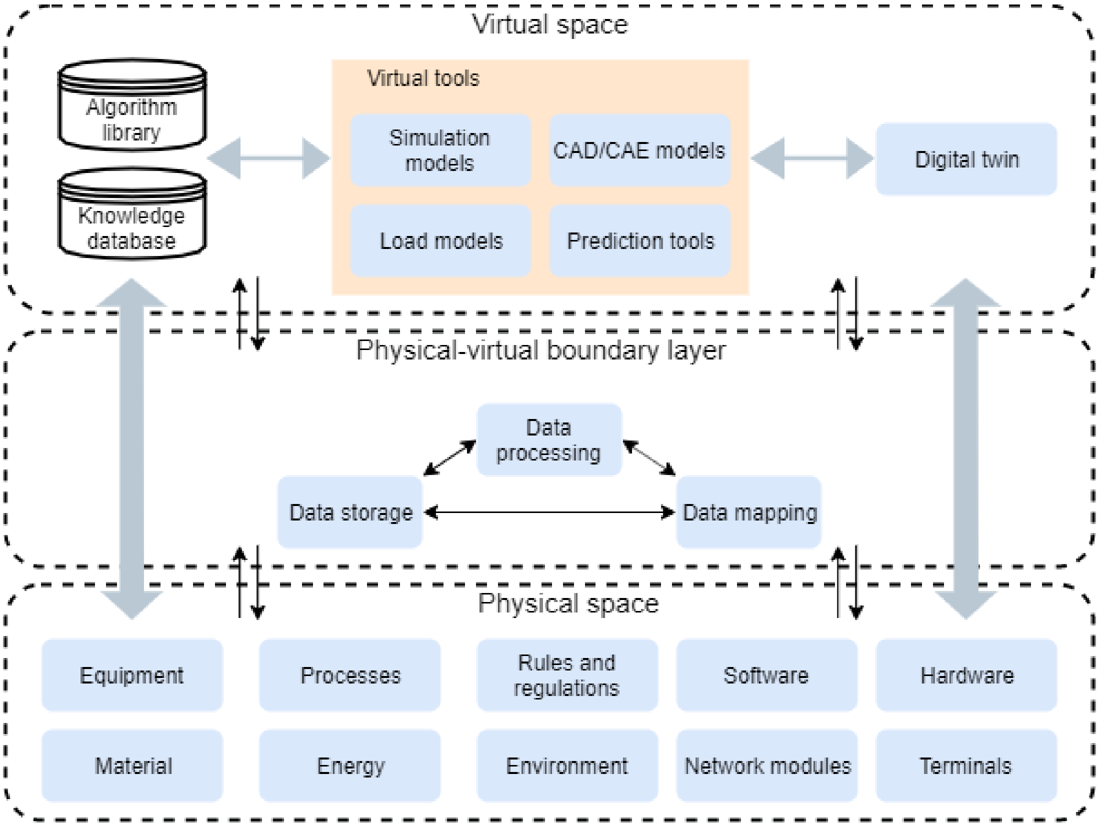

Figure 2.1: Framework of the physical- and virtual space and the mapping between them

Mapping between the two spaces goes through a boundary layer, where both physical and virtual resources exists. In a digital twin solution, this layer consists of data mapping, processing and storage (Zheng, Yang and Cheng, 2019), representing a bidirectional interface between real- and virtual assets. Data mapping is mainly data transformation, -correlation and time-sequence analyses, i.e. relating real and virtual data. Data processing is cleaning, structuring and clustering of data. Data storage is storage of data from both spaces, i.e. data of real equipment, -material and -environment, and virtual simulation data, evaluation- and prediction data (Zheng, Yang and Cheng, 2019).

System design by mapping

Design mapping is related to the mapping between the virtual and physical space in design of virtual services. When designing a system, we have three governing domains, namely Stakeholder Requirements domain, Functional Architecture domain, and Physical Architecture domain. The stakeholder requirements (SR) are stated early in a design process where high-level requirements of the design are obtained. Thereafter, we obtain functional requirements (FR) in the functional architecture domain which correspond to the stakeholder requirements. Functional requirements are a discretisation of system behaviour and function.

Lastly, design parameters (DP) are found in the physical architecture domain that corresponds to the functional requirements.

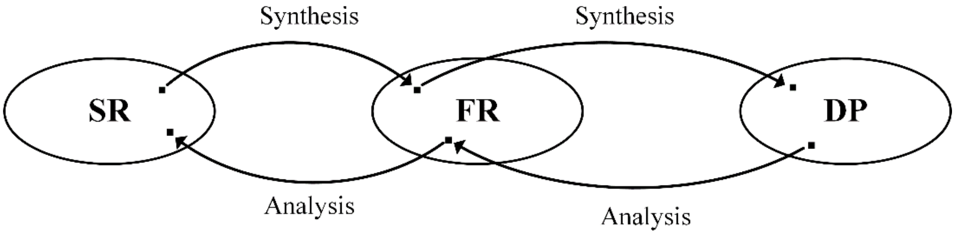

Figure 2.2: Design mapping

The procedure is an iterative mapping of synthesis and analysis between spaces (Farid and Suh, 2016), as shown in Figure 2.2. Synthesis is where a designer or engineer seeks to find a solution to the questions “what needs to be achieved” and “how can we achieve it” (Suh, 2001). Analysis is where a designer or engineer reviews if the found solution satisfies its initial requirements.

2.1.2 Dependencies and complexity

Marine systems are tightly coupled complex systems, where boundaries and interactions often are obscure and interconnected due functional dependencies and restricted volumes. To represent the system state and behaviour, we therefore need approaches to discretise and describe the system. Farid and Suh (2016) presents a set of axioms for good design practice, where the independence- and information axioms are presented here.

Firstly, the independence axiom states that a set of functional requirements (FR) should be mutually exclusive and collectively exhaustive (Farid and Suh, 2016, p. 26). Meaning, when we design a system, we should strive for functional independence within the system we are designing. If we have a system with two mutually exclusive functional requirements, as seen in Equation (2.1), the system is completely uncoupled. Moreover, if we change a functional requirement, the change should correspond 1-to-1 with a change in the same unique design parameter. This is best practice design, however for marine systems, uncoupled functional requirements are practically impossible. Nonetheless, system design should aim to achieve as uncoupled functional requirements as practically possible.

$$\begin{bmatrix} FR_1 \\ FR_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \end{bmatrix} \tag{2.1}$$

Secondly, the information axiom states that by minimising information (I) needed to explain a design, you maximise the probability of fulfilling the functional requirement $(P(FR_i))$ (Farid and Suh, 2016, p. 28), see Equation (2.2). This axiom regard simplicity in design, where good practice is to minimise complexity.

$$I = -\sum_{i=1}^{N} \log_2 P(FR_i) \tag{2.2}$$

2.1.3 Simulation

A simulation is an imitation of processes or systems over time. The concept is broadly used in CAE, offering the possibility to test performance and responses of a system or process to e.g. changes in the functional requirements. A simulation represents a system operation or process in time, whereas a simulation model represents the system or process. For marine systems, simulation have numerous applications, from fleet scheduling to offshore training or drilling processes.

In addition to simulation of existing assets in operation, simulation can be used in design phases. Simulation-based design (SBD) is design by using simulation-software to imitate expected performance throughout the lifecycle of an asset. Throughout the lifecycle of marine assets, roughly 20-50 years, market volatility and changes in the operational context are inevitable. To design more robust marine systems, dynamic requirements needs to be taken into consideration to meet periods of e.g. low demand or new regulations governing the operation.

A clear advantage of simulation is that high quality results can be achieved with low effort, given that an accurate enough simulation model can be constructed for our system. For example, if we are changing the traffic pattern of a road to clear bus lanes, simulation lets us test numerous patterns based on theoretical models, without the need to build and physically test every pattern. Limitations to simulation is that some systems or processes are hard to model due to the nature of the system or limitations of the simulation software. Additionally, model verification is often based on increasing trust in the model, rather than absolute accuracy. Thus, without proper information and knowledge about the real system, deviations and uncertainties of the simulation model are expected.

2.2 The digital twin

A digital twin is a virtual representation of a real asset, e.g. a ship, platform or automobile, capable of rendering the state and behaviour of the real asset in real time. The digital twin concept has been around from the first CAD models, however with a substantially different application and definition. These models were static and descriptive, and it has not been until the last decade that the dynamic digital twin has gained foothold as a virtual entity on its own.

The term digital twin can be interpreted in two ways. At one hand, there are dizygotic twins, where the difference between real asset and twin is too large to have any practical information of how the real asset affects the twin. On the other hand, monozygotic twins have a clear resemblance. The twins have identical initial conditions, though free to evolve separately. The evolution of both twins will follow the same set of initial conditions, however how the two individuals reacts and makes conscious choices is up to the individual. Thus, a 1-to-1 digital twin resemblance is only present at the initial condition.

Therefore, the concept of the digital twin has some apparent logical issues. To avoid the logical flaws, the digital twin can rather be explained as a digital shadow or mirror image to the real asset. A shadow or a mirror image will mimic both physical state and -behaviour to the real asset, without any conscious actions. Thus, the digital representation is a unique entity without the choice to alter its own state differently from the state of the real asset.

2.2.1 Characteristics of the digital twin

To further converge towards a common understanding of what the digital twin is, Erikstad (2017) proposes 5 essential characteristics. These are summarised as:

- *Identity*: The digital twin gains identity by connecting to a real and unique physical asset. Ideally, there is a 1-to-1 cardinality between real- and virtual asset, however a 1-to-N cardinality is often more practical for complex systems. For such relationships, multiple digital twins covering subsets or parts of the real asset's state, behaviour and physics are connected to form a close-to complete representation of the real asset.
- *Representation*: The digital twin is a representation of the real asset, where physical properties are replicated in a digital format, e.g. as CAD models.

- *State*: The digital twin renders quantifiable measures of the state of the real asset in close-to real time.
- *Behaviour*: The digital twin mimics responses from external stimuli in an operational context, e.g. forces from wind and waves.
- *Context*: The digital twin operates in the same operational context as the real asset. This can be e.g. waves, wind, current.

These characteristics explains the basic requirements of a digital twin. A key point here is that the digital twin is an entity on its own, however only useful when well-connected to the real asset in close-to real time. Literally interpreted, the digital twin is not a digital twin until it receives real observations of characteristic of a unique and real asset.

Luckily, there is a loophole in the interpretation by emphasising that the digital twin is an entity on its own. The real asset would therefore reflect the digital twin, rather than the other way around. The real asset would in this case be designed with a hardware-in-the-loop (HIL) configuration, where the digital twin is developed and continuously updated throughout the life cycle of the real asset. This is beneficial for design phases, operational phases and decommissioning. In addition, such a digital twin will have full control of alterations and manipulations done to the real asset, reducing the risk of “resemblance drift”, i.e. non-updated digital twins. Furthermore, this approach forms the basis for smart digital twins, where modifications and maintenance to the real asset would be done based on reflected and predicted behaviour by the digital twin.

2.2.2 Design patterns

Digital twin technology is one possible solution to make use of the IoT and big data to make smarter systems. However, connecting sensors and endpoints to physical assets does not add artificial intelligence or make it any smarter, it merely adds streams of data (Datta, 2017). If this (big) data is not managed, the only thing you will have is a large quantity of measurements losing relevance by the minute, taking up place in a hard drive somewhere never to be found. Therefore, we need some way to connect the endless stream of sensor data to the digital twin. This is where design patterns come in to play. Design patterns are in general reusable patterns to solve programming architecture.

As earlier stated, pragmatic considerations may imply a 1-to-N cardinality between real- and virtual asset. The building blocks of the digital twin will therefore consist of multiple connected DTs that in total forms a larger structure. This is the concerns of structural patterns, governing the composition of interfaces (structural class patterns) and dynamic functionality (structural object patterns) (Gamma *et al.*, 1994). Class patterns can then be used to make the N twins work together to form a working model. Object patterns allow the implementation of dynamic compositions, which can support integration of machine learning and artificial intelligence.

Erikstad (2018) proposes six structural patterns for a digital twin; product data model (PDM) twin, big data twin, baseline twin, load-based twin, benchmark twin and the ML proxy twin (Erikstad, 2018). Of these patterns, the baseline-, load-based- and benchmark twins are of interest to this thesis. The following paragraphs will briefly present the three patterns, where abbreviations in figures are: Real Asset (RA), Sensor measurements (S), Digital Twin (DT) and Operational Context (OpCtx).

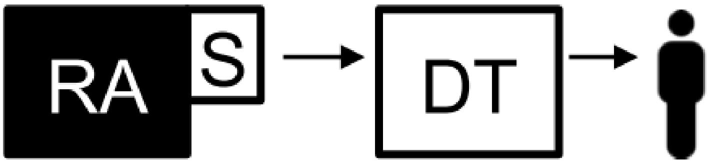

Figure 2.3: The baseline digital twin (Erikstad, 2018, p. 359)

The baseline twin is the fundamental pattern for a digital twin where a real asset with sensors connects measurements to physics-based analyses. The digital twin replicates behaviour and state based on real sensor data, which is the basic requirement of a digital twin. The pattern is applicable for most real assets. However, it does require that sensor placement is possible on our real asset and that the responses are possible to replicate. In addition, the data needs a required fidelity without causing high computational latency, potentially compromising the real-time requirement. This pattern forms the core of the digital twin, where other patterns could be added to lower computational time and provide better insight. Furthermore, the replicated responses derived from the digital twin can be used for further analyses to detect deviations and continuously improve the digital twin.

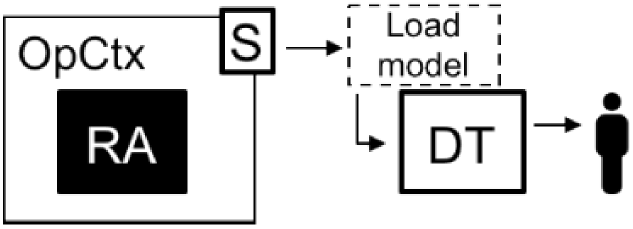

Figure 2.4: The load-based twin (Erikstad, 2018, p. 360)

The load-based twin is where measurements of the operating context are used in load models to simulate asset state, thus indirectly replicating the state of the real asset. The real asset does not need to have sensors equipped, which in some cases can be costly and practically challenging. On the other hand, we need measurements from the operational context which have both a required accuracy and a required timeframe. Additionally, this pattern relies heavily on load models that replicate the state and behaviour of the real asset, which may be challenging when modelling the complexity of ship and its dynamic environments. As such, anomaly detection and deep insight might be insufficient. Key advantages of this pattern are that it is applicable for a wide range of assets and can provide decision support to more than one asset. Consequently, such a twin could be used in fleet management, given a homogenous fleet and known operational contexts for each fleet member.

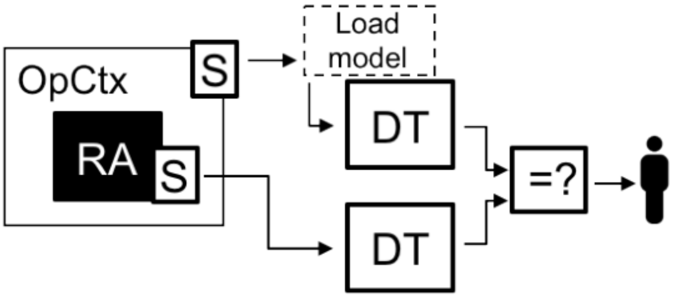

Figure 2.5: The benchmark twin (Erikstad, 2018, p. 361)

The benchmark twin is a pattern where both the baseline- and load-based twin are running in parallel. Here, we have two models for comparison, which can combine the strengths of both patterns, possibly granting deep insight to our load models, digital models and the state of the real asset. Disadvantages of this pattern is the cost of running two twins simultaneously. Additionally, running two twins could be computationally heavy, potentially scrutinising the value of both twins by latency challenges. Moreover, this pattern does not directly answer what observations are correct, thus supplementary patterns and models are needed to complete the benchmark twin.

2.2.3 Application and generating value from a digital twin

To justify the investment- and running costs of a digital twin, it must add a value to the real asset. The value of a digital twin is information and insight of processes, interactions and responses that are hard to observe, model and predict on the real asset. This information is a trade-off between costs and quality. The lowest level of added insight is observations of the state and the highest level of insight is when the digital twin predicts and decides possible outcomes with high certainty. Erikstad (2019) suggests a digital service maturity index ranging from 1 to 5 to evaluate the application and value of a digital service, see Figure 2.6.

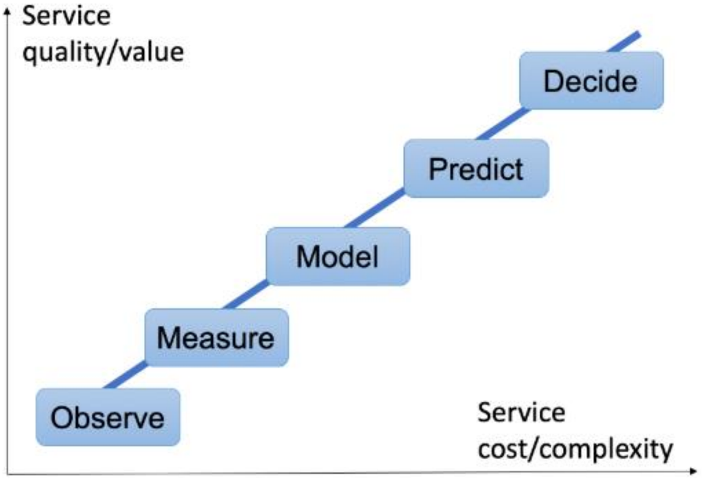

Figure 2.6: Digital service trade-offs highlighting the maturity index from observations (1) to decisions (5) (Erikstad, 2019, p. 464)

The application of digital twins for marine systems varies based on the requirements of the system for which we are applying it. Static requirements are stated early on in development phases of a marine system, whereas dynamic requirements change due to e.g. market volatility and regulations throughout the lifetime of the system. Static requirements form the foundations of the system we are designing, where tasks, capacities and operational contexts are stated. Dynamic requirements change with varying operational context, which is expected for marine systems with a lifetime well over 20 years. Applying digital twins to these systems could give better insight to how these requirements could be met, by simulating new operational contexts and evaluate the responses on the digital twin, rather than the real system which is often costly and complex. Ultimately, the digital twin can predict future states of the real asset, thus granting decision support in short- and long-term planning.

To evaluate the value of a digital twin service, Erikstad (2019) suggests a set of factors for consideration:

- What is the overall goal of the service, i.e. what does the digital twin seek to improve?
- Who are the primary users?
- What is the scope of the service, i.e. how should the digital twin be used?
- What is the quality of the service?

This list gives a preliminary set of basic needs to address. As an example, let us consider a crew transfer vessel used for crew transportation from shore to offshore wind farms. The requirement of the vessel is to transport people within a certain time frame. A requirement of a digital service is e.g. to offer real-time support of optimal speed based on the required freight rate while minimising fuel consumption. Thus, insight to passenger comfort levels or long-term decision-support may be secondary. By evaluating the requirements of the real service, we can therefore derive a set of needs for the digital twin while excluding information the users does not directly benefit from.

2.2.4 Current state of digital twin technology

Digital twin technology is still in its infancy. There has been an increase in literature of digital twin technology in recent years. Wang, Lee and Hsu (2020) reviews the state-of-the-technology by patent analysis, concluding that the technology lacks common understanding and that most patents regard the manufacturing industry. The industry incentive is creation of DTs for factory sub-systems to achieve more automated manufacturing. Furthermore, Wang, Lee and Hsu (2020) presents suitable definitions for types of digital models which are summarised below to clarify the semantics.

- *Digital twin*: A virtual object totally integrated with a physical object by two-way data flow of the status of each object.
- *Digital shadow*: A virtual object receiving automatic one-way data flow of the status of the physical object.
- *Digital model*: A virtual object that is a digital representation of the physical object without automatic data exchange.

Digital twin development for marine systems is within research phases, where full-scale prototyping could be expected in short-term. DNV GL is prototyping their ReVolt scale model ship, where a digital twin solution for several parts of the ship has been developed (Danielsen-Haces, 2018). The focus of their development has been user interfaces, engine monitoring and data collection and -storage. Coraddu *et al.* (2019) constructed a conceptual digital twin using on-board sensors to estimate speed loss caused by marine fouling. Their findings concluded that using a digital twin solution gave better and more accurate predictions of speed loss than the corresponding state-of-the-art method.

Furthermore, Johansen and Nejad (2019) presents a digital twin approach to monitor drivetrains in marine applications. The research regarded fidelity and accuracy of digital twins, where they concluded that low fidelity gave high-accuracy results, however, did not meet the requirements of a digital twin. By increasing the fidelity, accuracy challenges became apparent, even though the solution could be regarded as a digital twin.

A field where digital twin technology is being tested for full-scale marine systems, is offshore wind. General Electric are using digital twins to estimate internal wind turbine temperature in real-time (Pomerantz, 2018). A key feature of their development is the use of virtual sensors to estimate the state of the real asset. The virtual sensors and digital twin allow high-accuracy estimations of measures that otherwise are hard to predict by traditional methods.

2.3 The digital platform

The digital platform is a virtual environment where the relevant virtual resources of the real asset exists. This is the virtual space where data, CAD models, product data models (PDM) digital twin(s) and more are stored, managed and used. The main components of the digital platform are data storage, models, user interface (UI) and what we can denote the “junction”. The latter is the core interface of the virtual platform, where all models, data, inputs and outputs communicate. The junction’s main objective is to structure and relate processes and interactions efficiently while minimising complexity and computational time.

The platform architecture is presented in Figure 2.7. The virtual space interacts with the environment where inputs from real asset sensor data transcend to the could data storage. The outputs of the digital platform interact with the physical space by a user interface. This interface is modelled monodirectional in the proposed framework, however with a low level of platform autonomy, this interaction goes both ways. Meaning, the operator should be able to obtain information based on his or her own desires, rather than just inputs from real- and virtual asset measurements. For example, an operator could manually state operational contexts used in a simulation model to see the predicted behaviour of the digital twin. In this case, the state and responses are computed based on an active decision done by the operator, suggesting a bidirectional interface.

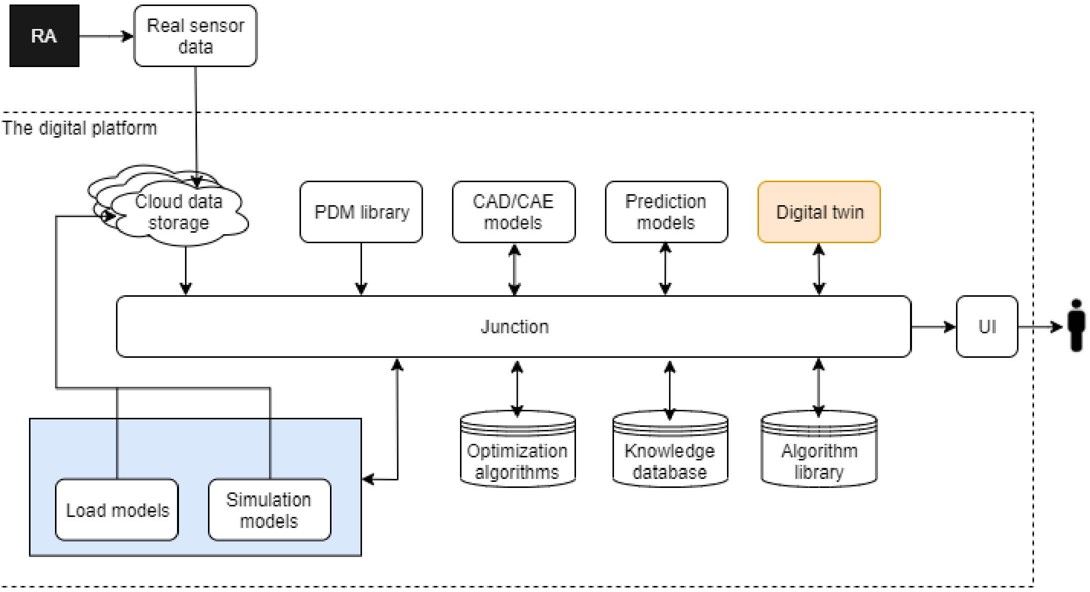

Figure 2.7: Digital platform architecture

2.3.1 Data storage and -management

As the concept of the digital twin relies on measurements, an auxiliary, but important, part of the digital twin is how to manage big volume data. Data storage and -management is included in the digital platform, and following, the virtual space, by regarding cloud data storage units as fully virtual entities. Local data management and -storage is thus regarded as entities in the physical-virtual boundary layer.

A key requirement of the data recorded by the real asset is that it must represent the correct physical conditions and loads we are investigating. In addition, the storage should have volume- and time control to avoid storage units to fill beyond limits and potential clogging of measurements that do not represents the real asset in real-time. Historical measurements are of importance when analysing behaviour over time, where e.g. pattern recognition algorithms can simulate and potentially predict future states.

Moreover, where and when the data is stored is of importance. A solution that is pointed at multiple times is cloud storage, where measurements are continuously uploaded to a remote storage unit. This requires an internet connection and broadband bandwidth that satisfies the required transfer rate. This may not be feasible in short-term for ocean-going ships frequently measuring multiple 6 DOF sensors. This could be solved by having local storage and periodical uploads to a cloud. When a ship enters waters with higher broadband capacities, the data could be uploaded from local storage to the cloud, thus resetting the local storage unit. If the digital twin is to replicate real-time state with local short-term storage this requires that the digital twin service is stored and ran on board.

Ultimately, this depends on the required measurement fidelity and -interval. High frequency and -fidelity measurements give large quantities of real-time data which in many cases are not necessary. This should be taken into consideration when creating the digital twin, where latency could diminish the value of real-time information, as the replicated state is no longer real-time. Oppositely, by decreasing the measurement frequency, deviations and extreme condition measurements may be missed out or misinterpreted. Meaning, if the measurement interval is too small to capture the real loads, state and behaviour; the virtual state and -behaviour are falsely replicated.

2.3.2 Model libraries

The model libraries included in the platform are product data models (PDM), computer-aided design/-engineering (CAD/CAE) models, prediction models, load models and simulation models. PDM and CAD models are 3D models and information about the processes and equipment of the system. These libraries are not directly linked to the digital twin, however used to observe the system, useful for e.g. design phases, maintenance and configuration management. The libraries serve as catalogues for the operator, owner or designer where relevant specifications, configurations and requirements of the real asset is stored and continuously updated.

CAE models are analysis models for the virtual asset. These are 3D models and computational tools, e.g. Ansys and ShipX, used to analyse responses and performance of the virtual asset. The CAE model library has a correlation to the load models, where the load models often are used together with CAE software. Load models are physics-based estimations and predictions of real behaviour, e.g. responses in 6 DOF of a ship in operation with incident waves, current and wind.

Simulation models are also related to load models and CAE models. These models differ from the static analyses of CAE by using simulation software to analyse dynamic behaviour and responses over time. A notable feature is the possibility to include co-simulation, where multiple actors can provide various simulation models to the digital platform. This development would benefit operators, designers and researchers, where everyone can contribute to a comprehensive simulation platform. Creating models in accordance with the functional mock-up interface (FMI) would also allow easy implantation of functional mock-up units (FMUs). For such platforms, designers, suppliers and manufacturers can provide FMUs of subsystems to the simulation platform. The platform operator will not have any information of internal processes, dependencies and interactions within the different FMUs, only what inputs to give and what outputs are possible to obtain.

The algorithm library consists of e.g. pattern recognition algorithms that can be used to generate insight and knowledge. Erikstad (2018) presents three insight patterns for a digital twin solution: anomaly patterns, root cause patterns and foresight. Anomaly patterns use time series to detect anomaly behaviour, e.g. deviations that is hidden for the naked eye. Root cause patterns are causal analyses that give insight to causal relationships of e.g. events, objects and processes. Foresight is the pattern enabling prediction of a digital twin. With growing technology maturity, the list of patterns is expected to expand to include more patterns.

Prediction models are related to the algorithm library and knowledge database. These are physics- and algorithm-based models to predict future states and behaviour on the real asset based on historical data. This is digital service maturity level 4, where the digital twin could predict possible outcomes with a high level of certainty.

2.3.3 Current state of digital platforms and -services

Digital platforms for marine systems would benefit from model sharing and common interfaces. Marine systems are complex, which means describing and modelling them demands extensive efforts and information. If sub-system suppliers could distribute FMUs of their respective systems to a common platform, the need for “scattered” modelling efforts by the industry could reduce drastically. However, marine assets are high-value assets, which can explain the need to protect business insight and designs.

There are however joint industry projects (JIP) moving towards co-simulation, resource sharing and common platforms. The Open Simulation Platform (OSP) was a JIP that aimed to create an open source platform using the FMI standard (OSP, 2020). The platform has been central in projects related to digital twin development and has shown the strengths of model sharing to accelerate innovation in software-based marine engineering. Moreover, DNV GL (a partner in the OSP JIP) has developed and launched their platform Veracity (DNV GL, 2020). The platform acts as a hub for applications, and data storage and -management, as well as a common interface for industry partners.

A noteworthy development in digital platforms is Ansys' newly launched twin builder (Ansys, 2020). The platform allows bottom-up creation of digital twins, as well as product libraries of reusable existing components. In addition, there are solver features and HIL-integration which gives designers, manufacturers and operators a powerful platform for digital twin development in an environment and Ansys interface many engineers are familiar with.

2.4 Coastal navigation

Coastal navigation is navigation where there generally is open sea in one direction and shore in the other. Coastal navigation also includes navigation through limited sea space, e.g. fairway navigation. This type of navigation causes a variety of route options, as well as hazards to avoid, such as banks, reefs, skerries and traffic. In addition, coastal waters have environmental hazards as fog, light, tidal variation of sea elevation, tide, current, wind and waves. These factors raise challenges to navigators, where analysis of coastal bathymetry and metocean conditions in close-to real time is necessary.

Modern navigation with electronic navigational aids has replaced printed maps and charts. Instead of navigating by manual special reference, the position of the vessel is given by global positioning system (GPS) updates, minimising navigation errors and mitigating human errors. Some systems used in coastal navigation are: GPS, electronic chart display and information systems (ECDIS), electronic nautical charts (ENC), automatic identification systems (AIS) and route plotting systems.

These systems allow better communication between vessels and ports within coastal waters and route planning based on vessel movement and metocean conditions. Electronic navigational aids also store completed voyages by the vessel and in many cases also includes recommended voyages for fine navigation, e.g. fairway navigation through the Solent outside Southampton, UK. AIS, and in some cases radar, is useful for navigation in traffic, where the heading, speed and whereabouts of other vessels are valuable information to avoid collision and contact.

Another factor in coastal navigation is the possibility to go inshore or offshore. Going inshore often means finer navigation, requiring an active bridge crew aware of present hazards. Routing inshore is sheltered, thus reducing the risk of encountering large waves and wind. Sailing offshore is more exposed to oceanic conditions and loads, thus larger ship responses. Navigating offshore does not require the crew to be as aware as inshore, due to the larger sea space available for navigation. When sailing along the coast of e.g. Norway, sailing offshore is often shorter than sailing inshore, reducing “detours” into sheltered fjords and sailing around islands and skerries.

2.4.1 Route plotting

Route plotting is done either by manual sketching or selection of previously used routes. As mentioned, some coastal waters also have recommended passages for e.g. port, fairway and canal passages to control traffic and reduce the risk of collision, contact, grounding and stranding. Prior to a voyage, a captain and/or navigator plots the planned route and either manually steer the vessel along the route or let an autopilot system move along it. A typical user interface (UI) can be seen in Figure 2.8.

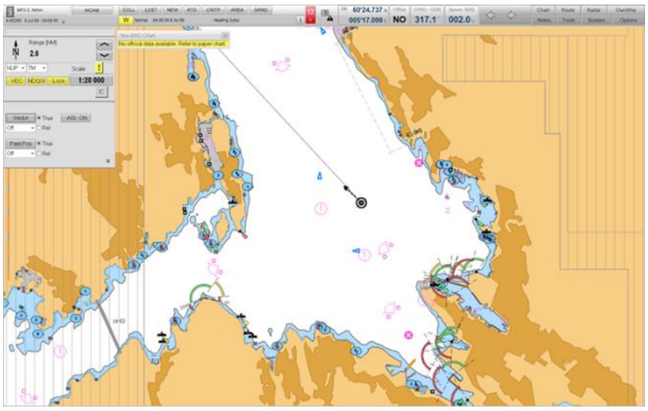

Figure 2.8: Typical ECDIS display in coastal navigation (Kongsberg Maritime, 2020).

2.4.2 Route modelling

Ship routing and route modelling in literature generally concerns network design, where the focus areas are what ports to visit, when and how often to visit, and optimal size and speed of ships and fleets. Within network design, set partitioning (SP) is frequently used (Christiansen, Fagerholt and Ronen, 2004), which is a methodology of interest to this project. This thesis investigates routing between two known points and can be regarded as a liner problem where SP models could be applicable.

Sandvik (2019) presents a simulation-based sea passage scenario model to optimise a future voyage based on occurring weather conditions. The model generates route alternatives based on vessel heading relative to the shortest path between an origin and a destination, and a fixed time frame of each leg. After sailing a leg, i.e. when we get a new weather forecast, a new vessel heading and speed is calculated based on an optimal route emphasising two decision criteria, namely fuel consumption and time delay. The research highlighted the importance of fidelity when generating routes and discusses the advantages of simulation-based route modelling.

2.5 Case description

The goal of this thesis is to give insight and decision support in route planning based on simulated behaviour of a digital twin. To do so, a route model and a digital platform is proposed for further use. The digital twin acts as the analysis tool to make better decisions, simplified to show the application of digital twins in marine operations. This section describes the case vessel and -mission, as well as detailed descriptions of requirements of the digital service and a proposed value of the service.

2.5.1 Vessel particulars

The case vessel is the multi-purpose research vessel Gunnerus, shown in Figure 2.9. The vessel is owned and operated by NTNU, providing a platform for scientists and students from multiple disciplines.

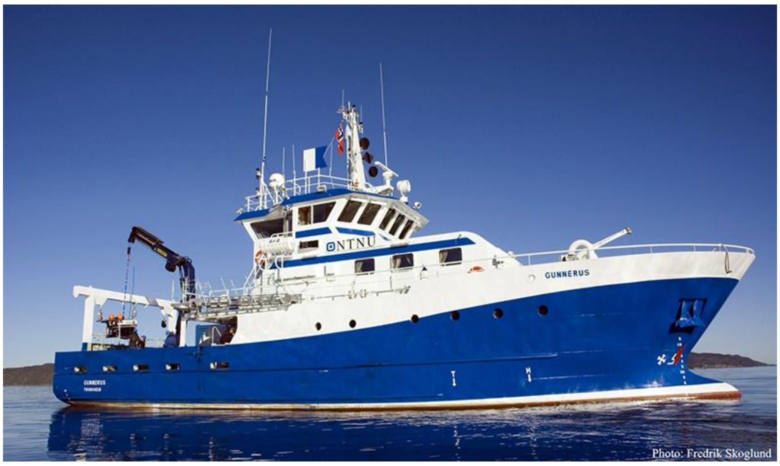

Figure 2.9: RV Gunnerus. Photo: Fredrik Skoglund (NTNU, 2019)

The vessel has been involved in a NTNU-based student summer project in 2018, where foundations of a digital twin have been created (Bjørum *et al.*, 2018). The project resulted in a detailed CAD model of the vessel, digital twin with sensor data and -information, and PDM libraries. The vessel particulars are summarised in the table below.

Table 2.1: RV Gunnerus particulars

Particular	**Abbreviation**	**Value**	**Unit**
Length between pp	Lpp	33.9	m
Breadth	B	9.6	m
Draught	T	2.8	m
Displacement	Δ	574.0	tons
Block coefficient	C_b	0.62	-
Cruise/max speed	$V/V_{100\%MCR}$	9.4/12.6	kn

RV Gunnerus is capable of numerous marine operations. It carries winches for trawl experiments, multibeam-echosounder (MBES), crane for remotely operated vehicle (ROV) and autonomous underwater vehicle (AUV) deployment, conductivity, temperature, depth (CTD) sonde winch, and more. In addition, there are multiple lab areas and extra accommodation for researchers. These capabilities offer versatility, presumably at the cost of seakeeping capabilities and optimal hull. Moreover, to offer even more versatility, the vessel had its hull extended in the longitudinal direction by approximately 5 m. This expansion could have had a negative effect on seakeeping capabilities.

2.5.2 Mission statement

The case mission is a transit from Trondheim to the cold-water reef outside Træna. Træna is a remote island in the Norwegian sea. The island is famous for rich birdlife and cold-water reefs. In the case, scientists from NTNU are sailing to Træna to conduct oceanographic surveys of the coral reefs to map the biodiversity and health of the reef. The surveys are of interest to the government of Norway, as they have been and are looking for exploration drill opportunities for offshore oil and gas production near the reef.

The voyage takes approximately 20-30 hours by sailing offshore in calm seas at a speed between 8-10 knots. The optimal route, according to the voyage planner "searoutes.com", is shown in Figure 2.10. Sailing this route at a constant speed of 9 knots gives a voyage duration of approximately 24 hours.

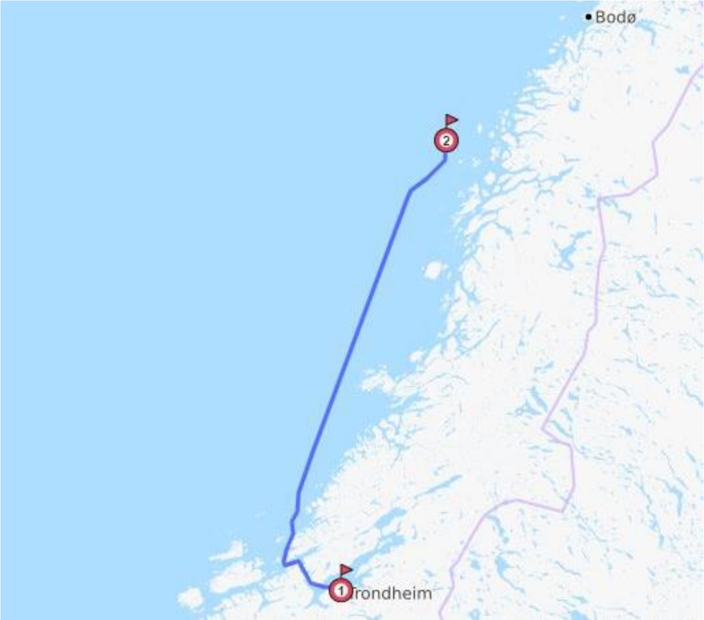

Figure 2.10: Optimal voyage between Trondheim and Træna according to searoutes.com. Source: snapshot from searoutes.com UI (Searoutes, 2020)

The crew of RV Gunnerus are not convinced that the proposed route from searoutes.com is a suitable voyage. The vessel will carry a full set of researchers and expensive equipment for the surveys. They are concerned that sailing offshore could induce high risks to assets, crew and passengers given the elusive weather conditions of the Norwegian sea. Therefore, they propose that other route alternatives should be investigated by simulating expected vessel behaviour when encountering predicted weather conditions. The value proposition of the mission is formulated as follows:

RV Gunnerus is in need of a routing model that can predict expected vessel behaviour of multiple route alternatives between a given origin and destination. The model should give insight and decision-support of optimal route alternatives in close-to real-time by using weather forecasts. Optimal routes should be derived by a set of relevant KPIs.

2.5.3 Mission breakdown

To design and create a system that satisfies the value proposition, we use the mapping procedure presented in Section 2.1.1. Firstly, we extract a set of stakeholder requirements from the mission statement and value proposition. Thereafter, we find functional requirements based on the stakeholder requirements. Finally, we obtain design parameters which are concrete parameters to solve the task. The following subsections presents the mission breakdown.

Stakeholder requirements

The stakeholder requirements are first and foremost within the operational domain, where owner, crew and passengers require a model to operate with a higher level of safety to people on board (POB) and assets. In addition to safety, the service could also strengthen comfort and possibly the performance of the POB, which is beneficial to the research vessel.

Secondary needs of the service are environmental, commercial and technical. By smarter routing, we can assume that the vessel could sail more environmentally friendly by mitigating the risks of encountering harsh oceanic conditions which often means significantly higher fuel consumption. The commercial and technical aspects concern the expectations that novel engineering should give interest and publicity in academic circles and industry. This would be beneficial to an academic institution such as NTNU, where possible research grants could be given for further research within the field.

Functional requirements

From the stakeholder requirements, we can list a set of functional requirements of the service. The functional requirements are high-level design constraints, functional- and behavioural expectations to concretise how to satisfy the stakeholder needs. The main functional requirements of the service are: response prediction, route generation, model maintenance and end use. A functional hierarchy is presented in Figure 2.11.

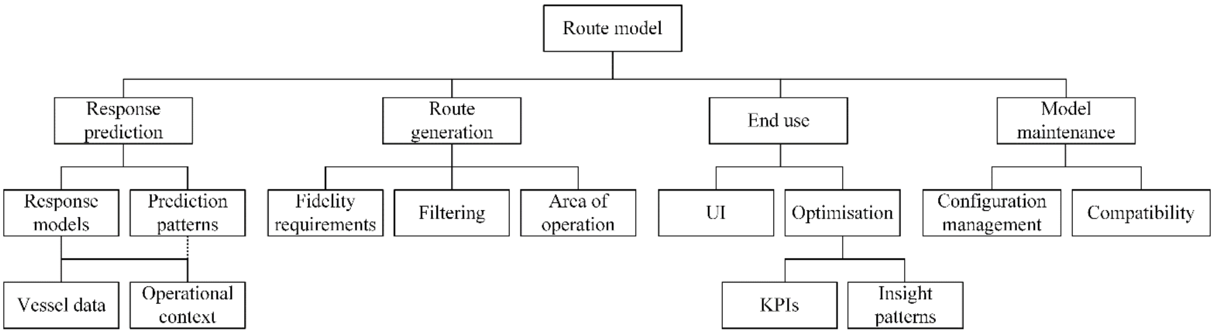

Figure 2.11: Functional hierarchy of the route model and -service

Response prediction have two sub-functions: Response models and prediction patterns. Response models are mathematical models to estimate vessel response based on simulated operational context of a vessel. Prediction patterns are algorithm-based patterns that relates previous operational contexts to previous responses. Route generation concerns route alternatives, dependent on the sub-functions: fidelity management, filtering and operational context (area of operation).

The latter two main requirements are concerned with the end use and maintenance of the model. The mission statement indicates that the route model should be reusable, as *the vessel* is in need of a route model. Therefore, maintenance is needed with the two branching sub-functions of configuration management and compatibility. This is to ensure that software updates and hardware replacement does not alter the functionality of the service. The end use functions are UI and optimisation. A clear and informative UI is key to give insight and decision-support to make better decisions. Optimisation is the actual insight-generation of the service, where the end user can obtain optimised routes based on his/her desires. This can be achieved by insight patterns, e.g. algorithms, and a set of KPIs that is relevant to make better decisions.

Design parameters

In the functional hierarchy we have 4 main groups of functional requirements. The corresponding design parameters are proposed in the following paragraphs.

Firstly, we consider response prediction, dependent on response models and prediction patterns. To calculate responses of a vessel we need vessel data of physical particulars, transfer functions, vessel behaviour models and an operational context. Vessel data are e.g. geometric models and loading conditions, which exists for RV Gunnerus from various projects, including Bjørum *et al.* (2018). We have mathematical representations of vessel behaviour, given by transfer functions and response models. The operational context can be obtained by on-site measurements of the operational context, or by hindsight and forecast weather data. Prediction patterns exists, however for a digital service in early development, simulations based on physics are seemingly a preferable choice.

Moving on to route generation. Fidelity management needs to be assessed for the route model, which means there has to be a set of route variables that govern the fidelity. As briefly discussed in Section 2.4.2, there are methods to generate a route network, and SP seems fitting for coastal navigation. The design variables in route generation by SP could be network extent and number of subsets. Filtering routes can be done by checking feasibility in physical terms, i.e. it is not allowed to sail across land and generated routes should not converge towards areas with no way out, e.g. fjords. The firmer can be filtered by pixel analyses and the latter by node parental analysis. This depends on an area of operation, that in practice is stated by the end user of the service, i.e. where are we sailing. To analyse route alternative, rasters over the relevant area can be efficiently extracted, which also allows pixel analyses.

Thirdly, we consider model maintenance, dependent on the two sub-functions: configuration management (CM) and compatibility. These are somewhat complementary, however are two distinct functional requirements needed to maintain the service. CM is the field within engineering that concerns management of updates, interfaces, legacy systems and communication across different configurations of a system. Compatibility is the functional requirement that states that the service needs to be compatible with the systems which the end user are using. To achieve these two FRs, we have knowledge of CM and application programming interfaces (API). For the route service, CM should be used to track system changes and to keep it up-to-date. The service should be created according to standard API and use the FMI standard for its components. In addition to maintenance, the service requires some security, however this is not included in this assessment.

Lastly, we have end use. This is probably the most important value-generating functional requirement for the service, as the purpose of the service is to give insight and decision-support to a specific problem. To achieve this, the UI and results should be simple to understand in real-time. Plotting and printing optimal route alternatives are quite unmistakable given a solid UI, where standardised icons and symbols should be included, e.g. DNV GL's Veracity or Openbridge (Oslo School of Architecture and Design, 2020). In addition to UI, we have optimisation. Optimisation can be done by mathematical approaches, e.g. convex programming or multi-objective optimisation. For route problems, we have well-known algorithms such as Dijkstra's shortest path algorithm. This algorithm concerns shortest path only, which is included in the set of KPIs we are assessing.

As the value proposition states that the optimal routes should be derived from a set of KPIs, we need to concretise them. The stakeholders requests route alternatives in plural, i.e. the UI should present different optimal routes based on different KPIs. This is much like Google Maps when driving a car, where you can enter an origin and a destination, and you will see multiple route alternatives based on different criteria. In transport by car, the KPIs are e.g. shortest distance, estimated time, avoid highways, -tolling stations or ferries, see Figure 2.12.

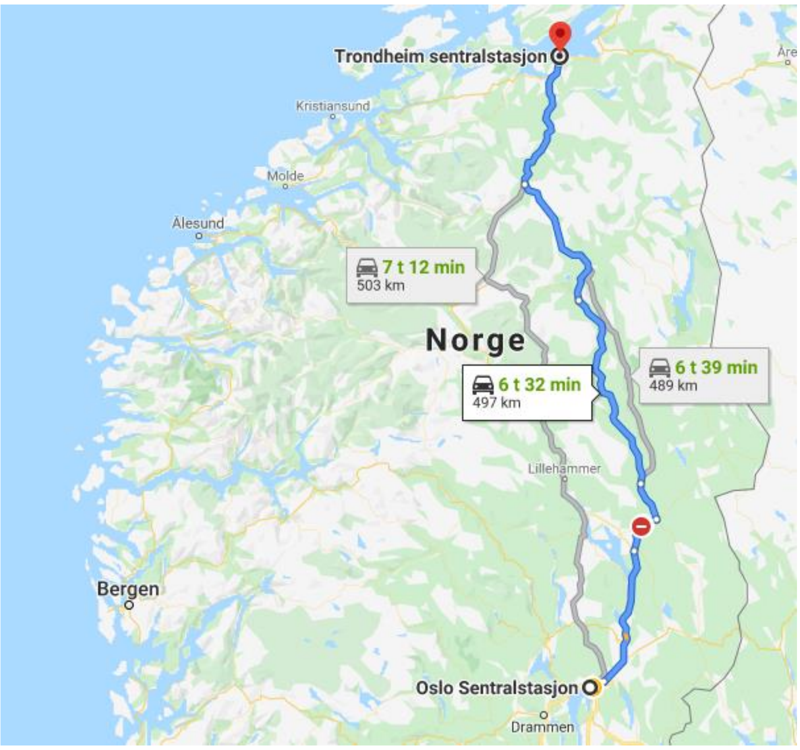

Figure 2.12: Snapshot of Google Maps showing route alternatives from the central station in Oslo to the central station in Trondheim (Google, 2020).

A set of relevant KPIs for the route model of RV Gunnerus are proposed in Table 2.2.

Table 2.2: KPIs for the route service of RV Gunnerus

KPI	**Why**	**How**
Level of comfort	By monitoring the level of comfort, we can mitigate the chance of discomfort and low performance of the POB.	Models to estimate the chance of seasickness based on vessel motions.
Fuel consumption	Fuel accounts for most voyage-related expenditures (VOYEX).	Models to estimate fuel consumption based on operational context or machinery monitoring, possibility to find optimal speed.
Voyage time consumption	Time spent in transit for a research vessel is not of value to a research mission.	Estimation of vessel velocity based on the trade-offs between fuel consumption, voyage length and required timeframe.
Voyage length	Scheduling, timeframes and the trade-off between inshore/offshore.	Sum of leg length estimation by use of SP, optimisation by Dijkstra’s shortest path algorithm.
Level of traffic	Mitigate risk of collision, close-calls and “active” navigation.	Real-time AIS tracking, route patterns and weather predictions, e.g. sunny afternoon during summer indicates more pleasure crafts.

Chapter 3

Methods

In this chapter, the methods and way of thought are presented. Firstly, a holistic view of how the digital service is modelled and constructed is presented. Following, details of the building blocks are presented.

3.1 Overview

To develop a digital service, there are multiple functions and sub-functions that needs to work together. Holistically, the service has one input where an operator states some requirements of the service, and one output, where the operator can see the results of the analysis. The creation of the service could therefore be top-down or down-up. Creating a top-down approach focuses on the end use of the service, where UI and insight are of concern. Down-up creation goes from the key requirements of the service and from there finds functional requirements of the service, as described in Section 2.5.3. This approach is chosen for this thesis.

The down-up approach requires clear interfaces between sub-functions and a structured workflow of the service. An overview of the framework is presented in Figure 3.1. The service starts at a UI where input variables are manually assigned. Thereafter, relevant external data are read and filtered. Following, the analysis starts where route alternatives are generated, filtered and assigned corresponding weather conditions based on historical data. The segment weather conditions are used to generate a seastate and related vessel responses for each route alternative. Furthermore, the set of route alternatives are analysed by optimisation emphasising relevant KPIs. Finally, the results are presented in a UI. The platform was developed in MATLAB. A flowchart and descriptions of the modelling in MATLAB are attached in Appendix C.

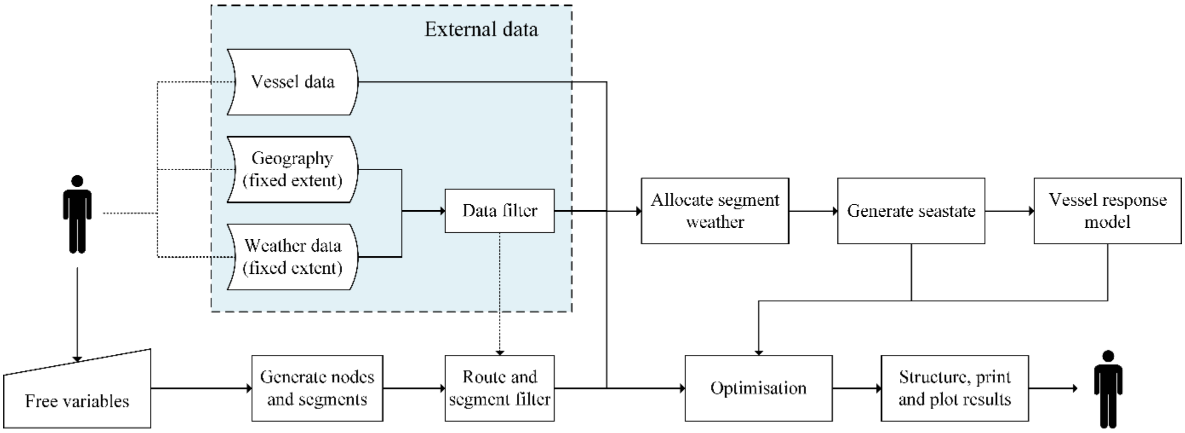

Figure 3.1: Overview of the digital service

3.2 Geography

In the case mission, there are two fixed points with known coordinates. The coordinate system is WGS 1984 Mercator in degrees-minutes-seconds (DMS). The destination, Træna, is located at approximately (66°30’00” N, 12°08’00” E). Trondheim is at approximately (63°27’00” N, 10°24’00” E). To cover a wide enough sea space, the boundaries were set to (62.75°N, 67.25°N) in latitudinal direction and (8.75°E, 13.25°E) in longitudinal direction, both in units decimal-degrees (DD).

The geography was extracted as a greyscale map from ArcGIS. To efficiently work in MATLAB, the maps were formatted as a raster file, consisting of pixels with corresponding RGB values for each pixel. This allowed easy analysis of feasible sailing areas, i.e. what is sea water and what is land. The basemap is shown in Figure 3.2.

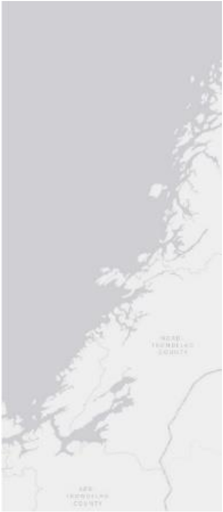

Figure 3.2: Greyscale map of selected area. Source: Esri, HERE, Garmin, ©OpenStreetMap contributors, and the GIS User Community (Esri, 2011).

The basemap itself and the formatting to raster file did not give clear boundaries to what is sea and what is land, thus the RGB values needed filtering. The basemap had a reference layer consisting of national borders, province borders and reference text. Furthermore, the conversion to raster file gave a variety of RGB values to compensate the resolution of boundaries between sea and land, i.e. shorelines. The RGB value for sea of Figure 3.2 is [208, 207, 212]. To filter allowable sailing areas, pixels with RGB values not within a range (*R*) of the RGB values for sea, were set equal to land with RGB [239, 239, 239], see Equation (3.1).

$$X_{ij} = \begin{cases} X_{ij} & , X_{ij} \leq [208, 207, 212] + R \\ [239, 239, 239] & , \mathrm{X}_{ij} > [208, 207, 212] + R \end{cases} \tag{3.1}$$

Where X_{ij} is a [R, G, B] vector of a pixel in location (i, j), and $R = 9$.

3.2.1 Exporting raster file from ArcGIS

Exporting from ArcGIS required some filtering. The standard export of a raster file from ArcGIS is an export of the user interface (UI) view and not necessarily the specified data frame. As such, the export gave a larger area than the one indicated in Figure 3.2. Without a fixed scale, the data frame would be approximately 2372x5619 (LxH) pixels. However, given the export format, the raster file was exported as an 8894x5619 raster, where the data frame is centred, shown in Figure 3.3. This corresponds to 600 dpi resolution export using a 1920x1080 (16:9) screen.

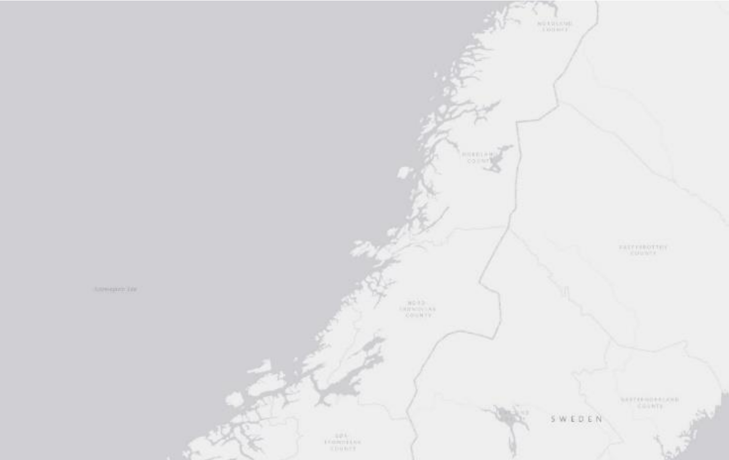

Figure 3.3: Exported map from ArcGIS, showing a raster frame extent wider than the data frame. Source: See Figure 3.2.

To allocate the data frame within the exported raster, the UI was cropped to the data frame. This generated a raster with the same resolution as in Figure 3.3, however with the data frame as the only visible data. That is, pixels outside the data frame were set to RGB [255,255,255] = White. This raster file was used to allocate the data frame pixel boundaries, i.e. the points in longitudinal direction (x_{1-DF}, x_{2-DF}) where $X_{ij} \neq [255, 255, 255]$. Using this method, the data frame boundaries, relative to an 8894x5619 raster file, were:

Table 3.1: Data frame boundaries of the exported raster from ArcGIS

Longitudinal boundaries	(x_{1-DF}, x_{2-DF})	(3261, 5633)	[px]
Latitudinal boundaries	(y_{1-DF}, y_{2-DF})	(1, 5619)	[px]

3.2.2 Assigning distances

Raster maps were the preferable workspace for the analysis. To work in a single format, reference frame, and a fixed coordinate system, the raster maps extracted from ArcGIS needed to be allocated in accordance to WGS ’84 Mercator. The coordinates of the data frame, in meters relative to the Greenwich MT (longitude) and the equator (latitude), were extracted from ArcGIS. The distance of 1 pixel in meters (L_{px}) was found with the WGS ’84 coordinates and the resolution of the raster maps, see below.

$$L_{px} = L_{px-x} = (x_2 - x_1) / (x_{2-DF} - x_{1-DF}) = L_{px-y} = (y_2 - y_1) / H \tag{3.2}$$

Where

- $\mathrm{L}_{px-x} = \mathrm{L}_{px-y}$ $[m]$ is the distance of 1 pixel in longitudinal- and latitudinal direction, respectively,
- Pixel coordinates (x_{1-DF}, x_{2-DF}) are the data frame boundaries, given in Table 3.1,
- *H* (height) is the raster resolution (LxH). Due to the raster export, L could not be used to find L_{px-x},
- Coordinates x_1, x_2, y_1, y_2 $[m]$ are the data frame boundaries in ArcGIS. x_1 is the western boundary, x_2 is the eastern boundary, y_1 is the southern boundary, y_2 is the northern boundary.

3.2.3 Assigning waypoints

Using raster maps, assigning waypoints were done by graphic approximation. A waypoint is here referred to as a fixed location, where we force the vessel to pass through between start- and end point. This is stated prior to the voyage. The start- and end point have known coordinates which were manually assigned as pixel locations based on the Mercator coordinates stated earlier. Each waypoint can be selected arbitrary by graphic approximation of the raster map, creating a network of sections from start- to end point.

The navigation from the start point, Trondheim, to outside Trondheimsfjorden was deemed neglectable in the routing problem. The fjord has only one way out to open sea, thus not included in the generation of alternative routes. The starting point were therefore set outside Trondheimsfjorden, near Ørland, approximately at 63°37’00’’ N, 9°14’00’’ E. This simplification avoided problems of no connecting routes from start- to end point. The pixel coordinates, relative to the raster in Figure 3.3, of the start- and end point are given in the table below.

Table 3.2: Pixel coordinates of start- and end point

Location	Notation	Pixel coordinates [px]
Start point (Ørland)	*ST*	(3550, 1075)
End point (Træna)	*FI*	(4800, 4700)

3.3 Weather modelling

Weather data were obtained from the open-source ERA5 project by the European Centre for Medium-Range Weather Forecasts (ECMWF) (Copernicus Climate Change Service (C3S), 2017). The project provides high resolution reanalyses of hourly weather conditions from 1979 and more. The selected timeframe for the analysis were 2019. The dataset particulars are summarised in the table below.

Table 3.3: Weather dataset particulars

Dataset name	ERA5 hourly data on single levels from 1979 to present (reanalysis)
Timeframe	Jan 1st 2019 – Dec 31st 2019.

Step	Every 2nd hour every day
Area boundaries	North: 67°N, South: 63°N, West: 9°E, East: 13°E
Variables	Mean wave direction $(\bar{\theta}_{wave})$, mean zero-crossing wave period $(\bar{T}_z)$, peak wave period (T_p), significant height of combined wind waves and swell (H_s), 10m u-component of wind (u), 10m v-component of wind (v)

The data area corresponds to the mapped area with the same reference system, WGS 1984 World Mercator. The area was split into 9-by-9 cells, where one cell is 0.5°N-by-0.5°E, shown in Figure 3.4. The centre of one cell is the weather recording coordinate, e.g. segment A1 has the boundaries [(67.25°N, 66.75°N), (8.75°E, 9.25°E)].

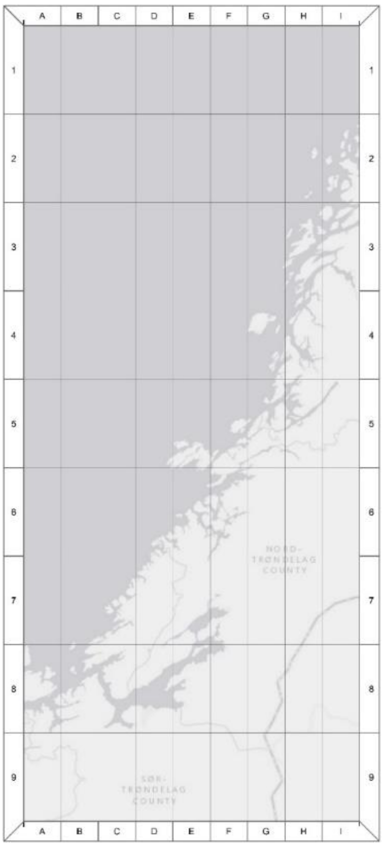

Figure 3.4: Weather data cells. Source: See Figure 3.2.

Further, the pixel distance of one weather data cell, shown in Figure 3.4, were found. The map were split in *n*-by-*m* cells, where each cell is L_{wf-x}-by-L_{wf-y}, see Equation (3.3).

$$\begin{aligned} L_{wf-x} &= (x_{2-DF} - x_{1-DF})/n \quad [px] \\ L_{wf-y} &= (y_{2-DF} - y_{1-DF})/m \quad [px] \end{aligned} \tag{3.3}$$

Where n = 9 and m = 9 for our dataset.

3.3.1 Dataset clean-up

The dataset was formatted in GRIB and netCDF. The GRIB-file was used for graphical visualisation in ArcGIS. The netCDF-file was used for dataset analyses in MATLAB. The netCDF-file exported the dataset from ECMWF C3S in longitude-latitude-time(T), thus giving data matrices flipped in the Y-direction and rotated -90°, relative to the global coordinate system. Additionally, the netCDF-dataset was split in 17x17xT cells, which indicated formatting errors, since weather recordings were found in the upper left 9x9xT cells. This was confirmed when comparing to the dataset in the GRIB-file. The netCDF-dataset was therefore reduced to a 9x9xT matrix.

Furthermore, the oceanic variables $(\overline{\theta}_{wave}, \overline{T}_z, T_p, H_s)$ did not have recordings for all cells in Figure 3.4. The cells without recordings contained Not-a-Number (NaN), which were cleared prior to weather condition allocation of generated routes further on. The NaN-cells, relative to Figure 3.4, without recordings were: [A8:C9], [D6:F9], [G5:H9], [I3:I9]. To clear NaN-cells, Equation (3.4) was used. This equation relies on non-NaN-cells in both directions (*i*, *j*), which for cells [A8:A9] was not possible. The values for [A8:A9] is therefore estimated by Equation (3.5).

$$Z_{ij}(t) = \eta \frac{Z_{(i-1)j} + Z_{i(j-1)}}{2} \text{ for } Z_{ij}(t) = NaN, \ (ij) \neq [A8, A9] \tag{3.4}$$

$$Z_{ij}(t) = \eta Z_{(i-1)j} \text{ for } Z_{ij}(t) = NaN, \ (ij) = [A8, A9] \tag{3.5}$$

Where

- $Z_{ij}(t)$ is an oceanic variable $(\overline{\theta}_{wave}, \overline{T}_z, T_p, H_s)$ in coordinates (*i* ,*j*) at time *t*. $i = A...I$, $j = 1...9$, $t = 1... T$,
- η is a scaling factor, $\eta = 0.8$ for $Z_{ij}(t) = H_{s_{ij}}(t)$, and = 1.0 for $Z_{ij}(t) = \left\{\overline{\theta}_{wave}, \overline{T}_z, T_p\right\}$.

3.3.2 Wind speed and -direction

The wind data was given by the velocity components [m/s] in longitudinal and latitudinal direction at a height of 10 meters above the sea- and land surface. The dataset contained recordings of all weather cells indicated in Figure 3.4. The longitudinal component (*u*) is air moving towards the east, i.e. positive x-direction relative to our coordinate system. The latitudinal component (*v*) is air moving towards north, i.e. positive y-direction of our coordinate system. The wind speed (V_{wind}) was found by Equation (3.6).

$$V_{wind} = \sqrt{u^2 + v^2} \tag{3.6}$$

To find the wind direction (θ_{wind}) relative to our coordinate system, where 0 rad = moving towards east and π/2 rad = moving towards north, we the four-quadrant inverse tangent was used. This is a built-in MATLAB function (*atan2(Y, X)*) and is based on the relation below.

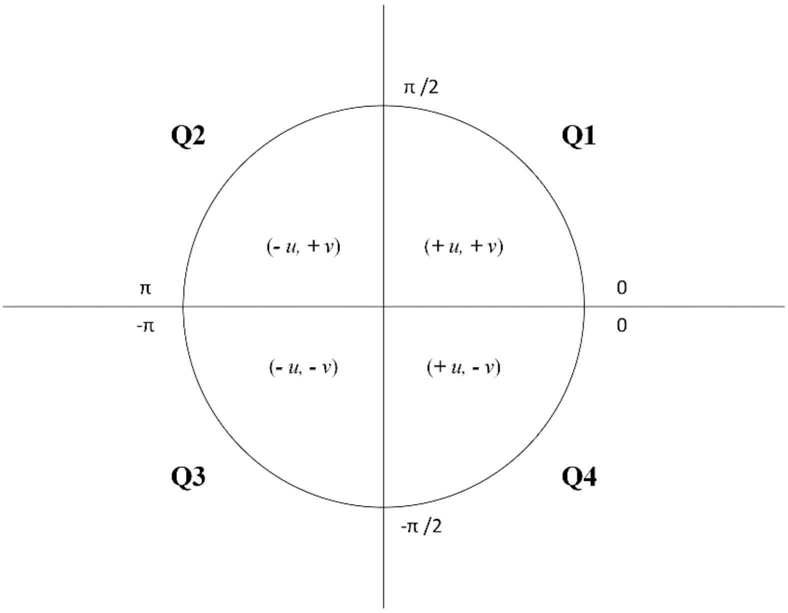

Figure 3.5: Four-quadrant inverse tangent emphasising wind components u and v. Q1-Q4 is the quadrant notation.

Using this relation, the wind direction, without using the built-in function *atan2*, can be found by Equation (3.7). With the use of *atan2(Y, X)*, Equation (3.7) simplifies to Equation (3.8).

$$\theta_{wind} = \begin{cases} \tan^{-1}(v/u) & , u \geq 0 , v > 0 \ , (u,v) \in Q1 \\ \pi + \tan^{-1}(v/u) & , u < 0 , v \geq 0 \ , (u,v) \in Q2 \\ \pi + \tan^{-1}(v/u) & , u \leq 0 , v < 0 \ , (u,v) \in Q3 \\ 2\pi + \tan^{-1}(v/u) & , u > 0 , v \leq 0 \ , (u,v) \in Q4 \end{cases} \tag{3.7}$$

$$\theta_{wind} = \begin{cases} \text{atan2}(v,u) & , v \geq 0 , (u,v) \in [Q1, Q2] \\ 2\pi + \text{atan2}(v,u) & , v < 0 , (u,v) \in [Q3, Q4] \end{cases} \tag{3.8}$$

3.3.3 Mean wave direction

Wave direction was given as the mean wave direction of sea surface waves ($\bar{\theta}_{wave-\deg}$). The dataset had unit degrees relative to true north, i.e. 0° means waves coming from north, and 90° means coming from east. To convert this data to radians relative to our coordinate system, Equation (3.9) were used.

$$\bar{\theta}_{wave} = \frac{3\pi}{2} - \frac{\pi}{180}\bar{\theta}_{wave-\deg} \tag{3.9}$$

The data which were initially between 0° and 90° got negative radians according to Equation (3.9). To solve this, we add 2π:

$$\bar{\theta}_{wave} = 2\pi + \bar{\theta}_{wave} \text{ for } \bar{\theta}_{wave} < 0 \tag{3.10}$$

Further, the wave direction unit is converted to waves *moving towards*, rather than *coming from*:

$$\bar{\theta}_{wave} = \begin{cases} \pi + \bar{\theta}_{wave} & , \bar{\theta}_{wave} < \pi \\ \bar{\theta}_{wave} - \pi & , \bar{\theta}_{wave} \geq \pi \end{cases} \tag{3.11}$$

3.3.4 Generating wave timeseries with the JONSWAP spectrum

For the Norwegian Sea, the Joint North Sea Wave Observation Project (JONSWAP) spectrum is applicable (Gudmestad, 2015). The spectrum is the Pierson-Moskovitz (PM) spectrum multiplied by a steepness factor, γ. The PM spectrum assumes that the sea is fully developed, where γ = 1, however for the North Sea and further north this is generally not applicable. The PM spectrum density function (SDF, $S_{PM}(\omega)$) is given by (Chandrasekaran, 2015):

$$S_{PM}(\omega) = \frac{\alpha g^2}{\omega^5}\exp\left[-1.25\left(\frac{\omega}{\omega_p}\right)^{-4}\right] \tag{3.12}$$

Where α is Phillips constant ≅ 0.0081, $\omega_p = 2\pi / T_p$ is the angular spectrum peak frequency.

The JONSWAP spectrum by Hasselmann *et al.* (1973) has five independent variables: significant wave height (H_s), angular spectrum peak frequency (ω_p), steepness factor (γ) and spectral width parameters (σ_A and σ_B). The steepness factor for the JONSWAP spectrum is γ = 3.3, however in this project, we allow it to vary based on the significant wave height (H_s), and peak wave period (T_p). The value of γ was found according to the guidelines in DNV-RP-C205 (Det Norske Veritas (DNV), 2010). The JONSWAP SDF ($S_J(\omega)$) is given by:

$$S_J(\omega) = \frac{\bar{\alpha}}{\alpha} S_{PM}(\omega)\,\gamma^{r(\omega)} \tag{3.13}$$

Where $\bar{\alpha}$ is the modified Phillips constant, given by Equation (3.14). *r(ω)* is given by Equation (3.15). The values for γ is given by Equation (3.16).

$$\bar{\alpha} = 5.058\left[\frac{H_s}{\left(T_p\right)^2}\right]^2\left(1-0.287\ln\left(\gamma\right)\right) \tag{3.14}$$

$$\begin{gathered} r(\omega) = \exp\left[-\frac{\left(\omega-\omega_p\right)^2}{2\sigma^2\omega_p{}^2}\right] \\ \sigma = \begin{cases} \sigma_A & , \omega \le \omega_p \\ \sigma_B & , \omega < \omega_p \end{cases} \end{gathered} \tag{3.15}$$

Where σ is the spectral width parameter with values σ_A = 0.07 and σ_B = 0.09.

$$\gamma = \begin{cases} 5 & , \dfrac{T_p}{\sqrt{H_s}} \le 3.6 \\ \exp\left[5.75 - 1.15\dfrac{T_p}{\sqrt{H_s}}\right] & , 3.6 < \dfrac{T_p}{\sqrt{H_s}} < 5 \\ 1 & , \dfrac{T_p}{\sqrt{H_s}} \ge 5 \end{cases} \tag{3.16}$$

Simulating wave elevation timeseries

There are multiple methods to generate a wave elevation timeseries, where two were chosen for comparison. Both methods use the superposition principle to sum a finite number of sinusoidal, harmonic Fourier components. In the first method, called "deterministic amplitude scheme" (DAS) (Merigaud and Ringwood, 2017), a discrete sequence is generated with uniformly distributed random phases and deterministic amplitudes, see Equation (3.17). The second method, called "random amplitude scheme" (RAS) (Merigaud and Ringwood, 2017), generates a discrete sequence with uniformly distributed random phases and normally distributed amplitudes with zero-mean and variance $S_J(\omega_k)\Delta\omega$, see Equation (3.18).

$$\eta_{3,t_i}^{DAS} = \sum_{k=1}^{M} A_k \cos\left(\omega_k t_i + \phi_k\right) \tag{3.17}$$

Where

- M is the number of Fourier components
- η_{3,t_i} is the wave elevation at time $t_i = i\Delta t$, $i =$ 1...N ,
- $\omega_k = k\Delta\omega$, $k =1...M$,
- $A_k = \sqrt{2S_J(\omega_k)\Delta\omega}$,
- ϕ_k is a random integer governed by a uniform distribution in [0, 2π].

$$\eta_{3,t_i}^{RAS} = \sum_{k=1}^{M} a_k \cos\left(\omega_k t_i\right) + b_k \sin\left(\omega_k t_i\right) \tag{3.18}$$

Where

- $a_k = \mathrm{C}_{t_i} \sqrt{S_J(\omega_k)\Delta\omega}$ and $b_k = \mathrm{D}_{t_i} \sqrt{S_J(\omega_k)\Delta\omega}$,
- C_{t_i} and D_{t_i} are randomly chosen variables at time t_i following the standard normal distribution.

Both timeseries had *N* number of time components with a total length of $T = \mathrm{N}\Delta t$, where Δ*t* is the component timestep. The timeseries consisted of *M* number of Fourier components, set prior to the generation of a sequence. We let N be equal to M. The timeseries variables are specified in Table 3.4.

Table 3.4: Wave timeseries simulation variables

Variable	Value	Unit
N = M	300	-
ω	[0.01, 3.0]	rad / s
Δω	0.01	rad / s
t_i	[0, 300]	s
Δ*t*	0.3344	s

Velocity and acceleration at free surface of the generated wave

The simulated wave elevation timeseries concerns elevation of the free surface. In addition to elevation, we want to extract the wave velocity $(\dot{\eta}_{3,t_i})$ and -acceleration $(\ddot{\eta}_{3,t_i})$ at free surface. This was done by Equations (3.19) and (3.20), for DAS and RAS, respectively.

$$\begin{aligned} \dot{\eta}_{3,t_i}^{DAS} &= \sum_{k=1}^{M} -\omega_k A_k \sin\left(\omega_k t_i + \phi_k\right) \\ \ddot{\eta}_{3,t_i}^{DAS} &= \sum_{k=1}^{M} -\omega_k^2 A_k \cos\left(\omega_k t_i + \phi_k\right) \end{aligned} \tag{3.19}$$

$$\begin{aligned} \dot{\eta}_{3,t_i}^{RAS} &= \sum_{k=1}^{M} -\omega_k \left(a_k \sin\left(\omega_k t_i\right) - b_k \cos\left(\omega_k t_i\right)\right) \\ \ddot{\eta}_{3,t_i}^{RAS} &= \sum_{k=1}^{M} -\omega_k^2 \left(a_k \cos\left(\omega_k t_i\right) + b_k \sin\left(\omega_k t_i\right)\right) \end{aligned} \tag{3.20}$$

3.4 Route generation

The following subsections explains route generation without waypoints and varying fidelity. Subsection 3.4.4 presents the extension to allow varying fidelity. To generate a network of alternative routes, the idea was to construct a versatile route generator, where altering a set of input parameters would effect the shape and extent of the generated network. The variables are presented in Table 3.5.

Table 3.5: Route generation variables

Variable	Notation	Unit
Longitudinal number of splits	*S*	-
Transverse number of splits	*H*	-
Transverse offset distance	*d*	px
Start node (*x*, *y*)	*ST*	px
Final node (*x*, *y*)	*FI*	px

The method is based on splitting a linear path between two known points into smaller segments, i.e. SP for a liner problem. The linear path between two known points can be split in two directions, where the number of splits in both directions and the transverse offset distance from centreline are variables. The longitudinal direction (*x*) refers to the linear direction between two known points, whereas transverse direction (*y*) is the perpendicular direction relative to *x*. A network of nodes and connecting routes is presented in Figure 3.6.

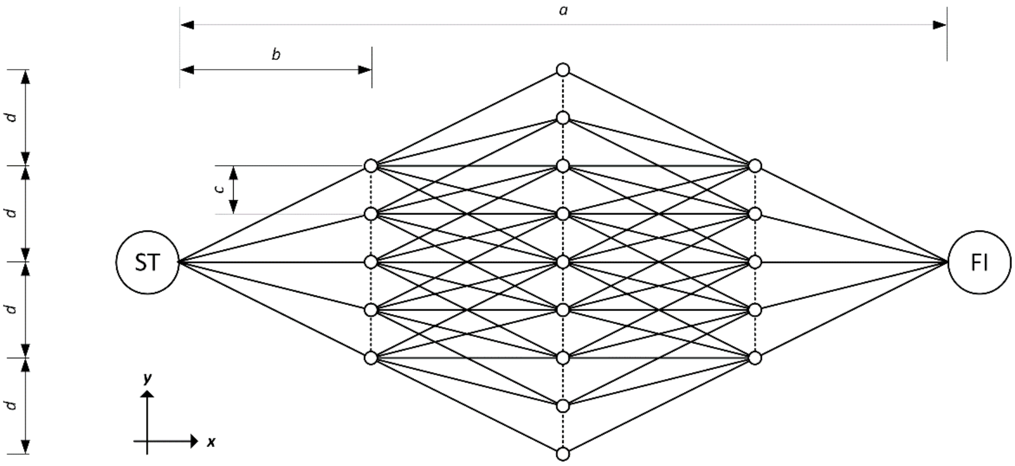

Figure 3.6: A route network between a start- and final node (H=5 and S=3)

The linear distance from *ST* to *FI* (*a*), the distance between S-splits (*b*) and the distance between H-splits (*c*), indicated in Figure 3.6, can be calculated from the input values and geometric relations, see Equation (3.21).

$$
\begin{aligned}
a &= \sqrt{\left(x_{FI} - x_{ST}\right)^2 + \left(y_{FI} - y_{ST}\right)^2} \\
b &= \frac{a}{S+1} \\
c &= \frac{2d}{H-1}
\end{aligned}
\tag{3.21}
$$

Where (x_{ST}, y_{ST}) and (x_{FI}, y_{FI}) are the pixel-coordinates of *ST* and *FI*, respectively.

3.4.1 Initiating network generation

Before generating the network, the sizes of the matrices were preallocated to ease the computation in MATLAB. There are two generated matrices: a node matrix with size $N_n \times 3$, and a segment matrix with size $N_s \times 3$. The number of nodes (N_n) can be found by Equation (3.22), only dependent of the input variables *H* and *S*.

$$
N_n = \begin{cases} 2 + \left(\dfrac{S+1}{2}\left(H-1\right)+1\right) + 2\displaystyle\sum_{t=1}^{(S-1)/2} 1 + t\left(H-1\right) & , S \in \mathbb{N}_o^+ = \{1,3,5,7,...\} \\ 2 + 2\displaystyle\sum_{t=1}^{S/2} 1 + t\left(H-1\right) & , S \in \mathbb{N}_e^+ = \{2,4,6,8,...\} \end{cases}
\tag{3.22}
$$

When counting segments, the number of segments differ due to odd and even numbers of both *H* and *S*. For odd numbers of *S*, the counting is the same for $H \in \mathbb{N}_o^+$ and $H \in \mathbb{N}_e^+$, see Equation (3.23). As shown in Figure 3.7, the connecting nodes of the middle section $(S_2 - S_3)$ does not have an equal number of connecting nodes as the rest of the network. The maximum number of connecting nodes to a given node is *H*, i.e. the maximum length of one segment is $l_{\max} = \sqrt{b^2 + d^2}$. For even numbers of *S*, Equation (3.24) was used for $H \in \mathbb{N}_o^+$, and Equation (3.25) is used for $H \in \mathbb{N}_e^+$.

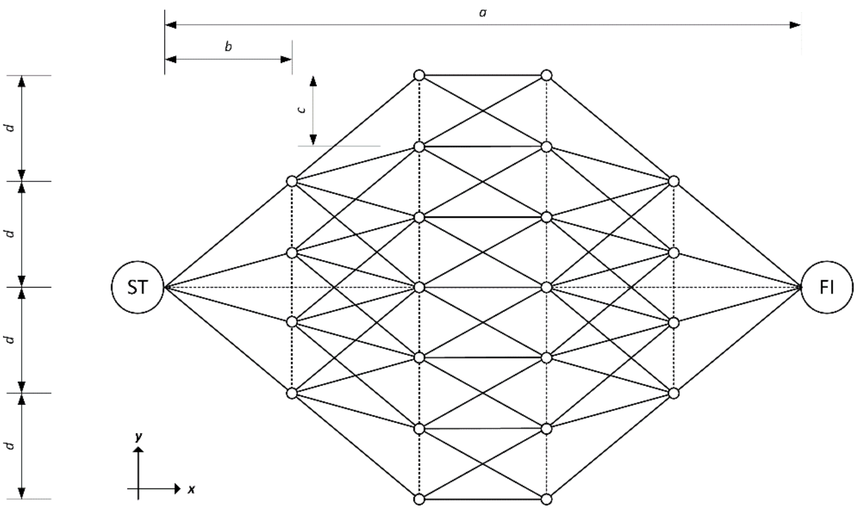

Figure 3.7: Network of nodes and segments (H=4, S=4)

$$N_s = 2H\left[1+\sum_{t=1}^{(S-1)/2} 1+t\left(H-1\right)\right], S \in \mathbb{N}_o^+ \tag{3.23}$$

$$N_s = 2H\left[\left(1-\frac{H}{2}\right)-\frac{S}{4}\left(H-1\right)+\sum_{t=1}^{S/2}1+t(H-1)+\frac{1}{H}\sum_{r=1}^{(H-1)/2}(H-r)\right], S \in \mathbb{N}_e^+, H \in \mathbb{N}_o^+ \tag{3.24}$$

$$\begin{aligned} N_s = &\left(H-1\right)\left[\frac{S}{2}\left(H-1\right)-H+3\right] \ \ldots \\ &+2H\left[\left(1-\frac{S}{2}H\right)+\sum_{t=1}^{S/2}\left(1+t\left(H-1\right)\right)+\frac{1}{H}\sum_{r=1}^{(H-2)/2}\left(\left(H-1\right)-r\right)\right], S \in \mathbb{N}_e^+, H \in \mathbb{N}_e^+ \end{aligned} \tag{3.25}$$

3.4.2 Assigning node ID and coordinates

After preallocation of both matrices, all nodes and segments were assigned an ID number. Each node is a vector [ID_i, x_i , y_i], where $ID_i = i$, $i \in [1, N_n]$, (x_i, y_i) is the pixel-coordinate of node *i*. Each segment is a vector $[ID_j, \mathrm{N1}_j, \mathrm{N2}_j]$, where $ID_j = j$, $j \in [1, N_s]$, $\mathrm{N1}_j$ and $\mathrm{N2}_j$ are the node IDs of the connecting nodes in segment ID_j . How to assign *N1* and *N2* is explained in Subsection 3.4.3.

When assigning coordinates to each node, we have known coordinates of *ST* and *FI*. The reference point was chosen to be *ST*, where the relative distance and heading can be found by geometric relations. The network starts at node 1 with $ID_1 = 1$, and branches out from the top left corner. If we consider the network in Figure 3.7, that gives S-split 1 (S_1) nodes ranging from 2-5. The next generation (S_2) follows with nodes 6-12 and so forth.

From Figure 3.6 and Equation (3.21), we have the global geometric relations between nodes. Furthermore, the route generator needed to account for $y_{ST} \neq y_{FI}$, i.e. a linear path that propagates with an angle relative to the *x*-axis. The angle (α) can be found by Equation (3.26) and shown in Figure 3.8.

$$\alpha = \left\{ \begin{matrix} \cos^{-1}\left(\dfrac{x_{FI} - x_{ST}}{a}\right) & , y_{ST} \leq y_{FI} \\ 2\pi - \cos^{-1}\left(\dfrac{x_{FI} - x_{ST}}{a}\right) & , y_{ST} > y_{FI} \end{matrix} \right\} \tag{3.26}$$

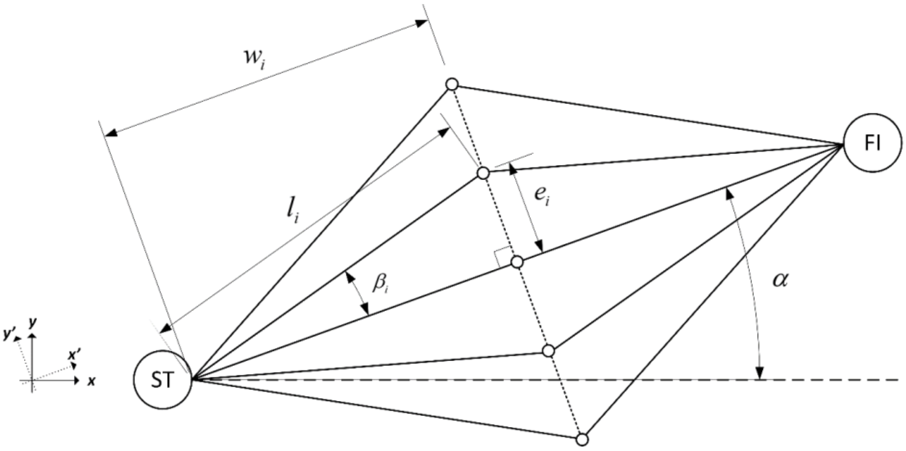

Figure 3.8: A tilted network with rotation angle α (H=5, S=1).

As indicated in Figure 3.8, the problem can be solved by using local variables and a sum of angles (α and β). The relationship between axes (*x*, *y*) and (*x'*, *y'*) is given in Equation (3.27).

$$\begin{aligned} x &= x'\cos\alpha - y'\sin\alpha \\ y &= x'\sin\alpha - y'\cos\alpha \end{aligned} \tag{3.27}$$

The local variables are β_i, w_i, e_i, l_i. β_i is the angle between (x_1, y_1) and (x_i, y_i), relative to the (*x'*, *y'*)-plane, w_i is the length in *x'* of (x_i, y_i), e_i is the length in *y'* of (x_i, y_i), l_i is the length between (x_1, y_1) and (x_i, y_i). Equation (3.28) can be used to calculate coordinates (x_i, y_i), and Equations (3.29) and (3.30) was used to compute l_i and β_i, respectively.

$$\begin{aligned} x_i &= x_1 + l_i \cos\left(\alpha + \beta_i\right) \\ y_i &= y_1 + l_i \sin\left(\alpha + \beta_i\right) \end{aligned} \tag{3.28}$$

$$l_i = \sqrt{w_i^2 + e_i^2} \tag{3.29}$$

$$\beta_i = \tan^{-1}\left(\frac{e_i}{w_i}\right) \tag{3.30}$$

The following two paragraphs presents the equations used to find w_i and e_i for both $S \in \mathbb{N}_o^+$ and $S \in \mathbb{N}_e^+$.

Assigning coordinates in a network with odd S $(S \in \mathbb{N}_o^+)$

$$w_i = bk \text{ , } S \in \mathbb{N}_o^+ \tag{3.31}$$

$$e_i = \begin{cases} dk - \left(z_k - 1\right)c & , k \le \left(S+1\right)/2 \\ d\left(\left(S+1\right) - k\right) - \left(z_k - 1\right)c & , k > \left(S+1\right)/2 \end{cases} \text{ , } S \in \mathbb{N}_o^+ \tag{3.32}$$

$$z_k = \begin{cases} H + \left(k-1\right)\left(H-1\right) & , k \le \left(S+1\right)/2 \\ z_{k-1} - \left(H-1\right) & , k > \left(S+1\right)/2 \end{cases} \text{ , } S \in \mathbb{N}_o^+ \tag{3.33}$$

Where

- k is a S-split counter, $k = 1, 2, ..., S$
- z_k is a H-split counter, $z_k = 1, 2, ..., N_k$. N_k is the number of nodes in S-split k, $N_k \in \left[H,\ H + \left(\left(S-1\right)/2\right)\left(H-1\right)\right]$, see Equation (3.34).

$$N_k = \begin{cases} H + \left(k-1\right)\left(H-1\right) & , k \le \left(S+1\right)/2 \\ N_{k-1} - \left(H-1\right) & , k > \left(S+1\right)/2 \end{cases} \text{ , } S \in \mathbb{N}_o^+ \tag{3.34}$$

Assigning coordinates in a network with even S $(S \in \mathbb{N}_e^+)$

$$w_i = bk \text{ , } S \in \mathbb{N}_e^+ \tag{3.35}$$

$$e_i = \begin{cases} dk - (z_k - 1)c & , k \le S/2 \\ d((S+1)-k) - (z_k - 1)c & , k > S/2 \end{cases} , S \in \mathbb{N}_e^+ \tag{3.36}$$

Where

- k is a S-split counter, $k = 1, 2, ..., S$,
- z_k is a H-split counter, $z_k = 1, 2, ..., N_k$. N_k is the number of nodes in S-split k , $N_k \in \left[H,\ H + ((S-2)/2)(H-1)\right]$, see Equation (3.37).

$$N_k = \begin{cases} H + (k-1)(H-1) & , k \le S/2 \\ N_{k-1} & , k = (S+2)/2 \\ N_{k-1} - (H-1) & , k > (S+2)/2 \end{cases} , S \in \mathbb{N}_e^+ \tag{3.37}$$

3.4.3 Assigning connecting nodes and segments

A segment was defined as a vector $[ID_j, \mathrm{N1}_j, \mathrm{N2}_j]$. To find the connecting nodes $\mathrm{N1}_j$ and $\mathrm{N2}_j$ of a segment $j \in [1, N_s]$, three principles were applied. The first concerns the maximum length of a segment, where we force the distance between two nodes (i and $u \ne i$) to be shorter than or equal to $l_{\max} = \sqrt{b^2 + d^2}$, see Equation (3.38).

$$\sqrt{(x_u - x_i)^2 + (y_u - y_i)^2} + \varepsilon \le l_{\max} \ , u \ne i \tag{3.38}$$

Where ε represents the rounding error when calculating node coordinates, we let $\varepsilon = 1\ [px]$. The second principle states that for a segment *j*, the connecting nodes $\mathrm{N1}_j$ and $\mathrm{N2}_j$, cannot be a part of the same S-split, i.e. a segment from a node $\left[\mathrm{N1}_j, x_i, y_i\right]$ to node $\left[\mathrm{N2}_j, x_u, y_u\right]$ has to move towards the end point, see Equation (3.39).

$$\begin{aligned} w_i &< w_u \ , u \ne i \\ bk_i &< bk_u \ , u \ne i \end{aligned} \tag{3.39}$$

Where k_i and k_u is the S-split where nodes i and $u \ne i$, are located, $(k_i \ne k_u) \in \left[0, (S+1)\right]$.

The third principle is complementary to Equations (3.38) and (3.39), needed to precisely allocate the correct connecting nodes. With high fidelity ($H \gg 2,\ S \gg 1,\ d \ll a$), the nodes are placed so closely that Equation (3.38) is insufficient. The following two equations presents how a connecting node $\mathrm{N2}_j$ was found with $S \in \mathbb{N}_o^+$ and $S \in \mathbb{N}_e^+$.

$$\mathrm{N2}_j = \begin{Bmatrix} \mathrm{N1}_j + \left(N_k - 1\right) + r_t & , k \le \left(S-1\right)/2 \\ \mathrm{N1}_j + \left(N_k - H\right) + r_t & , k > \left(S-1\right)/2 \end{Bmatrix} , S \in \mathbb{N}_o^+ \tag{3.40}$$

$$\mathrm{N2}_j = \begin{Bmatrix} \mathrm{N1}_j + \left(N_k - 1\right) + r_t & , k \le \left(S-2\right)/2 \\ \mathrm{N1}_j + \left(N_k - \left(H+1\right)/2\right) + r_t & , k = S/2 \text{ , } H \in \mathbb{N}_o^+ \\ \mathrm{N1}_j + \left(N_k - H/2\right) + r_t & , k = S/2 \text{ , } H \in \mathbb{N}_e^+ \\ \mathrm{N1}_j + \left(N_k - H\right) + r_t & , k > S/2 \end{Bmatrix} , S \in \mathbb{N}_e^+ \tag{3.41}$$

Where

- $k = 0,1,2,...,S$ is a S-split counter,
- N_k is the number of nodes in split k. For $N_{k=0} = 1$, and $N_{k>0}$ given in Equations (3.34) and (3.37),
- $r_t = t$ is a connecting node counter. For an arbitrary node $\mathrm{N1}_j$, there are maximum H connecting nodes, see Equation (3.42). If the equation is satisfied, $t = 1$ for $\mathrm{N1}_j$, $t = 2$ for $\mathrm{N1}_{j+1}$ and so forth.

$$t = 1,2,...,H \quad \text{for } \left(\mathrm{N1}_j = \mathrm{N1}_{j+1}, \mathrm{N1}_{j+2}, ..., \mathrm{N1}_{j+(H-1)}\right) \ne \mathrm{N1}_{j-1} \tag{3.42}$$

3.4.4 Waypoints and varying fidelity

As stated in Section 3.2.3, the route generator needed to include the possibilities of navigation by waypoints (*WP*) to improve the versatility of the service. Waypoints can be placed anywhere in the selected area if they are placed with sea connection. Introducing waypoints and varying fidelity, the network will consist of multiple sections, where one section is referred to as a sub-network between waypoints. The procedure is shown in Figure 3.9.

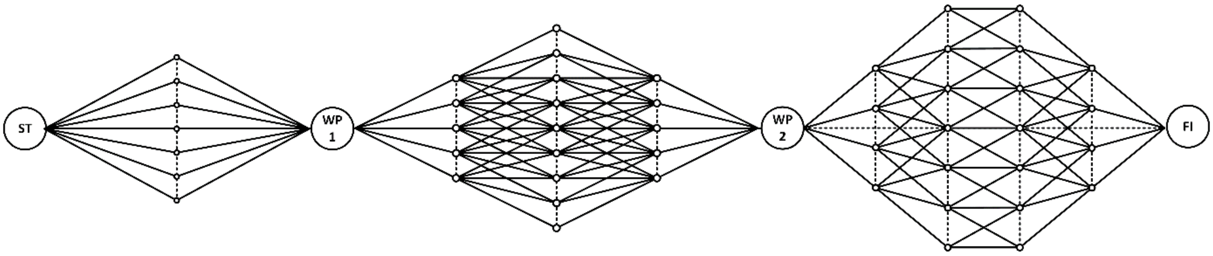

Figure 3.9: A network consisting of 3 sections, i.e. two waypoints, and varying fidelity. Section 1 with H=7, S=1; Section 2 with H=5, S=3; Section 3 with H=4, S=4.

The addition of multiple sections is solved by looping the route generator for as many sections stated. The input variables *ST* and *FI* were expanded to matrix form, consisting of $R\times 3$elements ($[ID_r, x_r, y_r]$), where *R* is the number of sections, $r = 1, 2, ...R$. The matrices *ST* and *FI* were formulated as shown in Equation (3.43), such that $ST_{r+1} = FI_r$. The input variables *S*, *H*, *d* can be either 1×1 vectors or $R\times 1$ vectors. If these variables are chosen to be 1×1 vectors, the shape of each section is equal. If they are $R\times 1$ vectors, the section shapes differ according to S_r, H_r, d_r.

$$\begin{pmatrix} ID_r & x_r & y_r \\ ID_{r+1} & x_{r+1} & y_{r+1} \\ & \vdots & \\ ID_R & x_R & y_R \end{pmatrix} \tag{3.43}$$

Furthermore, nodes and segments for each section were allocated in global matrices, consisting of all nodes and segments. Looping through *R* will count the waypoints twice, which was taken into consideration when adding each section to the global node matrix. Additionally, for both segments and nodes, the assignment of ID numbers was done by continuously updating the ID of waypoints in the matrices *ST* and *FI*.

When following this procedure, two matrices (*NOD* and *SEG*) was generated. Matrix *NOD* is the global node matrix $[N_{NOD} \times 3]$, and matrix *SEG* is the global segment matrix $[N_{SEG} \times 3]$. N_{NOD} and N_{SEG} can be found by Equation (3.44).

$$\begin{aligned} N_{NOD} &= 1 + \sum_{r=1}^{R} \left(N_{n,r} - 1 \right) \\ N_{SEG} &= \sum_{r=1}^{R} N_{s,r} \end{aligned} \tag{3.44}$$

Where $N_{n,r}$ and $N_{s,r}$ is the number of nodes and segments, respectively, in section *r*.

3.4.5 Filtering infeasible nodes and segments

After network generation, nodes and segments that are not feasible were filtered out. Two filters were applied. Firstly, nodes outside the weather data area are not deemed feasible. To filter these out, the relations in Figure 3.10 were used.

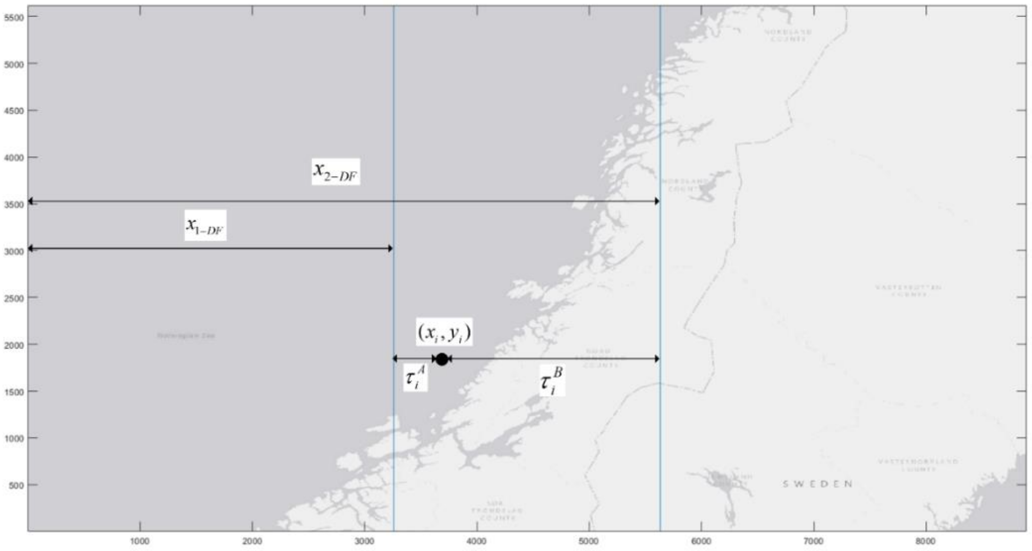

Figure 3.10: Node filter for nodes outside the data frame.

Here, we have known coordinates (x_i, y_i) of a node $i \in [1, N_{NOD}]$, and known distances in *x* for the data frame: x_{1-DF} and x_{2-DF}. To find nodes that are not located within the data frame, Equation (3.45) was used.

$$\begin{aligned} \tau_i^A &= x_i - x_{1-DF} \geq 0 \\ \tau_i^B &= x_{2-DF} - x_i \geq 0 \end{aligned} \tag{3.45}$$

The second filter states that feasible nodes cannot be located on land, see Equation (3.46).

$$X_{ij} \leq [208, 207, 212] + R \tag{3.46}$$

Where X_{ij} is the RGB vector of coordinate (x_i, y_i), $i = x_i$, $j = y_i$, and $R = 9$.

The two filters concern the placement of nodes. In addition to the two, all nodes are forced to have at least one connected node, i.e. isolated nodes are not allowed. To filter this, the procedure was to make sure that all filtered nodes were a part of the segment matrix, i.e. $\left(\mathrm{N1}_j, \mathrm{N2}_j\right) \in NOD$ and $ID_i \in SEG$ for all *i* and *j*.

3.5 Segment weather conditions and vessel response

When the vessel sails along a route, the weather conditions will need to change according to the length of the segment. This approach is fully dynamic, which for a high-fidelity network would be computationally heavy and possibly hard to model. Consequently, we needed an approach that eased computation time and was simpler to model. The weather conditions of each segment were therefore found by a quasi-static approach. The method was to select an interval of weather updates and a number of intervals. This way, we can obtain a matrix containing weather variables for each segment at given times. The variables are given below.

Table 3.6: Segment weather fidelity variables

Variable	Notation
Start time	t_0
Update interval	Δt
Number of intervals	$N_{\Delta t}$

The timespan investigated is $\left[t_0, t_0 + \left(N_{\Delta t} - 1\right)\Delta t\right]$. Notice that $t_0 + \left(N_{\Delta t} - 1\right)\Delta t$ needs to be large enough to include all route alternatives. If the network is wide and the vessel velocity is low, the outermost route alternatives will use considerable time and the timespan needs to be stated accordingly. Furthermore, if we consider a large Δt and high-fidelity network, the time to complete a segment would be far less than Δt. As such, the weather conditions of a segment *j* at a given time $t^* \neq t = \left\{t_0, t_0 + \Delta t, t_0 + 2\Delta t, ..., t_0 + \left(N_{\Delta t} - 1\right)\Delta t\right\}$ is estimated as the weather conditions of the nearest *t*.

The weather variables of interest are the variables governing the JONSWAP spectrum, H_s and T_p. If a segment has connecting nodes that are not in the same weather section, the area with the largest H_s is chosen, i.e. $H_s\left(j,t\right) = \max\left\{H_s\left(\mathrm{N1}_j, t\right), H_s\left(\mathrm{N2}_j, t\right)\right\}$ for segment *j* at time *t*. Furthermore, a wave timeseries is simulated at each interval for each segment. The timeseries are generated as described in Section 3.3.4. For each time step *t*, the maximum values of RAS- and DAS-timeseries are compared, and the timeseries with the largest peak wave elevation is chosen.

3.5.1 Vessel velocity

The route model was constructed to include vessel velocity variations based on weather conditions. For rough sea, we can assume that the vessel needs to operate at a lower speed than for calm seas. To model this variation, we allow the vessel velocity to vary based on the wave- and wind direction relative to the vessel heading, absolute wind velocity and H_s. The equation used to compute velocity is given below.

$$V_j(t) = r_1(t) r_2(t) V_{\max} \tag{3.47}$$

Where

- $V_j(t)$ is the vessel velocity of segment *j* at time *t*,
- $r_1(t)$ and $r_2(t)$ are velocity reduction factors due to waves and wind, respectively, given in Equation (3.48),
- $V_{\max}$ is the maximum cruising speed of the vessel, set equal to 10 kn.

$$r_v(t) = 1 - 0.8\left(\frac{\theta_{r,v}(t)/\pi}{2} + \frac{k_v(t)}{2}\right), \ v = \{1, 2\} \tag{3.48}$$

Where

- $\theta_{r,v}(t)$ is the relative angle between vessel heading and waves or wind, where the angle of waves and wind is given in radians *moving towards*, i.e. $\theta_{r,v} = \pi$ are head waves or -wind, $\theta_{r,v}(t) \in [0, \pi]$. The vessel heading is found by using the built-in MATLAB function atan2(), explained in Section 3.3.2, using segment nodes $(x_{\text{N1}}, y_{\text{N1}})$ and $(x_{\text{N2}}, y_{\text{N2}})$,
- $k_v(t) = \lambda_v(t) / \lambda_{v,\max}$ is the fraction of the largest waves or highest wind velocities, where $\lambda_v(t) = \{H_s(t), V_{wind}(t)\}$ is the weather variable at time *t*, $\lambda_v(t) \in [0,1]$. The

largest waves and highest wind velocities are assumed to be found in weather section A1.

The segment length can be found by $l_j = \sqrt{\left(x_{\mathrm{N1}} - x_{\mathrm{N1}}\right)^2 + \left(y_{\mathrm{N2}} - y_{\mathrm{N1}}\right)^2}$ for a segment between N1 and N2. The time to complete a segment *j* at time *t* can be found by $T_j(t) = l_j / V_j\left(t\right)$.

3.5.2 Vessel response

The vessel responses in 1 DOF was calculated by assuming that the vessel responses in heave $\left(\eta_3, \dot{\eta}_3, \ddot{\eta}_3\right)$ are equal to a fraction of the timeseries generated with the JONSWAP spectrum, see Equation (3.49).

$$\begin{aligned} \eta_3\left(t\right) &= \left(0.95 + 0.1\left(A - B\right)\right)\eta_{t_i}\left(t\right) \\ \dot{\eta}_3\left(t\right) &= \left(0.95 + 0.1\left(A - B\right)\right)\dot{\eta}_{t_i}\left(t\right) \\ \ddot{\eta}_3\left(t\right) &= \left(0.95 + 0.1\left(A - B\right)\right)\ddot{\eta}_{t_i}\left(t\right) \end{aligned} \tag{3.49}$$

Where *A* and *B* are random integers governed by the standard uniform distribution in the interval $\left[0,1\right]$, and $\left(\eta_{t_i}, \dot{\eta}_{t_i}, \ddot{\eta}_{t_i}\right)$ are the generated timeseries of free surface elevation, -velocity and -acceleration, respectively, at time *t*.

3.6 Route optimisation

The find the favourable route alternatives, two methods were used in the analysis. The first concerns shortest path, where Dijkstra’s algorithm (Kirk, 2020) was used. With a route network with odd H, the shortest path between an origin and destination is a straight line, however with even H, the algorithm could provide useful results. The second method concerns the KPIs stated in the case description.

The second method is based on finding the local optima of each segment. The optimisation starts at ST, finds the best connecting node based on the weighing between $T_j(t)$, $\eta_3(t)$, $\dot{\eta}_3(t)$ and $\ddot{\eta}_3(t)$. This procedure continues until the final node is reached. The objective function $\left(z\right)$ for each step is presented in Equation (3.50), where $r_i(t) = \left\{T_j\left(t\right), \eta_3\left(t\right), \dot{\eta}_3\left(t\right), \ddot{\eta}_3\left(t\right)\right\}$ is a decision variable vector at time t, w_i is the weight coefficient of r_i, $i = 1, 2, ..., N_w$, and N_w is the number of decision variables.

$$\min\left(z = \sum_{i=1}^{N_w} \max\left(\left|r_i(t)\right|\right) w_i \right) \tag{3.50}$$

To prevent the path from moving towards dead ends, we need to constrain the selection of connecting nodes. Firstly, we force segment variable $\mathrm{N2}_j \in \mathrm{N1}$ for any segment *j*, where $\mathrm{N1} \subset SEG$. This procedure works well if we have a low-fidelity network with few possible dead ends. However, by increasing fidelity, the chance of ending on the wrong side of e.g. a peninsula increases. Hence, we need to force any of the connecting nodes of the next 3 generations to have connecting nodes as well, see Figure 3.11. If we consider an arbitrary node $\mathrm{N1}_j$ (black) in a segment set $A \subset SEG$, the possible paths for the next three generations are within $\{B, C, D, E\} \subset SEG$. The constraint is met if any of the nodes in subset *D* have connecting segments to subset *E* (grey target nodes). For Figure 3.11, this implies that all segments moving through green nodes meet the constraint, whereas infeasible routes are passing through red nodes.

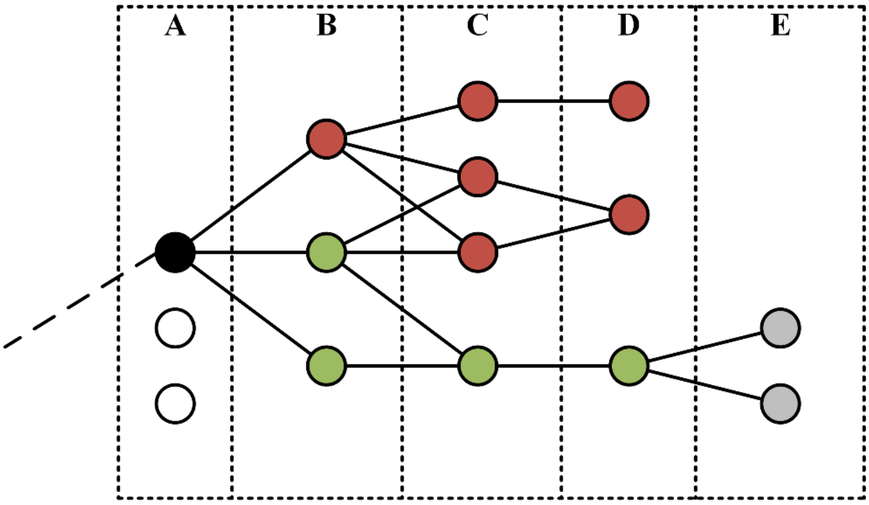

Figure 3.11: Local optima connecting nodes constraint from an arbitrary node (black) to a target node (grey)

For this analysis, 5 different weightings were chosen: (1) equal contributions, i.e. all $w_i = 0.25$, and sole contribution of each decision variable, i.e. (2) $w_1 = 1$, (3) $w_2 = 1$, (4) $w_3 = 1$, and (5) $w_4 = 1$.

Chapter 4

Results

In this chapter, the results of the performed analyses with regards to the research objectives of this thesis are presented. First, the network fidelity analysis is presented where the route generation variables are free to vary. Thereafter, the effects of weather variable variation are presented for a set of network fidelity configurations. Lastly, optimal routes with seasonal variations of the case, presented in Section 2.5, are presented.

4.1 Network fidelity analysis

The fidelity analysis was conducted solely by network generation and visual interpretation, i.e. run only route generation model and needed sub-functions in MATLAB. The analysis investigated the contributions of H, S, d , as well as running time. The latter depends on the computer setup, where MATLAB 2019b was used on a Windows 10 laptop with 8 GB RAM and a quad-core 2.1-3.7 GHz processor.

For the assessment, routes were generated at certain levels of extent, controlled by the variable *d*. Note that the route variable *H* should be kept an odd number., because for $H \in \mathbb{N}_e^+$, the connecting nodes $\left[\mathrm{N2}_j, \mathrm{N2}_{j+H-1}\right]$ of a random node $\mathrm{N1}_j = \cdots = \mathrm{N1}_{j+H-1}$, will never be the shortest distance between two S-splits (b), except for the middle S-split of the network.

4.1.1 Small extent network

A small extent network was defined, relative to the raster being used, as all networks with $d \leq 50$ px . Given the geography of the case area and the position of *ST* and *FI*, narrow networks with small *S*, routes offshore for all route alternatives. This is because the maximum heading angle, when moving towards network centre, from any node is $\beta = \tan^{-1}(d / b)$, i.e.

with small *S*, *b* becomes very large and β moves towards 0. Consequently for very large *S*, β moves towards $\pi/2$. The differences in network extent when keeping $d = 50$ px , $H = 3$ and varying *S* is shown in Figure 4.1 and Figure 4.2.

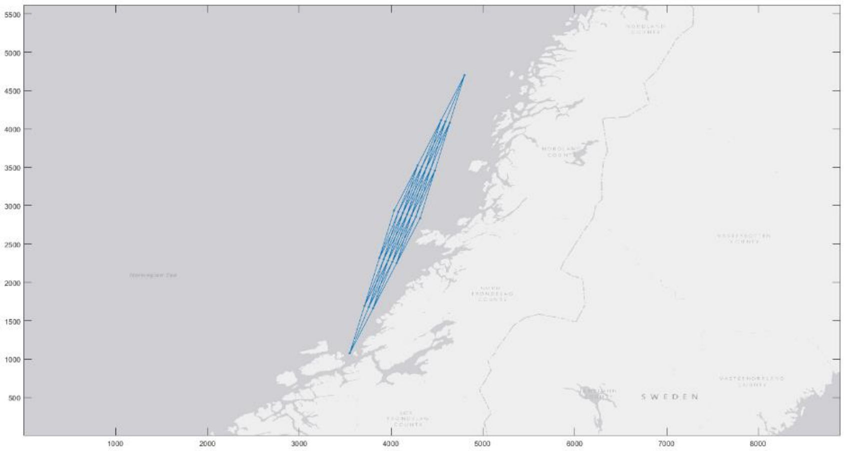

Figure 4.1: Small extent network with S = 5

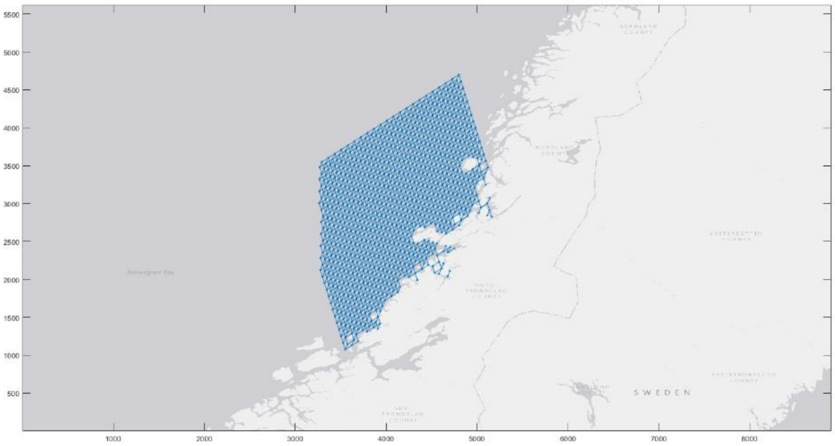

Figure 4.2: Small extent network with S = 50

The network in Figure 4.2 consists of 1061 nodes and 2948 segments. The computational time for both networks is neglectable. As we can see, the network includes some segments that are not connected to the main network. Furthermore, it is apparent that *S* had a large effect on filtering infeasible segments, i.e. segments that crosses land. Moreover, it is notable that the straight through Rørvik, approximately at (4600, 2550) px, has not been covered by the network. Sailing through this passage is the only way to avoid offshore routing through this area, thus a key straight to include in the generation of routes.

To find the required fidelity of a small extent network to include this passage, we let $H = 3$. It can be shown that by while $H = 3$, *S* would need to be very large to include the straight through Rørvik. This is due to the geography of the area. Thus, a required *H* needs to be found for configurations of *S*, or we need to allow generation of routes that cross land, i.e. $S \leq 50$. For the network to be wide enough to include the passage, we need $S \geq 25$. Table 4.1 tabulates a set of different network configurations of *S* and *H* that all include passage through Rørvik, where $T_{C,RG}$ is the route generation computational time.

Table 4.1: Configurations of H and S that includes the straight through Rørvik in a small extent network

S	H	N_{NOD}	N_{SEG}	$T_{C,RG}$ [s]
25	5	680	3,056	6
25	13	1,993	23,368	26
27	5	773	3,516	6
30	7	1,390	8,924	11
30	13	2,750	32,762	37
30	20	4,334	79,267	93
35	7	1,821	11,716	15
50	9	4,170	34,941	47

The only configuration in Table 4.1 that gave a network that picked up the straight and did so without generating infeasible segments was $S = 50$ and $H = 9$, shown in Figure 4.3. For this network, *d* and *S* are too small to include inshore routing north of Rørvik. To include inshore routing along the coastline north of Rørvik for a small extent network, $T_{C,RG}$ would be too high to be practically considered.

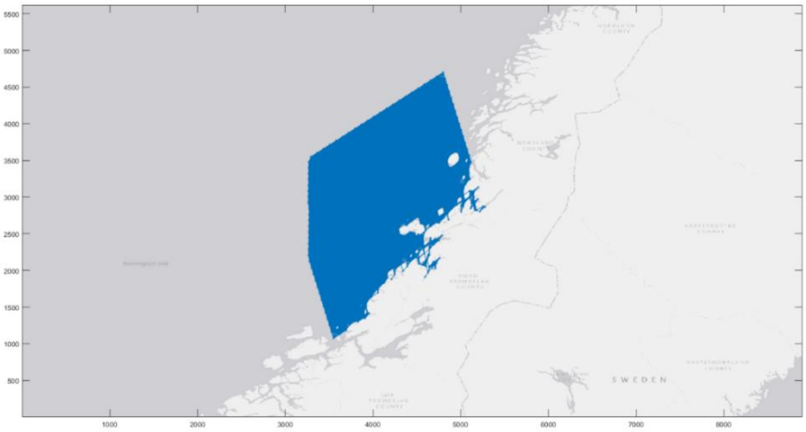

Figure 4.3: Feasible small extent network with S=50 and H=9

To relate the number of nodes and segments to the computational time, a linear regression was conducted in MS Excel with the dataset in Table 4.1. Regression of runtime and N_{NOD} gave no significant correlation with an R-value of 0.83. However, runtime and N_{SEG} gave significant correlation with an R^2-value of 0.99. The corresponding equation is given below.

$$T_{C,RG}\left(N_{SEG}\right) = 0.0012 N_{SEG} + 1.5 \tag{4.1}$$

4.1.2 Medium extent network

A medium extent network is defined as a network with $d \in \langle 50,100]$. First, we notice that also for medium extent networks, *S* needs to be of a certain size. $S \geq 20$ covers most of the coastline with $d = 60$, however the last half of the network does not include inshore routing options due to the extent of the network. Table 4.2 presents required *S* to include inshore routing north of Rørvik with increasing *d* and $H = 5$.

Table 4.2: Required S in a medium extent network

***d* [px]**	**Required *S***	**N_{NOD}**	**N_{SEG}**	**$T_{C,RG}$ [s]**
60	105	4,807	22,532	29
70	90	3,349	15,595	20
80	80	2,614	12,106	16
90	75	2,199	10,113	14
100	65	1,708	7,834	11

The data in Table 4.2 support the relation between number of segments and computational time given in Equation (4.1), with an R^2-value of 0.99. The network configurations do however not include routing options through Rørvik without altering *H*, except for $d = 60$ and $S = 105$. When comparing $T_{C,RG}$ of the medium- and small extent networks, we see that for increasing *d*, the computational time decreases while fulfilling the fidelity requirements. When moving on to the full analysis, this is advantageous.

4.1.3 Large extent network

A large extent network is defined as a network with $d > 100$. For large extent networks, *S* is not as governing as for small- and medium extent networks. It still plays a part in filtering out segments that cross land, however for the width of the network, large *d* generates suitable networks. In these networks, the contributions of *H* are more interesting, where we can generate a variety of route alternatives with a relatively small number of segments.

Firstly, we consider $d \in [101, 200]$, small *S* and *H*. By iteration, it is apparent that $S \geq 10$ and $d \geq 150$ gives promising networks, where two are presented here. The first network represents a coarse network with configuration $S = 5, H = 3, d = 190$, which gives 25 nodes and 54 segments in 3.0 seconds. It includes the straight through Rørvik, however has mulitple segments crossing islands. The second, presented in Figure 4.4, has configuration $S = 10, H = 6, d = 200$, where 136 nodes and 634 segments are generated in 3.4 seconds. This network includes the straight and does not include that many infeasible segments.

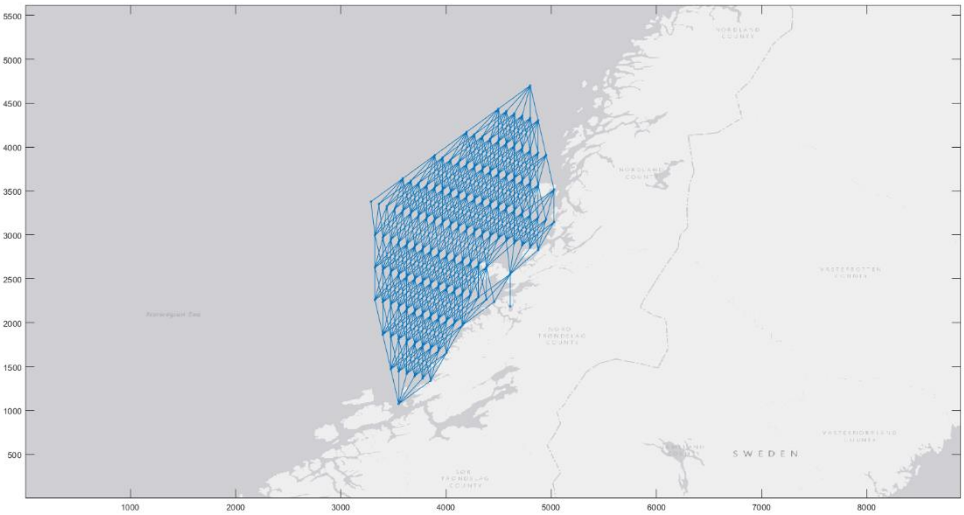

Figure 4.4: Feasible large extent network with S=10, H=6 and d=200 px

Moving on to large extent networks with $d > 200$. By increasing *d* and keeping $S \leq 10$ and $H \leq 10$, we can see that $d > 500$ generates networks of little relevance. Networks with *d* surpassing 500 px could be useful for sections with large islands and we know that inshore routing is possible, while keeping $T_{C,RG}$ low. Furthermore, linear regression of N_{SEG} and $T_{C,RG}$ for large extent networks support the correlation between the variables with an R^2-value of 0.98, however showing slight deviance, see Appendix A.1.

4.1.4 Network with waypoints and varying fidelity

The various extent levels have advantages and disadvantages. Small extent networks are great when moving across areas without inshore routing possibilities and no need for high-fidelity navigation. Medium networks are accurate enough to include some inshore routing, however the need for large *S* causes high computational time. Large networks perform well with coarse navigation and when multiple heading alternatives are needed, i.e. large *H*.

Instead of generating a complete network from *ST* to *FI*, a partition into smaller sections with different fidelity requirements could be useful. For the case area, we know that the straight through Rørvik and the section north of Rørvik causes challenges due to the limited sea space. Therefore, waypoints were stated on each side of island where the Rørvik straight is located, see Table 4.3.

Table 4.3: Coordinates of the 2 waypoints in a 3-section network

Waypoint	**Coordinates (x, y) [px]**
WP1	(4203, 2202)
WP2	(4655, 3127)

For this network, we have 3 sections between *ST* and *FI*. The first section has no need for high fidelity, however should include some route alternatives offshore, i.e. $d > 50$ while $S < 10$. Section 2 is the area around Rørvik, where we have two options: (1) low fidelity and allow some infeasibility, and (2) high fidelity with only feasible segments. The last section needs to have adequate fidelity and extent to allow inshore routing, i.e. $d > 150$ while $S < 10$. Table 4.4 presents a selection of different feasible configurations, where each element in the vectors are the fidelity of the respective section.

Table 4.4: Network configurations with 3 sections with varying fidelity

Config.	***S***	***H***	***d* [px]**	N_{NOD}	N_{SEG}	$T_{C,RG}$ **[s]**
1	[1, 3, 6]	[5, 7, 3]	[150, 150, 200]	56	130	3
2	[1, 15, 6]	[5, 7, 3]	[150, 50, 200]	374	1,988	5
3	[1, 15, 20]	[5, 5, 3]	[150, 50, 70]	459	1,531	5
4	[1, 15, 8]	[5, 5, 5]	[150, 50, 200]	316	1,248	4

5	[1, 15, 6]	[5, 5, 5]	[150, 50, 200]	287	1,118	4

The regression between $T_{C,RG}$ and N_{SEG} of the extended dataset, given in Table A.2, gave an insignificant correlation with an R^2-value of 0.92. Moreover, the regression parameters a and b $(y = ax + b)$ show no equivalence to Equation (4.1). Hence, we can assume that Equation (4.1) is invalid for network generation with multiple sections.

Of the different configurations in Table 4.4, configuration 1 and 3 present two different approaches with good quality, presented in Figure 4.5 and Figure 4.6, respectively.

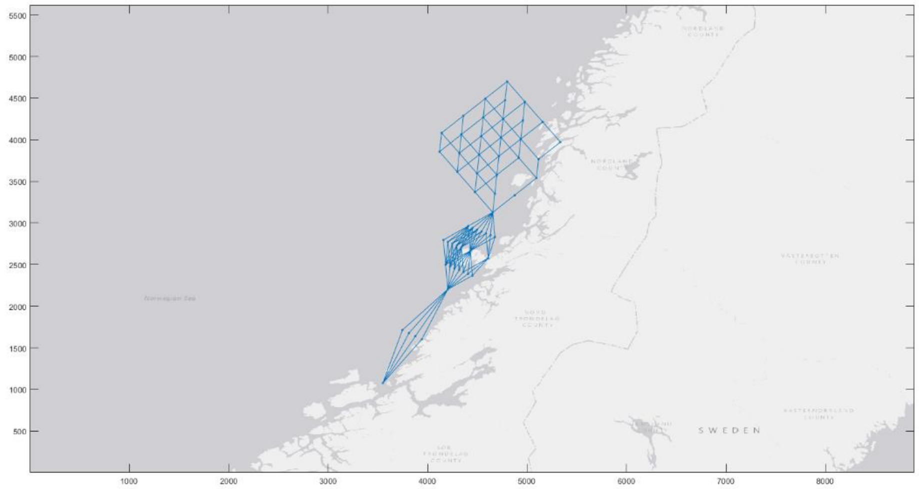

Figure 4.5: 3-section network with configuration 1

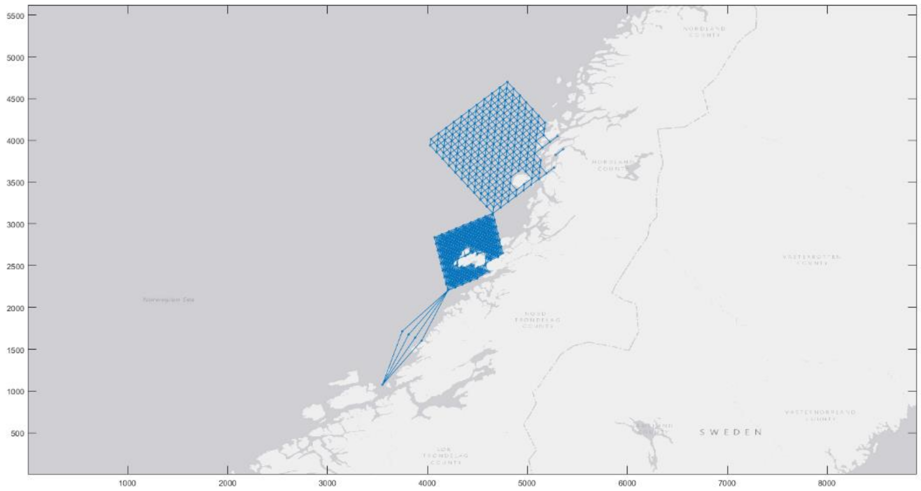

Figure 4.6: 3-section network with configuration 3

Configuration 1 presents a low-fidelity approach where we allow some infeasible segments. Especially within section 2, where we have multiple sections crossing the island we wanted to avoid. The network does however give quite good results with few available segments. When comparing to configuration 3, the latter has 11.8 times more segments. Configuration 3 allows an insignificant amount of infeasible segments, where the larger islands and peninsulas have no crossing segments. Furthermore, the computational times are quite similar due to the constant time to complete initiation of geography and weather data. However, when extending the analysis to include route weather, response and optimisation, the computational time per segment is expected to increase, i.e. configuration 1 will be preferable.

4.2 Weather fidelity variable variation

To assess the effect of the weather fidelity variables $(N_{\Delta t}, \Delta t)$, all MATLAB functions were run. The JONSWAP- and timeseries variables were not altered. The weather data used was weather for the first days of January, i.e. $t_0 = 0$. For this analysis, 4 different networks were assessed: 1-section low-fidelity (1LF), 1-section high-fidelity (1HF), 3-section low-fidelity (3LF), 3-section high-fidelity (3HF).

Table 4.5: Network configurations for the weather variable variation analysis

Network	***S***	***H***	***d* [px]**	***N_NOD***	***N_SEG***	***T_C,RG***
1LF	5	3	190	25	54	3
1HF	10	6	200	136	634	3
3LF	[1, 3, 6]	[5, 7, 3]	[150, 150, 200]	56	130	3
3HF	[1, 15, 20]	[5, 5, 3]	[150, 50, 70]	459	1,531	5

The absolute minimum sailing time is by sailing along the shortest path between *ST* and *FI* (*a*) of a 1-section network with the maximum speed $(V_{\max})$, given in Equation (4.2). The different networks were analysed with the same set of weather variable variations, given in Table 4.6. The next 4 subsections present the results of the analyses considering local optima when all decision variables are weighted equally, i.e. $w_i = 0.25$ for all *i*. The regression of the total computational time, $T_{C,tot}$, is presented in Section 4.2.5.

$$\left(N_{\Delta t} - 1\right)\Delta t \geq \frac{aL_{px}}{V_{\max}} = \frac{3834 * 211}{3600 * 10 * 0.5144} = 32 \text{ h} \qquad (4.2)$$

Table 4.6: Weather variable variations

Variation	Δt	$N_{\Delta t}$	$(N_{\Delta t}-1)\Delta t$
1	2	101	200
2	4	51	200
3	8	26	200
4	12	18	204

4.2.1 1-section low-fidelity network (1LF)

The results of the analysis of a 1LF network with weather variable variations are presented in Table 4.7. The optimal path crosses the island where Rørvik is located. The paths are given in vector format, where each element correspond to the node ID. The average vessel velocity is abbreviated Avg. V $\left(\bar{V}_V\right)$, and total sailing time is denoted T_S.

Table 4.7: Optimal paths of a 1LF network with varying weather update fidelity

Var.	$T_{C,tot}$ [s]	Path	Distance [m]	Avg. V [kn]	T_S [h]
1	30	[1, 4, 9, 15, 21, 24, 25]	832,458	4.01	112
2	17	[1, 4, 9, 16, 21, 24, 25]	844,114	4.07	112
3	10	[1, 4, 9, 16, 21, 24, 25]	844,114	4.07	112
4	8	[1, 4, 9, 16, 21, 24, 25]	844,114	3.68	124

For a 1LF network, the effect of varying weather update fidelity was minimal. We can see that the optimal paths tend to route inshore. This was confirmed when assessing all weather variations for all w_i, where none of the cases gave offshore routing. Note that for variation 4, $\bar{V}_V$ is much lower, i.e. if the network fidelity is low and the nearest weather interval is found at a H_s peak, the vessel is forced to sail a much longer distance with low velocity than for higher fidelity of weather updates, see Figure 4.7.

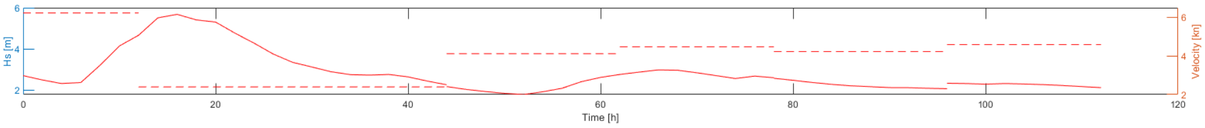

Figure 4.7a: 1LF variation 1

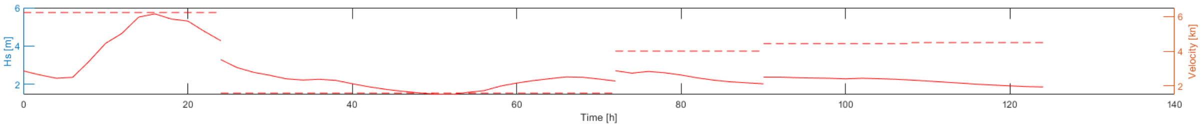

Figure 4.7b: 1LF variation 4

Figure 4.7: The effect of weather update fidelity of a 1LF network, where the solid line is the significant wave height and the dashed line is vessel velocity

4.2.2 1-section high-fidelity network (1HF)

For the 1HF network, path differences are more apparent, presented in Table 4.8. As for all variations of the 1LF network, the 1HF network tends to route inshore. This was confirmed with all weather variations for all w_i.

Table 4.8: Optimal paths of a 1HF network with varying weather update fidelity

Var.	$T_{C,tot}$ [s]	Path	Distance [m]	Avg. V [kn]	T_S [h]
1	277	[1, 7, 16, 30, 51, 75, … 103, 126, 144, 155, 161, 162]	875,070	4.22	112
2	140	[1, 7, 16, 30, 51, 75, … 103, 126, 144, 155, 161, 162]	875,070	4.07	116
3	74	[1, 7, 16, 30, 49, 75, … 100, 126, 144, 155, 161, 162]	880,400	4.32	110
4	53	[1, 7, 16, 30, 51, 75, … 100, 126, 144, 155, 161, 162]	880,281	4.32	110

The optimal path is fully feasible and routes inshore throughout the network of variation 1 and 2, while variation 3 and 4 choses routes further from the coastline north of Rørvik, see Figure 4.8 and Figure 4.9, respectively. The line colours are described in Table 4.9.

Table 4.9: Line colour properties

Line colour	Optimisation model
Yellow	Dijkstra’s shortest path algorithm
Red	Local optima with equal weight $(w_i = 0.25)$
Green	Local optima with sole contribution from $\eta_3(t)$ $(w_1 = 1)$
Black	Local optima with sole contribution from $\dot{\eta}_3(t)$ $(w_2 = 1)$

Cyan	Local optima with sole contribution from $\ddot{\eta}_3(t)$ $(w_3 = 1)$
Magenta	Local optima with sole contribution from $T_j(t)$ $(w_4 = 1)$

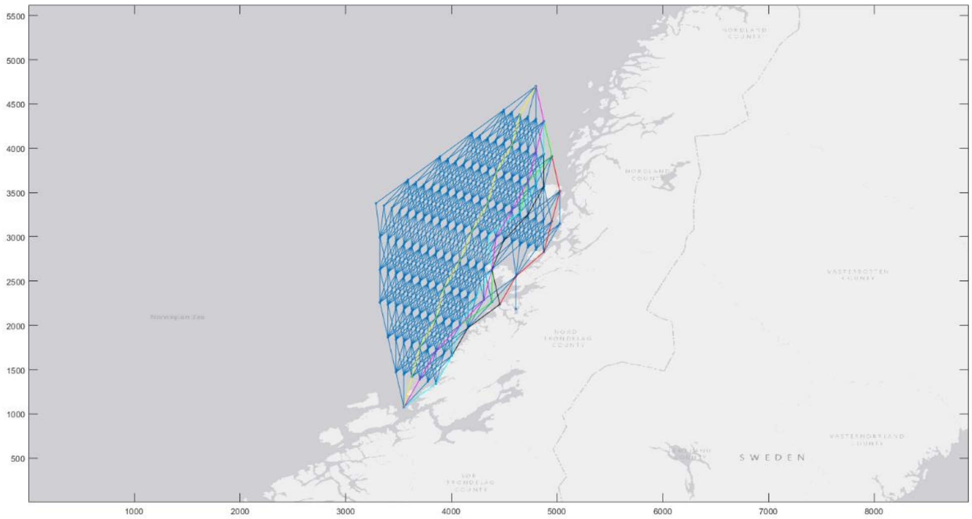

Figure 4.8: 1HF network with weather variation 1

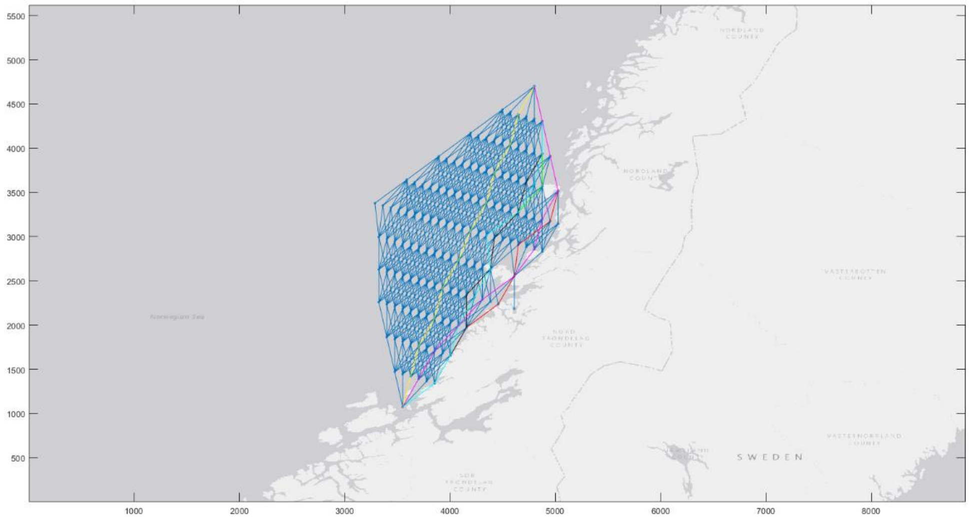

Figure 4.9: 1HF network with weather variation 4

4.2.3 3-section low-fidelity network (3LF)

The analysis of 3LF showed significantly more route variation for varying weather conditions. The results are presented in Table 4.10. Contradictory to 1LF and 1HF, the optimal paths for 3LF showed that the optimal routes select offshore routes for some variations in optimisation model. For variations 1 and 2, the optimal path crosses the island at Rørvik, thus in practice infeasible. For variations 3 and 4, the path goes through the Rørvik straight.

Table 4.10: Optimal paths of a 3LF network with varying weather update fidelity

Var.	$T_{C,tot}$ [s]	Path	Distance [m]	Avg. V [kn]	T_S [h]
1	60	[1, 3, 7, 12, 21, 30, 35, … 37, 41, 48, 55, 62, 65, 66]	895,402	4.17	116
2	32	[1, 3, 7, 12, 21, 29, 35, … 38, 43, 49, 57, 62, 65, 66]	933,239	4.34	116
3	19	[1, 3, 7, 14, 25, 34, 35, … 38, 43, 49, 57, 62, 65, 66]	942,378	4.24	120
4	15	[1, 3, 7, 14, 25, 34, 35, … 38, 43, 49, 57, 62, 65, 66]	942,378	4.17	122

The path alternatives for the different weather variations gives substantial differences in sailing distance. The average velocity does not differ as much, and by comparing the sailing time and -distance; differences in weather update fidelity seems governing. The velocity variations of all weather variations are given in Figure 4.10.

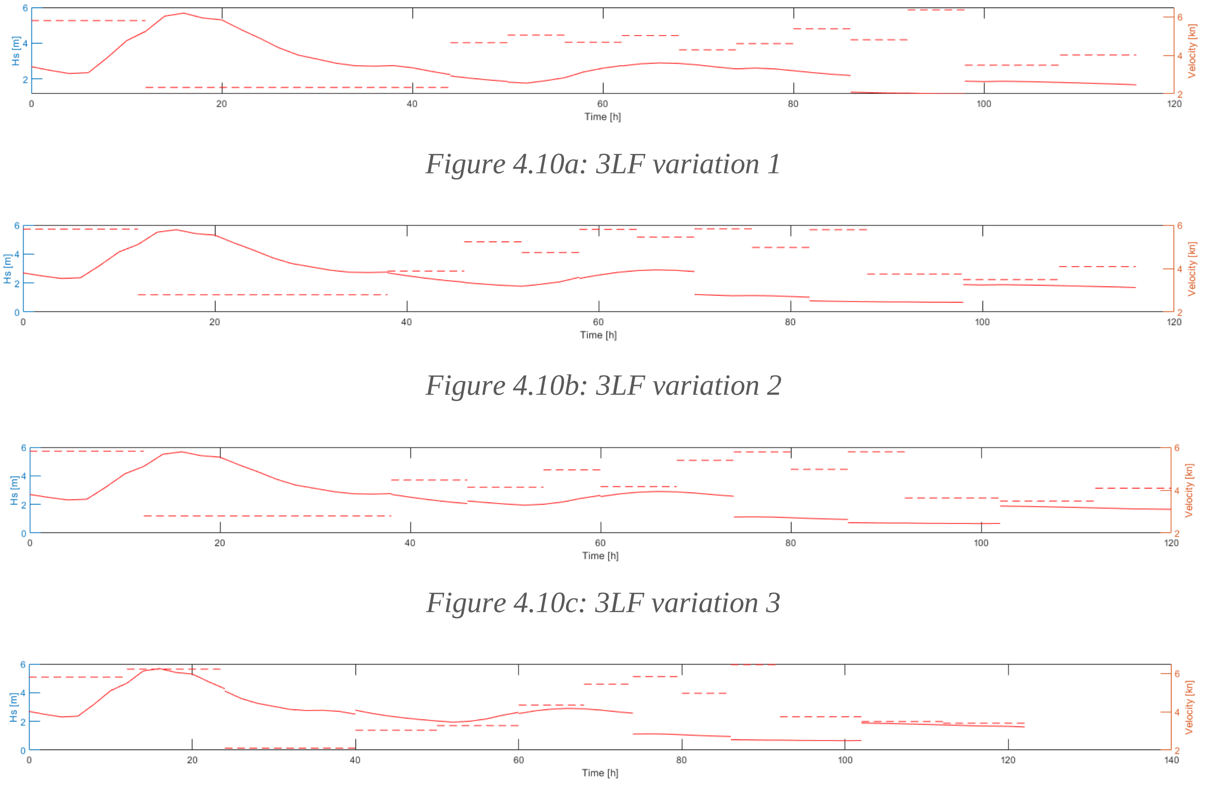

Figure 4.10a: 3LF variation 1

Figure 4.10b: 3LF variation 2

Figure 4.10c: 3LF variation 3

Figure 4.10d: 3LF variation 4

Figure 4.10: Velocity variation of a 3LF network with varying weather update fidelity, where the solid line is significant wave height and the dashed line is the vessel velocity

4.2.4 3-section high-fidelity network (3HF)

The variations of weather update fidelity show no apparent trend among the optimal routes in a 3HF network. For 1HF and 3LF, the main trend was that variations 1 and 2, and 3 and 4 routed similarly. However, for 3HF, the variations 1 and 3 routes through Rørvik, whereas variations 2 and 4 routes offshore. For the last section of the network, the optimal path is inshore or near-inshore for all weather variations. The optimal paths are presented in Table 4.11. Table 4.12 presents the particulars of each variation.

Table 4.11: Optimal paths of a 3HF network with varying weather update fidelity

Var.	Path
1	[1, 3, 7, 11, 16, 26, 41, 58, 82, 107, 140, 169, 196, 220, 237, 254, 267, 274, 279, … 282, 287, 293, 302, 313, 326, 341, 357, 375, 394, 415, 435, 454, 470, 484, 495, … 504, 511, 516, 519, 520]
2	[1, 3, 7, 11, 19, 31, 45, 66, 91, 120, 152, 181, 209, 231, 250, 263, 273, 278, 279, … 282, 287, 293, 302, 313, 326, 341, 357, 375, 394, 415, 435, 454, 470, 484, 495, … 504, 511, 516, 519, 520]
3	[1, 3, 7, 10, 16, 27, 41, 60, 82, 108, 139, 170, 198, 223, 243, 257, 268, 274, 279, … 281, 285, 292, 300, 310, 323, 337, 353, 371, 391, 413, 434, 453, 470, 484, 495, … 504, 511, 516, 519, 520]
4	[1, 3, 7, 10, 18, 29, 45, 66, 91, 120, 152, 182, 208, 231, 249, 264, 273, 278, 279, … 282, 287, 293, 301, 311, 323, 338, 354, 373, 394, 415, 435, 454, 470, 484, 495, … 504, 511, 516, 519, 520]

Table 4.12: Optimal path particulars of a 3HF network with varying weather update fidelity

Var.	$T_{C,tot}$ [s]	Distance [m]	Avg. V [kn]	T_S [h]
1	663	945,459	4.25	120
2	347	941,612	4.38	116
3	182	912,065	4.25	116
4	129	924,583	4.38	114

Moreover, the different weights gave more variations in routing than for 1LF, 1HF and 3LF. The differences may indicate correlation between the weights, which is investigated further in

Section 4.3.1. Figure 4.11 and Figure 4.12 presents the plots of the optimal paths for variation 2 and 3, respectively, highlighting differences in path choices. As we can see, variation 2 gives inshore routing, whereas variation 3 gives offshore routing, when $w_i = 0.25$.

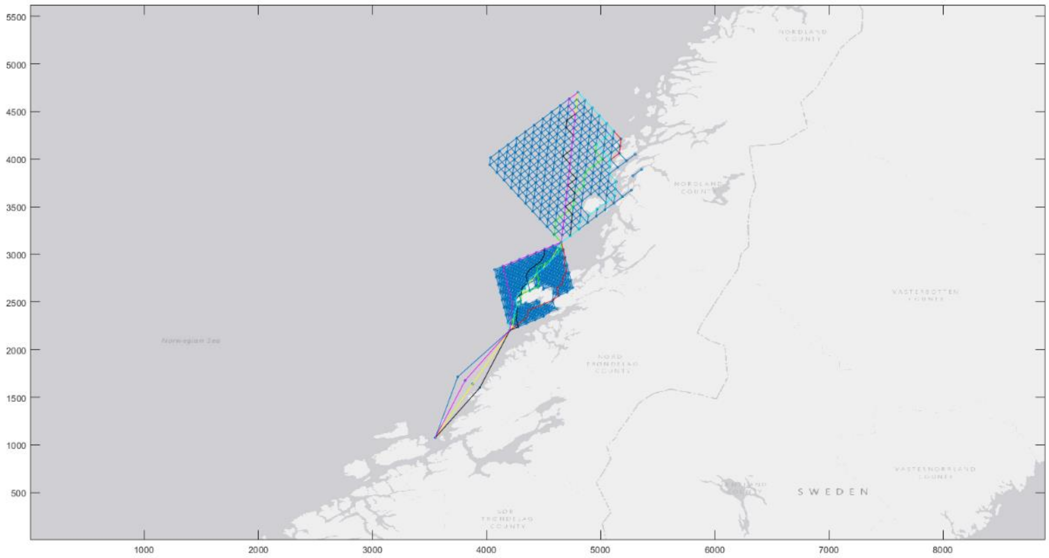

Figure 4.11: 3HF variation 2

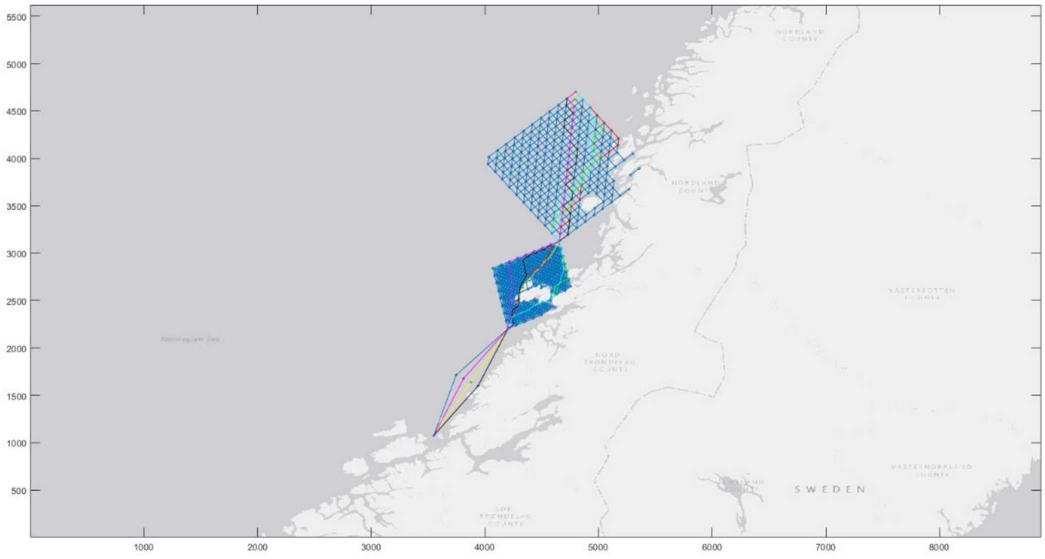

Figure 4.12: 3HF variation 3

4.2.5 Regression of total computational time

The analysis of total computational time gave significant results for the correlation between computational time and number of weather updates by using linear regression. The regression graph and data points are presented in Figure 4.13. The equations and corresponding R^2-value for the different networks are presented in Table 4.13.

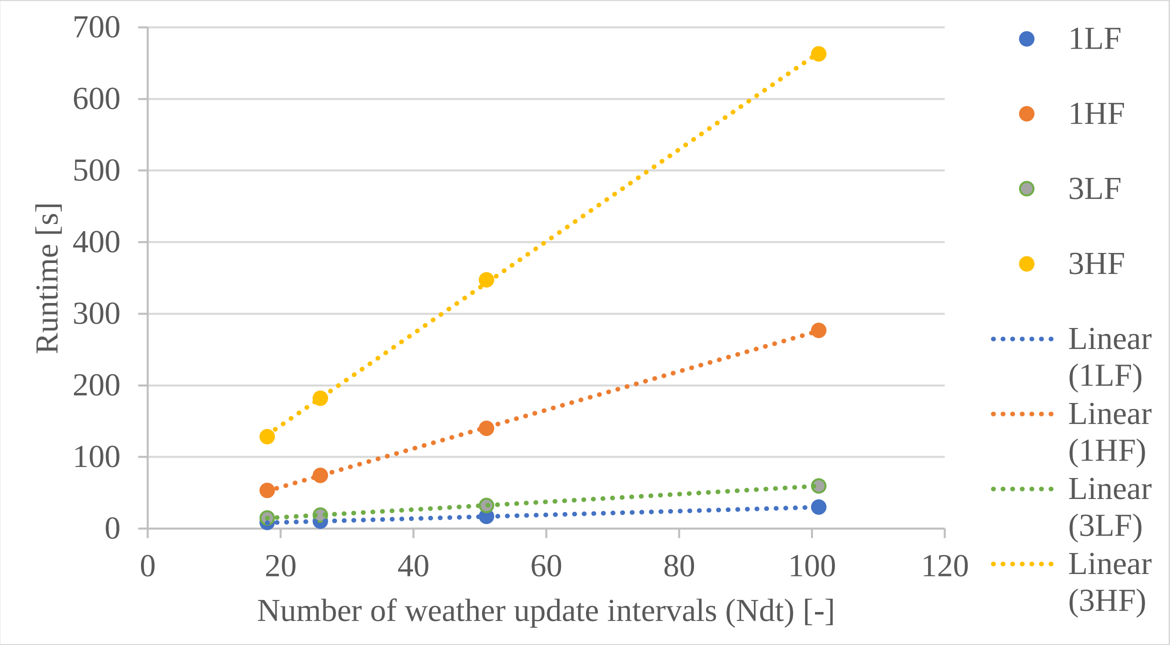

Figure 4.13: Linear regression of computational time as a function of number of weather updates

Table 4.13: Linear regression of computational time and number of weather updates for the different network configurations

Network	**Equation**	**R^2**
1LF	$T_{C,tot}\left(N_{\Delta t}\right)=0.26N_{\Delta t}+3.5$	0.9998
1HF	$T_{C,tot}\left(N_{\Delta t}\right)=2.69N_{\Delta t}+4.0$	0.9999
3LF	$T_{C,tot}\left(N_{\Delta t}\right)=0.54N_{\Delta t}+4.9$	0.9999
3HF	$T_{C,tot}\left(N_{\Delta t}\right)=6.43N_{\Delta t}+15.1$	0.9998

The equations above do not give any good model for the computational time of an arbitrary network; hence we need a different approach. The variables N_{SEG} and $N_{\Delta t}$ were chosen as the independent variables of a multiple linear regression of computational time, while forcing the intersect = 0. This gave no significant results with an R^2-value of 0.79 and an $\mathrm{R}_{adj}^2=0.70$ for Equation (4.3). It did however show significant correlation between $T_{C,tot}$ and each independent variable, where the p-value of N_{SEG} was 0.0008, and 0.0486 of $N_{\Delta t}$. The full regression results and datasets are attached in Appendix A.3.

$$T_{C,tot}\left(N_{SEG},N_{\Delta t}\right)=0.166N_{SEG}+1.194N_{\Delta t} \qquad (4.3)$$

4.3 Case: optimal fidelity configuration and seasonal variation

As presented in the prior sections, the analysis output differs based on the fidelity configurations. The optimal case mission fidelity was selected based on a consideration between computational time and required network fidelity. It is found that network configuration 3HF gives the most satisfying results, even though the computational time is high. Furthermore, weather variable variation 3 seems to be a suitable choice for the case, where high-quality results are obtained while minimising computational time. The optimal case fidelity variables are presented in the table below.

Table 4.14: Case optimal fidelity variables

Variable	**Value**
S	[1, 15, 20]
H	[5, 5, 3]
d	[150, 50, 70]
Δt	8
$N_{\Delta t}$	26

The optimal routes for the case mission are found throughout the year, which also gives us an indication of how the model performs with seasonal variation. The weather datasets contain weather every 2 hours for 2019, which gives a total of 4380 entries over 12 months, and average 365 datapoints per month. When assessing seasonal variation, we let t_0 be equal to the first recording of every month, see Table 4.15.

Table 4.15: Values of t_0 for each month

Month	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**Jun**
t_0	0	365	730	1,095	1,460	1,825
Month	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
t_0	2,190	2,555	2,920	3,285	3,650	4,015

Simulating the network for all months gives us the seasonal variations presented in Table 4.16. The optimal paths are based on equal weighting of the decision variables, i.e. $w_i = 0.25$. The full datasets of the paths with node ID, as well as plots, are attached in Appendix B.

Table 4.16: Optimal paths by month of the case mission

Month	Path	Distance [m]	Avg. V [kn]	T_S [h]
Jan	Offshore	912,065	4.25	116
Feb	Inshore	929,710	4.18	120
Mar	Offshore	933,784	5.25	96
Apr	Offshore	908,500	4.63	106
May	Inshore	918,073	4.13	120
Jun	Inshore	928,822	4.11	122
Jul	Offshore	928,454	4.82	104
Aug	Offshore	917,316	6.35	78
Sep	Inshore	904,096	3.76	130
Oct	Offshore	895,542	4.17	116
Nov	Offshore	937,680	4.96	102
Dec	Inshore	912,740	5.73	86

Of the results presented above, there are 4 months that stand out: March, August, September and December. Sailing in March gives us an optimal path with a long distance, however with high average velocity and corresponding low sailing time, even though the weather conditions for March were presumed to be harsh. For August, the simulation gives us the lowest sailing time of all months, which corresponds well to the presumption that August often have calm oceanic conditions. For September, the story is quite the opposite, presenting the highest sailing time of all months. A well-known phenomenon is autumn storms, where the peak season is within September, thus corresponding well to the presumption that September would give inshore routing and low average velocity.

However, most surprisingly is the route simulation in December. December is usually a month with harsh weather, however the simulation presents this month as the third best month to complete the case mission when only emphasising sailing time. The optimal path for December even routes inshore, see Figure 4.14, which we know represents a longer sailing distance than offshore. The shortest path by Dijkstra's algorithm, routes offshore with a distance of 831,000 meters. The path is tabulated in Table 4.17.

Table 4.17: Dijkstra's shortest path of the case's optimal network

Path	[1, 4, 7, 10, 16, 27, 41, 60, 82, 109, 140, 172, 199, 223, 242, 258, … 269, 276, 279, 281, 285, 291, 299, 309, 321, 335, 351, 369, 389, … 410, 430, 448, 464, 478, 490, 500, 508, 514, 518, 520]

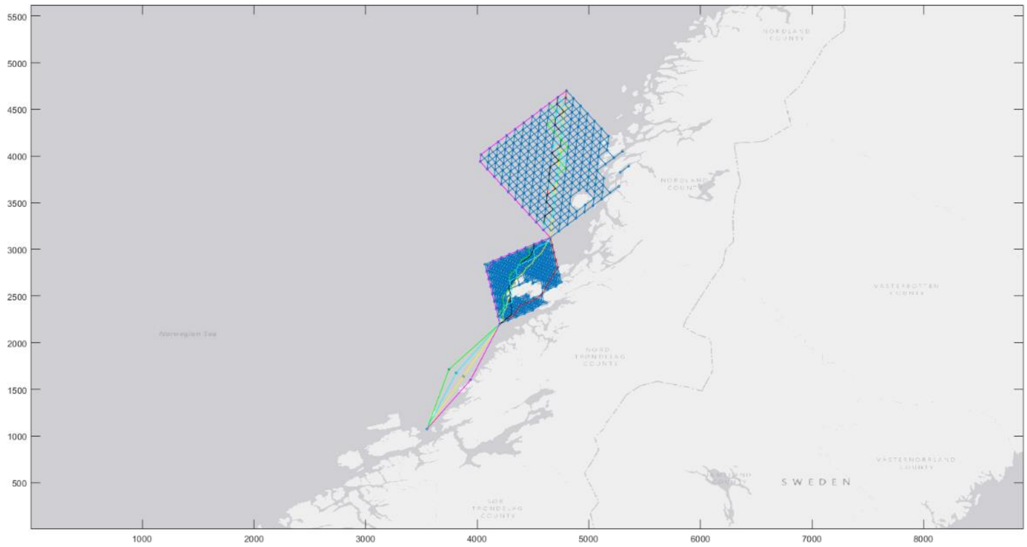

Figure 4.14: Case optimal paths for December. Line colours are described in Table 4.9

4.3.1 The effect of decision variable weights in local optima optimisation

The effect of decision variable weightings was analysed by multiple linear regression of the variation in result variables in MS Excel. The optimal path results gives sailing distance $\left(D_S\right)$, average vessel velocity $\left(\bar{V}_V\right)$ and sailing duration $\left(T_S\right)$. The dependent variables are thus, $D_S\left(\left(w_i=0.25\right),m\right),\bar{V}_V\left(\left(w_i=0.25\right),m\right),T_S\left(\left(w_i=0.25\right),m\right)$ for all months $m=1,2,...,12$ and all $w_i=\left\{w_1,w_2,w_3,w_4\right\}$. Following, the independent variables of the regression are $D_S\left(\left(w_i=1\right),m\right),\bar{V}_V\left(\left(w_i=1\right),m\right),T_S\left(\left(w_i=1\right),m\right)$ for all months $m=1,2,...,12$ and all $w_i=\left\{w_1,w_2,w_3,w_4\right\}$.

The regression of sailing distance and sailing duration show not significant results. Moreover, all p-values of the different optimisation models were insignificant, i.e. p >> 0.05. Hence, there are no clear correlation between the local optima of sole contribution of the decision variables and sailing distance and -duration. However, the regression of average vessel velocity showed significant results with an R^2 = 0.95. When assessing the contributions of each optimisation model, it was apparent that there is a correlation between the equal weighted results and the results of $\bar{V}_V\left(\left(w_4 = 1\right), m\right)$. This gave a p-value of 0.025, i.e. $\bar{V}_V\left(\left(w_4 = 1\right), m\right)$ has a significant effect of the values of $\bar{V}_V\left(\left(w_i = 0.25\right), m\right)$ for months $m = 1, 2, ..., 12$ and all $w_i = \left\{w_1, w_2, w_3, w_4\right\}$.

The full datasets and corresponding results of the regression analyses are attached in Appendix B.13.

Chapter 5

Discussion

In the introduction, three research questions were stated as guidelines of how digital twins can be used in coastal navigation. This chapter will discuss the success of the methodologies and results of this studies in light of the research questions. The following sections discusses the research questions below in a chronological order.

1. *What are the limiting factors of using simulation-based engineering in coastal navigation?*
2. *How can we manage fidelity in route network generation and what is the required fidelity to include routing options through limited sea space in a large-scale network?*
3. *How can we utilise digital twin technology to give decision-support in coastal navigation?*

5.1 Limiting factors of using simulation in coastal navigation

There are several limiting factors by using simulation in coastal navigation. First, simulation is often case-specific. The simulation model was set up with case-specific external data, which means that for any other case with different geography, weather area and perhaps also available weather data, the simulation model created for this project may be incompatible. Furthermore, extracting geography data from ArcGIS showed that supplementary functionality was needed, i.e. raster filtering and manual coordinate reference to WGS ’84 of the weather data frame. For further development using raster files in coastal navigation, the bathmetry should also be taken into consideration when filtering. This may require higher resolution rasters, where banks, reefs and skerries are detected by the route model.

Second, we can see that sub-functions within the created platform show uncertainty. Most recognisable is the uncertainty of our coordinate system, when considering the absolute minimum timeframe to complete a segment at a constant vessel velocity of 10 kn. This gives minimum 32 hours to complete the shortest distance between Ørland and Trondheim. By comparing the proposed optimal route of “searoutes.com” to the one generated in this project, we can see that our coordinate system is inaccurate. The optimal path from “searoutes.com” sails a seemingly longer distance with constant velocity of 9 kn, however completes the voyage in 24 hours. A possible solution to the coordinate system is to model the case area as a 3D surface rather than a 2D plane. This would increase complexity of the model and presumably give more accurate results.

Thirdly, the simulation of seastates impose uncertainties to the results. At the very least we have uncertainty in the dataset from ECMWF CDS, which is a reanalysis of predicted weather. Already when extracting netCDF format, we could see errors and uncertainties, where the correct dataset was assumed to be GRIB format. Furthermore, the filtering of NaN-values in the dataset was done by a coarse estimation. The effect of this estimation has not been investigated. Moreover, the weather data frames are large, which does not detect local variations in weather conditions, as we can expect for coastal areas. The seastates are assumed equal throughout the duration of one segment, where the generated wave timeseries iterates throughout $T_j(t)$. This simplification was needed and had a presumably small effect to the results.

Fourthly, we see that the vessel velocity of an arbitrary segment, $\bar{V}_V$, is greatly punished by the dynamic velocity model. The maximum average velocity of all months in the case analysis for all optimisation models is 6.35 kn. The cruising speed of RV Gunnerus is 9.4 kn, thus the dynamic velocity model is believed to be inaccurate.

Finally, the local optima method proposed in this project has inaccuracies. The decision variables are not normalised which gives different contributions of each optimisation variable. This was confirmed by the multiple linear regression in Section 4.3.1, where the value of average vessel velocity given $w_4 = 1$ had strong correlation to the average velocity of the equally weighted optimisation model. If all decision variables were normalised, the contributions of each $r_i(t) = \{T_j(t), \eta_3(t), \dot{\eta}_3(t), \ddot{\eta}_3(t)\}$ would be equal.

5.2 Fidelity management

The proposed route model follows the procedure of SP. After the analyses, it is apparent that the fidelity plays a significant role to what paths the vessel takes. Firstly, we consider network fidelity, where a high-fidelity network gives more alternatives and more path variation than a low-fidelity network. At glance, the procedure to find network configurations that included inshore routing was like firing bullets in the dark. To develop the route model further, efforts should be made to construct an optimisation model to find optimal fidelity while minimising computational time. The case analysis presented a wide range of path variations, much due to the high- and medium fidelity of section 2 and 3, respectively. This network partitioning seems preferable when analysing large-scale coastal areas, where the scale is governing the required fidelity.

If we were to analyse a case with a high-fidelity 1-section network, the number of segments would be intolerable with regards to computational time and pragmatic insight gained. A possible solution is to introduce subset-partitioning in areas where higher fidelity is required. If we consider the case study in this project, we can assume that all segments west of the shortest path between *ST* and *FI*, i.e. further offshore, do not require high fidelity. For the segments east of the shortest path, we can perform a subset-partitioning where the critical sections are around Rørvik and north of Rørvik. Following this procedure, the number of segments could be halved, and consequently halving the computational time.

Weather variable fidelity management was also analysed for different network configurations. The optimal weather update interval would be when $\Delta t = T_j\left(t\right)$ for all *j* and $N_{\Delta t} = N_p$, where N_p is the number of path elements in the given network. This is a fully dynamic problem, which would require additional complexity and modelling efforts. Considering the results of this analysis, a low-complexity quasi-static approach seems accurate enough.

The route optimisation model proposed in this project is only valid for networks where the available dead ends consists of maximum 4 path elements. Initially, the constraint was to check the next generation only, which did not work for the 3HF network. A possible way to solve this problem would be to filter out all nodes that are only connected to dead ends, i.e. filter the segment prior to parent nodes of dead ends. On the other hand, the optimisation model should be expanded to an algorithm-based procedure to find global optima. This would deem infeasible paths as non-optima, hence solving the issue of possible dead ends.

Computational time was analysed as an auxiliary part of the project, however played an important role in the selection of optimal fidelity configuration of the case study. The multiple-linear regression of total computational time suggests that both $N_{\Delta t}$ and N_{SEG} are significant. The MATLAB codes are iterating through both independent variables, which implies that the total number of iterations are $N_{\Delta t} * N_{SEG}$. This indicates that the computational time depends on $N_{\Delta t} * N_{SEG}$ rather than linearity. Furthermore, built-in functions are used in parts of the MATLAB platform. This has a presumably negative effect on computational when compared to logical operands.

5.3 Digital twins for decision-support in coastal navigation

The main efforts of the project have been to construct a digital platform for a route model, rather than a digital twin. The proposed response model is only included for illustrative purposes and has no relevance to simulation of the real state and behaviour of RV Gunnerus. For a complex response model, the accuracy of the simulated state and behaviour may be good enough, however we need to investigate conformance between simulated responses and actual responses to verify and validate the model.

Coastal navigation has many aspects that are hard to predict by simulation-based route modelling. Baseline digital twins could give better results while relieving modelling efforts and potentially decreasing inaccuracy. If we consider a baseline twin for RV Gunnerus, the DT could give real-time decision-support to optimal vessel heading and -velocity based on mimicking real behaviour. This would give the bridge crew active insight to how they can operate smarter, without the need of prediction of future operational contexts. The baseline DT would be a prototype for RV Gunnerus, and with growing technology maturity, additional patterns could be added to predict future states and behaviour in the proposed framework.

The functional requirements of a digital service for RV Gunnerus included maintenance. The framework in this project is created in MATLAB, however, to achieve compatibility the platform should be created in accordance to the FMI standard. This would also be beneficial for configuration management and the creation of a complete digital platform.

Chapter 6

Conclusion

The aim of this thesis has been to investigate how digital twin technology can be used in coastal route modelling. To do so, a digital framework was proposed for a route modelling approach based on set partitioning. The route model was constructed with high versatility, where altering a set of input variables could give a wide range of route alternatives. The input variables were network fidelity variables, and weather update variables. The research was successful in answering the research questions, highlighting the need for adequate fidelity management in route modelling by using a digital twin framework.

A case study of RV Gunnerus was conducted. The main results were that the proposed fidelity configuration of a 3-section network with 3 network fidelity variations gave satisfying results. Furthermore, it was concluded that for operational areas with large geography variations, a multiple-section network should be used, rather than a complete network from an origin to a destination. The model was successful in analysing seasonal variations, presenting optimal path variations, however with no clear seasonal trend. The value of a digital twin for RV Gunnerus is also discussed, where it was concluded that a baseline digital twin would be beneficial for real-time decision-support.

For further research, the main efforts should be to further develop the route model to include subset-partitioning. This would be beneficial to computational time and end-use of the service. Additionally, a more accurate optimisation model should be constructed with normalised decision variables. Moreover, the optimisation model should be algorithm-based, where the main goal is to find global optima.

References

Ansys (2020) *Ansys Twin Builder: Digital Predictive Maintenance Software*. Available at: https://www.ansys.com/products/systems/ansys-twin-builder (Accessed: 24 May 2020).

Bjørum, L. R. O. *et al.* (2018) *NTNU students build Digital Twin of RV Gunnerus during summer project*. Available at: https://www.ntnutechzone.no/en/2018/12/ntnu-students-build-digital-twin-of-rv-gunnerus-during-summer-project/ (Accessed: 20 December 2019).

Chandrasekaran, S. (2015) *Dynamic Analysis and Design of Offshore Structures*. Springer India. doi: 10.1007/978-81-322-2277-4.

Christiansen, M., Fagerholt, K. and Ronen, D. (2004) 'Ship Routing and Scheduling: Status and Perspectives', *Transportation Science*, 38(1), pp. 1–18. doi: 10.1287/trsc.1030.0036.

Copernicus Climate Change Service (C3S) (2017) *ERA5: Fifth generation of ECMWF atmospheric reanalyses of the global climate*, *Copernicus Climate Change Service Climate Data Store (CDS)*. Available at: https://cds.climate.copernicus.eu/cdsapp#!/home (Accessed: 14 April 2020).

Coraddu, A. *et al.* (2019) 'Data-driven ship digital twin for estimating the speed loss caused by the marine fouling', *Ocean Engineering*. Elsevier Ltd, 186(March), p. 106063. doi: 10.1016/j.oceaneng.2019.05.045.

Danielsen-Haces, A. (2018) 'Digital Twin Development', *Norwegian University of Science and Technology (NTNU)*, (June).

Datta, S. P. A. (2017) 'Emergence of Digital Twins - Is this the march of reason?', *Journal of Innovation Management*, 5(3), pp. 14–33. doi: 10.24840/2183-0606_005.003_0003.

Det Norske Veritas (DNV) (2010) 'DNV-RP-C205: Environmental Conditions and Environmental Loads'.

DNV GL (2020) *Veracity by DNV GL*. Available at: https://www.veracity.com/about

(Accessed: 24 May 2020).

Erikstad, S. O. (2017) ‘Merging Physics , Big Data Analytics and Simulation for the Next-Generation Digital Twins’, *Hiper’2017*, (11), pp. 140–150.

Erikstad, S. O. (2018) ‘Design patterns for digital twin solutions in marine systems’, pp. 354–363. Available at: https://www.researchgate.net/publication/325871050.

Erikstad, S. O. (2019) ‘Designing Ship Digital Services’, in *COMPIT’19*, pp. 458–469.

Esri (2011) *World Gray Canvas [basemap]*. Available at: https://www.arcgis.com/home/item.html?id=ed712cb1db3e4bae9e85329040fb9a49 (Accessed: 2 April 2020).

Farid, A. M. and Suh, N. P. (2016) *Axiomatic Design in Large Systems: Complex Products, Buildings and Manufacturing Systems*. Cham: Springer International Publishing.

Gamma, E. *et al.* (1994) *Design patterns: elements of reusable object-oriented software*. Reading, Mass: Addison-Wesley (Addison-Wesley professional computing series).

Google (2020) *Google Maps directions for driving from Oslo to Trondheim*. Available at: https://www.google.no/maps/ (Accessed: 25 May 2020).

Gudmestad, O. T. (2015) *Marine technology and operations : theory & practice*. Southampton: WIT Press.

Hasselmann, K. *et al.* (1973) *Measurements of wind-wave growth and swell decay during the joint North Sea wave project (JONSWAP)*. Hamburg: Deutsches Hydrographisches Institut (Ergänzungsheft zur deutschen hydrographischen Zeitschrift. Reihe A).

Johansen, S. S. and Nejad, A. R. (2019) ‘On digital twin condition monitoring approach for drivetrains in marine applications’, *Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE*, 10, pp. 1–10. doi: 10.1115/omae2019-95152.

Kirk, J. (2020) ‘Dijkstra’s Shortest Path Algorithm’. MATLAB Central File Exchange. Retrieved April 14 2020. Available at: https://www.mathworks.com/matlabcentral/fileexchange/12850-dijkstra-s-shortest-path-algorithm.

Kongsberg Maritime (2020) *K-Bridge ECDIS Display*. Available at: https://www.kongsberg.com/maritime/products/bridge-systems-and-control-centres/navigation-system/ecdis-electronic-chart-display-system/#downloads (Accessed: 9 June 2020).

Merigaud, A. and Ringwood, J. (2017) 'Free-Surface Time-Series Generation for Wave Energy Applications', *IEEE Journal of Oceanic Engineering*, PP, pp. 1–17. doi: 10.1109/JOE.2017.2691199.

NTNU (2019) *R/V Gunnerus*. Available at: https://www.ntnu.edu/oceans/gunnerus (Accessed: 20 December 2019).

Oslo School of Architecture and Design (2020) *Open Bridge Design*. Available at: http://www.openbridge.no/ (Accessed: 8 April 2020).

OSP (2020) *Open Simulation Platform*. Available at: https://opensimulationplatform.com/ (Accessed: 24 May 2020).

Pomerantz, D. (2018) *The French Connection: Digital Twins From Paris Will Protect Wind Turbines Against Battering North Atlantic Gales*. Available at: https://www.ge.com/news/reports/french-connection-digital-twins-paris-will-protect-wind-turbines-battering-north-atlantic-gales (Accessed: 24 May 2020).

Sandvik, E. (2019) *Sea Passage Scenario Simulation for Ship System Performance Evaluation*. NTNU.

Searoutes (2020) *searoutes.com*. Available at: searoutes.com (Accessed: 9 June 2020).

Suh, N. P. (2001) 'Axiomatic design : advances and applications'. New York: Oxford University Press (The MIT-Pappalardo series in mechanical engineering).

Vazirizade, S. M. (2020) 'WaveModeling'. GitHub. Retrieved 20 April 2020. Available at: https://github.com/smvazirizade/WaveModeling.

Wang, K. J., Lee, T. L. and Hsu, Y. (2020) 'Revolution on digital twin technology—a patent research approach', *International Journal of Advanced Manufacturing Technology*. The International Journal of Advanced Manufacturing Technology, 107(11–12), pp. 4687–4704. doi: 10.1007/s00170-020-05314-w.

Zheng, Y., Yang, S. and Cheng, H. (2019) 'An application framework of digital twin and its case study', *Journal of Ambient Intelligence and Humanized Computing*. Berlin/Heidelberg: Springer Berlin Heidelberg, 10(3), pp. 1141–1153.

Appendices

Appendix A

Computational time regression analyses

This appendix presents datasets that were not included in the main text of the regression of N_{SEG} and computational time. Appendices A.1 and A.2 presents the regression of large extent networks and multiple section networks, respectively, with route generation only, i.e. $T_{C,RG}$. Appendix A.3 presents the regression for the total computational time $\left(T_{C,tot}\right)$.

A.1 Large extent network with route generation only

Table A.1: Different configurations of large extent networks

d	***S***	***H***	N_{NOD}	N_{SEG}	$T_{C,RG}$ **[s]**
200	10	10	233	1,804	5.0
200	10	20	483	7,515	10.5
200	30	20	1,890	32,886	38.5
250	30	20	1,568	26,862	33.9
300	40	7	573	3,352	6.6
300	50	20	2,305	39,141	58.3
400	15	20	493	7,762	11.0
500	12	15	233	2,562	5.5
500	10	7	79	390	3.2
600	40	5	197	710	3.7

Linear regression with the dataset above gave an R^2-value of 0.9757 and the equation presented below. The equation slightly deviates from the one found for regression of medium- and small extent networks. There are two probable reasons for this: (1) large extent networks generates more segments that are filtered out, i.e. $N_{SEG}^{1'} / N_{SEG}^{1} > N_{SEG}^{2'} / N_{SEG}^{2}$ where $N_{SEG}^{1'} / N_{SEG}^{1}$ is the ratio between unfiltered and filtered segments of a large extent network, and

$N_{SEG}^{2'} / N_{SEG}^{2}$ is the ratio of a small- or medium extent network. This is simply because more of the network is located on land. And (2), rounding errors in MS Excel.

$$T_{C,RG}\left(N_{SEG}\right) = 0.0013N_{SEG} + 1.9$$

Figure A.1 presents the plot regression graph and the datapoints of Table A.1.

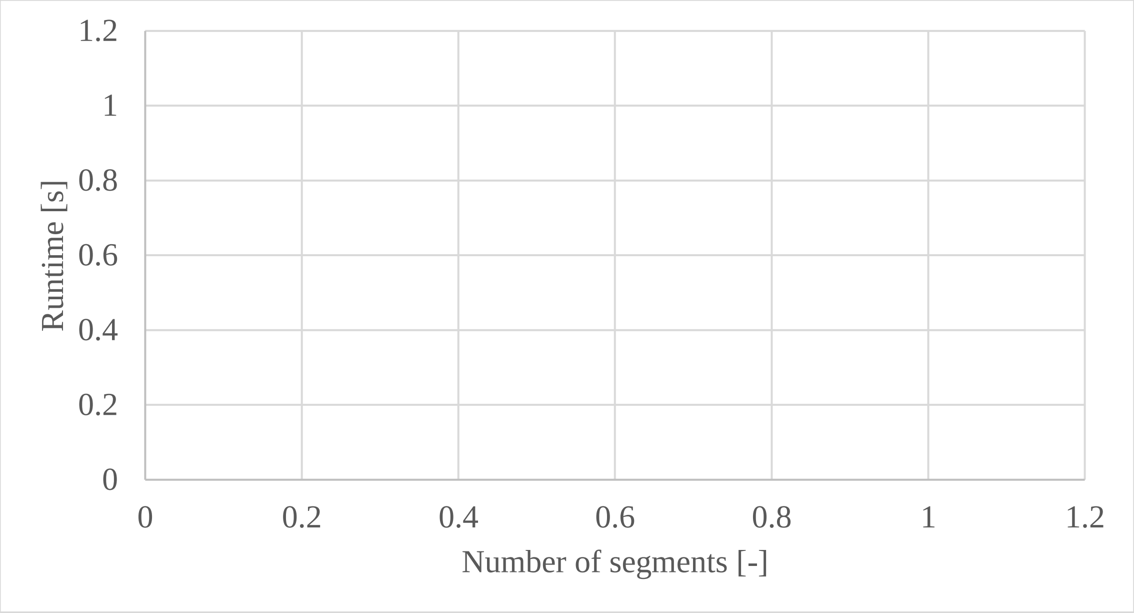

Figure A.1: Plot of the linear regression of large extent networks

A.2 Network using 2 waypoints and 3 sections

The full dataset of the route generation fidelity analysis using networks with 3 sections are presented here. Following, the results from the linear regression is presented.

The route generation variables are here given in vector format, where space is the element separator. Each element within a vector corresponds to the variable in the respective section.

Table A.2: Full dataset used in the regression of a network with 3 sections

S	*H*	***d*** **[px]**	N_{NOD}	N_{SEG}	**$T_{C,RG}$ [s]**
1 3 5	5 7 3	150 300 400	48	129	3.3
1 3 5	5 7 3	150 250 200	53	142	3.4
1 3 6	5 7 3	150 150 200	56	130	3.2
1 15 6	5 7 3	150 50 200	374	1988	5.2

1 15 15	5 7 3	150 50 100	476	2268	4.9
1 15 15	5 7 3	150 50 80	483	2291	5.1
1 15 20	5 7 3	150 50 70	568	2518	5.2
1 15 20	5 5 3	150 50 70	459	1531	4.9
1 15 8	5 5 5	10 50 200	316	1248	3.9
1 15 6	5 5 5	150 50 200	287	1118	4

The dataset in the table above gave the following linear regression expression with an R^2-value of 0.9210:

$$T_{C,RG}\left(N_{SEG}\right)=0.0008N_{SEG}+3.2$$

The regression line and datapoints are presented in the diagram below.

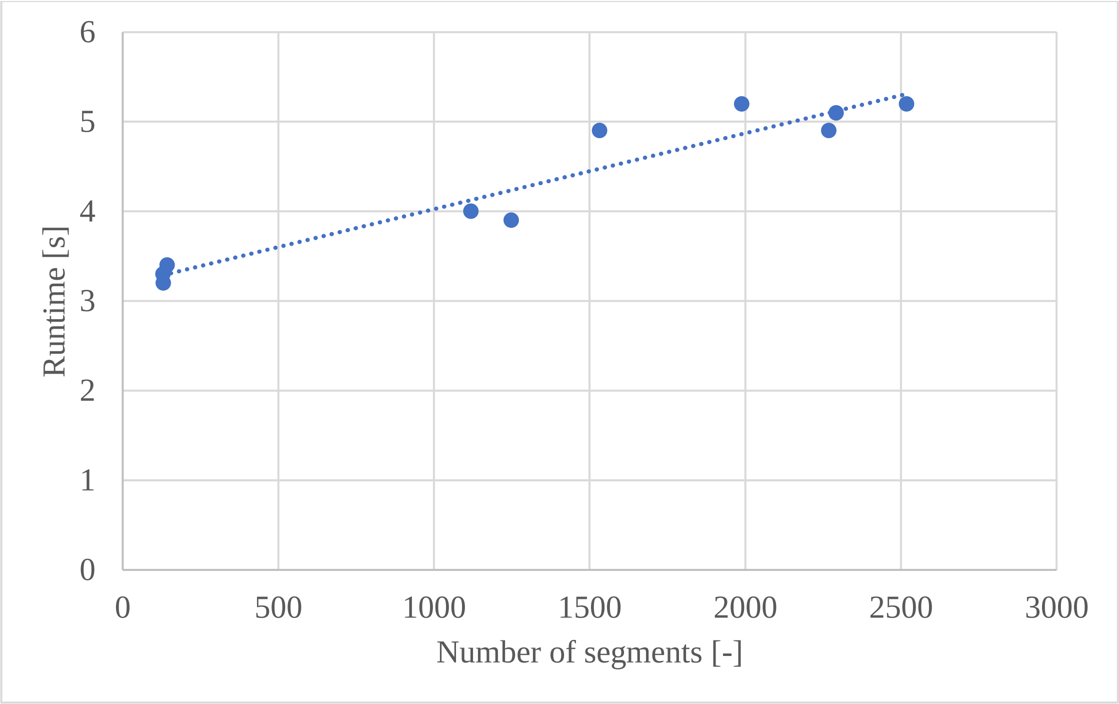

Figure A.2: Plot of the linear regression of a network with 3 sections

A.3 Total computational time

For the total computational time, the running time of segment weather, responses, optimisation and results printing and plotting are included. This appendix presents the dataset used in the multiple linear regression in Table A.3. Following, the results from the regression analysis in MS Excel are presented in Table A.4.

Table A.3: Dataset used in the multiple linear regression of computational time, number of weather updates and number of segments

Network	**Weather variation**	$N_{\Delta t}$ **[-]**	N_{SEG} **[-]**	**Runtime [s]**
1LF	1	101	54	30
1LF	2	51	54	17
1LF	3	26	54	10
1LF	4	18	54	8
1HF	1	101	634	277
1HF	2	51	634	140
1HF	3	26	634	74
1HF	4	18	634	53
3LF	1	101	130	60
3LF	2	51	130	32
3LF	3	26	130	19
3LF	4	18	130	15
3HF	1	101	1,531	663
3HF	2	51	1,531	347
3HF	3	26	1,531	182
3HF	4	18	1,531	129

Table A.4: Regression analysis report of the multiple linear regression from MS Excel

Regression Statistics	
Multiple R	0.8865
R Square	0.7859
Adjusted R Square	0.6992
Standard Error	104.9707
Observations	16

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	2	566223.5775	283111.8	25.6934	3.04405E-05
Residual	14	154263.9325	11018.85		
Total	16	720487.51			

	Coeff.	*Std. Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	0	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
Ndt	1.1942	0.5529	2.1599	0.0486	0.0084	2.3800	0.0084	2.3800
Nseg	0.1660	0.0391	4.2508	0.0008	0.0823	0.2498	0.0823	0.2498

Appendix B

Case mission results

This appendix presents all plots and paths of optimal routes by month in a chronological order. Finally, the full datasets of the multiple linear regression of the decision variables are presented.

B.1 January

Table B.1: Optimal paths of January of the different local optima optimisation models (OM)

OM	**Path**
w_i = 0.25	1 3 7 10 16 27 41 60 82 108 139 170 198 223 243 257 268 274 279 281 285 292 300 310 323 337 353 371 391 413 434 453 470 484 495 504 511 516 519 520
w_1 = 1	1 3 7 10 16 29 46 66 91 120 152 182 209 234 251 264 273 278 279 280 284 290 299 310 322 336 353 371 391 413 434 451 467 481 494 503 510 516 519 520
w_2 = 1	1 5 7 11 16 26 42 60 82 108 139 170 195 216 236 253 266 274 279 282 286 292 300 309 322 336 351 370 389 411 431 449 464 478 489 499 508 513 517 520
w_3 = 1	1 3 7 10 16 29 46 66 91 120 151 181 209 231 250 263 273 278 279 280 284 291 298 308 321 336 352 371 392 413 433 451 466 481 492 503 510 516 519 520
w_4=1	1 3 7 9 15 25 39 57 79 105 134 163 189 214 235 252 265 274 279 281 285 291 299 309 321 335 351 369 389 410 430 448 464 478 490 500 508 514 517 520

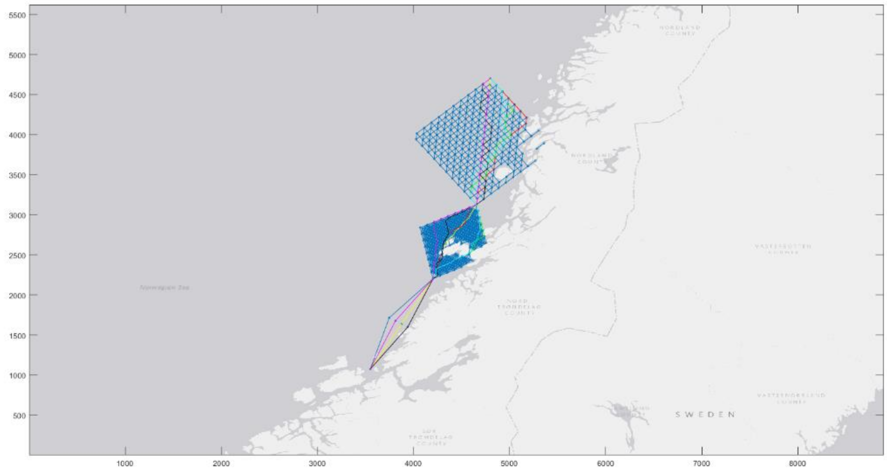

Figure B.1: Plot of optimal paths of January

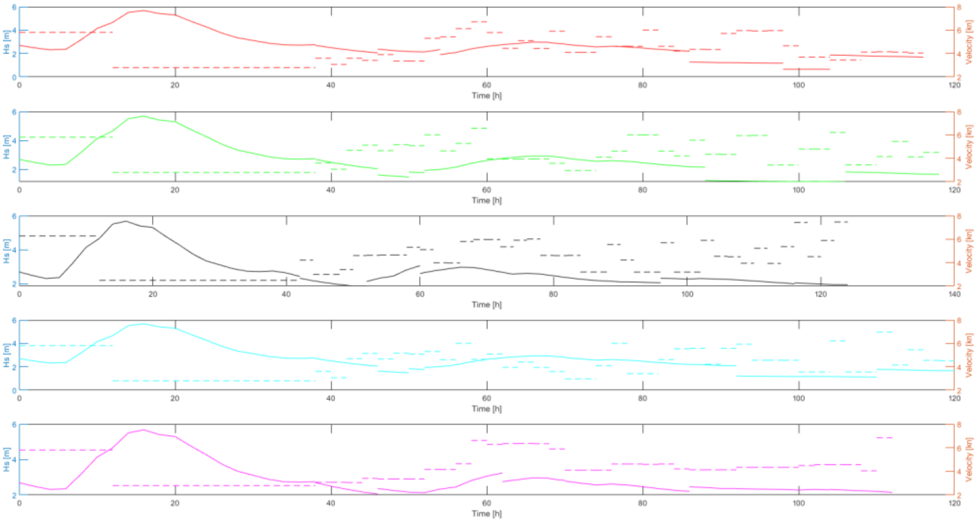

Figure B.2: Vessel velocity and H_s of January

B.2 February

Table B.2: Optimal paths of February of the different local optima optimisation models (OM)

OM	Path
w_i = *0.25*	1 5 7 10 18 29 45 66 91 120 152 184 211 234 251 264 273 278 279 282 287 293 301 311 323 338 354 373 393 415 435 454 470 484 495 504 511 516 519 520

w₁ = 1	1 5 7 10 16 29 45 66 91 120 152 182 209 234 251 264 273 278 279 280 284 290 299 309 321 334 351 369 390 412 433 452 467 482 494 504 511 516 519 520
w₂ = 1	1 5 7 11 19 29 45 66 91 120 150 179 204 226 243 257 266 275 279 282 286 293 302 313 326 341 357 375 394 414 433 452 467 480 492 503 511 516 519 520
w₃ = 1	1 3 7 10 16 29 45 66 91 120 151 181 209 231 248 263 273 278 279 282 287 294 303 313 326 339 355 372 391 413 434 451 467 482 495 504 511 516 519 520
w₄=1	1 4 7 10 17 27 42 60 82 109 140 171 197 219 237 252 265 274 279 281 285 291 299 309 321 335 351 369 389 410 430 448 463 476 488 498 506 512 517 520

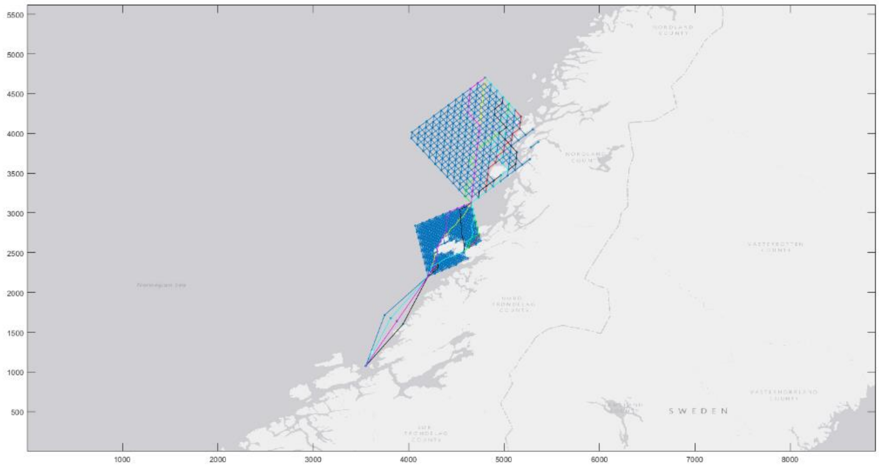

Figure B.3: Optimal paths of February

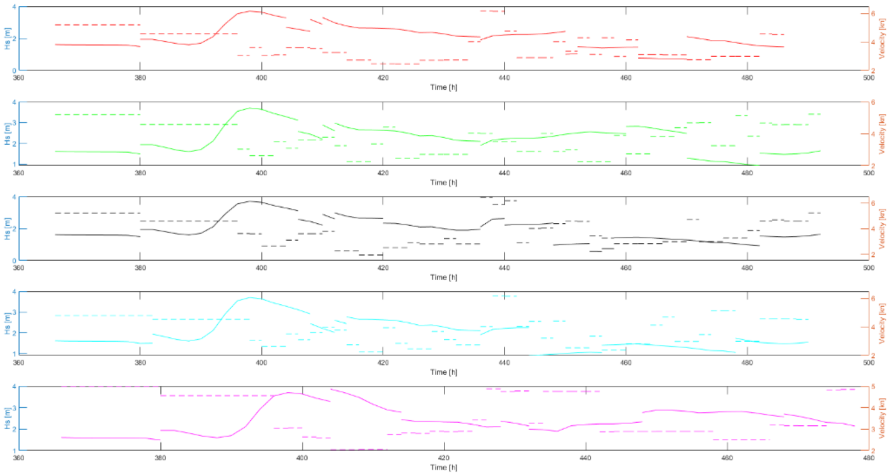

Figure B.4: Vessel velocity and H_s of February

B.3 March

Table B.3: Optimal paths of March of the different local optima optimisation models (OM)

OM	Path
w_i = *0.25*	1 2 7 11 16 26 40 60 82 107 137 169 197 219 236 253 266 275 279 282 285 291 298 309 321 334 350 368 388 409 429 448 465 479 491 502 511 516 519 520
w_1 = *1*	1 2 7 11 16 27 40 58 81 107 140 170 195 220 238 254 265 274 279 282 285 292 300 309 320 335 350 368 388 409 430 449 464 477 488 499 508 515 519 520
w_2 = *1*	1 5 7 12 17 30 47 66 91 120 150 182 210 231 249 264 273 278 279 280 285 291 300 309 321 335 350 368 388 410 429 448 464 478 491 502 511 516 519 520
w_3 = *1*	1 2 7 8 16 26 39 58 81 107 137 167 196 217 236 252 265 274 279 282 285 290 299 309 320 335 351 369 388 408 429 448 465 478 489 498 505 512 517 520
w_4=*1*	1 2 7 8 13 22 35 52 73 98 127 160 189 214 235 252 265 274 279 280 283 288 295 304 315 328 343 360 379 400 421 440 457 472 485 496 505 512 517 520

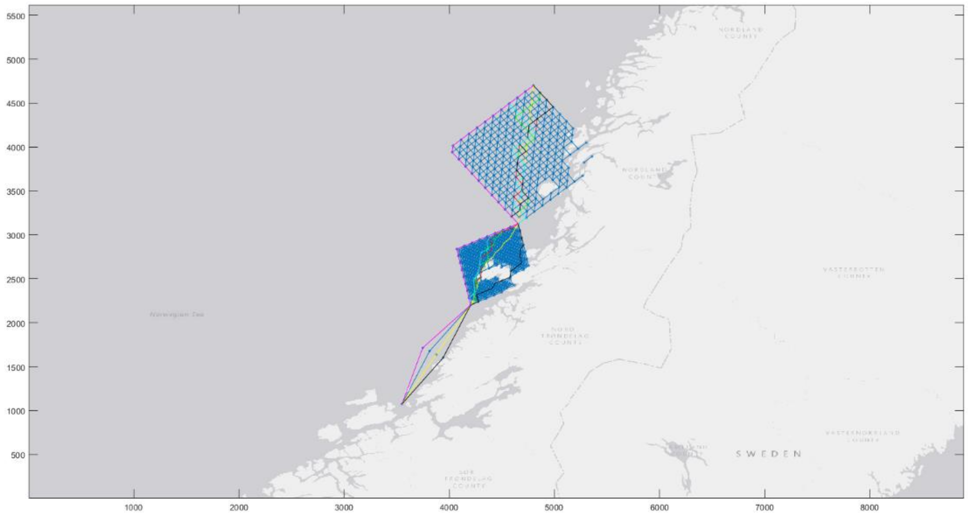

Figure B.5: Optimal paths of March

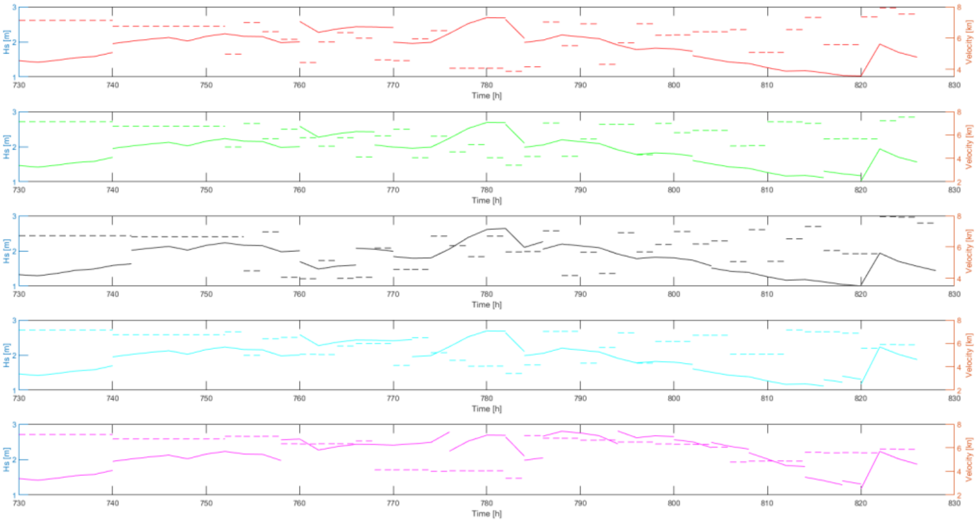

Figure B.6: Vessel velocity and H_s of March

B.4 April

Table B.4: Optimal paths of April of the different local optima optimisation models (OM)

OM	**Path**
$w_i = 0.25$	1 5 7 10 16 27 42 60 82 108 139 171 200 223 243 257 268 276 279 280 284 290 298 309 320 334 351 370 389 411 432 450 466 480 492 503 510 515 519 520
$w_1 = 1$	1 5 7 10 16 29 45 66 91 120 152 183 209 233 250 263 272 277 279 280 284 290 299 310 322 336 353 371 391 413 434 451 467 481 494 503 510 515 519 520
$w_2 = 1$	1 5 7 11 18 29 45 66 91 120 150 183 210 234 251 264 273 278 279 282 286 292 300 309 322 336 351 370 389 411 431 449 466 480 492 503 510 516 519 520
$w_3 = 1$	1 3 7 10 16 29 45 66 91 120 151 181 207 228 245 262 271 277 279 282 287 293 302 313 326 341 357 375 394 415 435 452 469 483 495 504 511 516 519 520
w_4=1	1 5 7 9 15 25 39 57 79 105 135 164 189 214 235 252 265 274 279 280 283 288 295 304 315 328 343 360 379 400 421 440 457 472 485 496 505 512 517 520

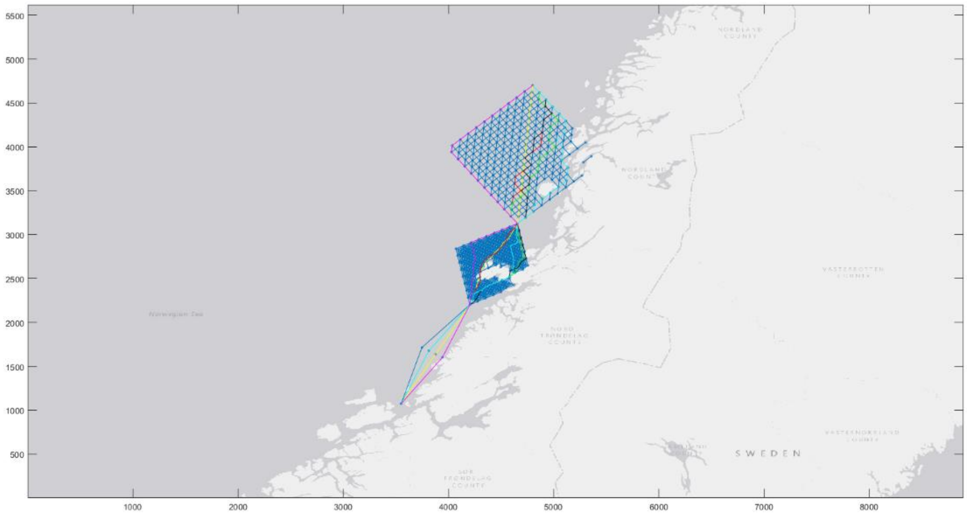

Figure B.7: Optimal paths of April

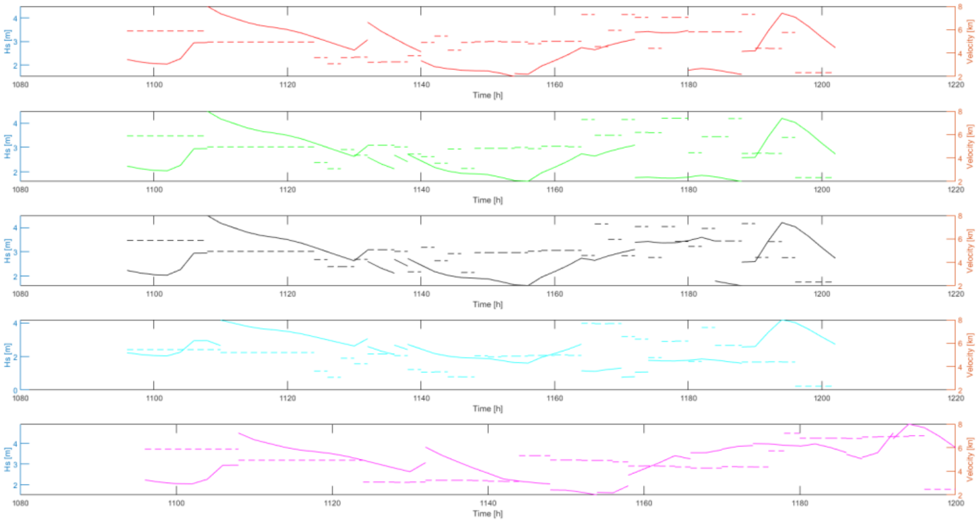

Figure B.8: Vessel velocity and H_s of April

B.5 May

Table B.5: Optimal paths of May of the different local optima optimisation models (OM)

OM	**Path**
$w_i = 0.25$	1 3 7 10 18 29 45 66 91 120 152 184 211 234 251 264 273 278 279 280 284 290 298 309 322 336 353 371 392 413 434 452 468 482 494 504 511 516 519 520

$w_1 = 1$	1 5 7 9 16 27 41 58 81 107 139 168 195 219 238 255 265 274 279 280 284 290 299 310 322 336 353 371 392 414 433 452 468 483 495 504 511 516 519 520
$w_2 = 1$	1 5 7 11 19 29 45 66 91 120 150 179 204 226 243 257 267 274 279 282 286 292 300 309 322 337 354 373 394 414 433 450 466 480 492 503 510 516 519 520
$w_3 = 1$	1 3 7 10 16 29 45 66 91 120 151 181 209 231 248 263 273 278 279 282 287 293 302 313 326 339 354 373 392 413 434 451 466 481 492 503 510 515 519 520
$w_4=1$	1 2 7 8 13 22 35 52 73 98 128 160 189 214 235 252 265 274 279 280 283 288 296 306 318 332 348 366 386 407 427 445 461 475 487 497 505 512 517 520

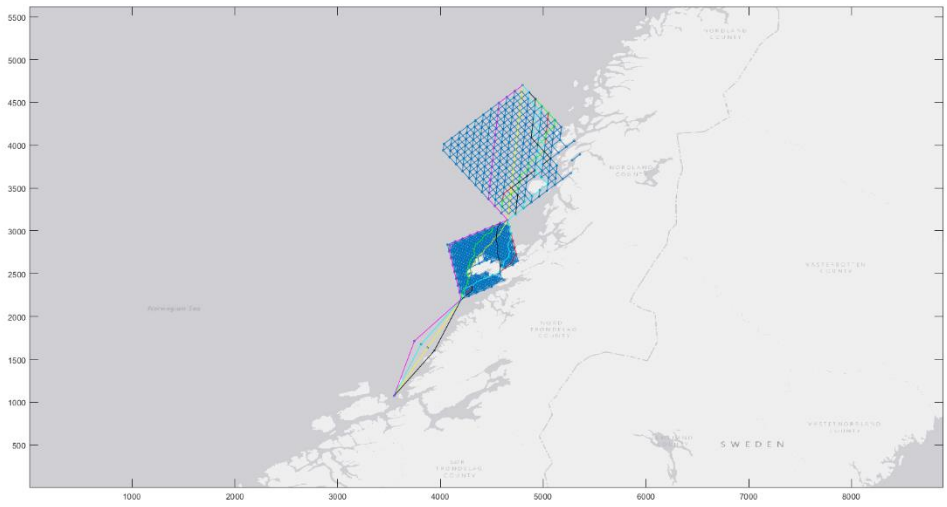

Figure B.9: Optimal paths of May

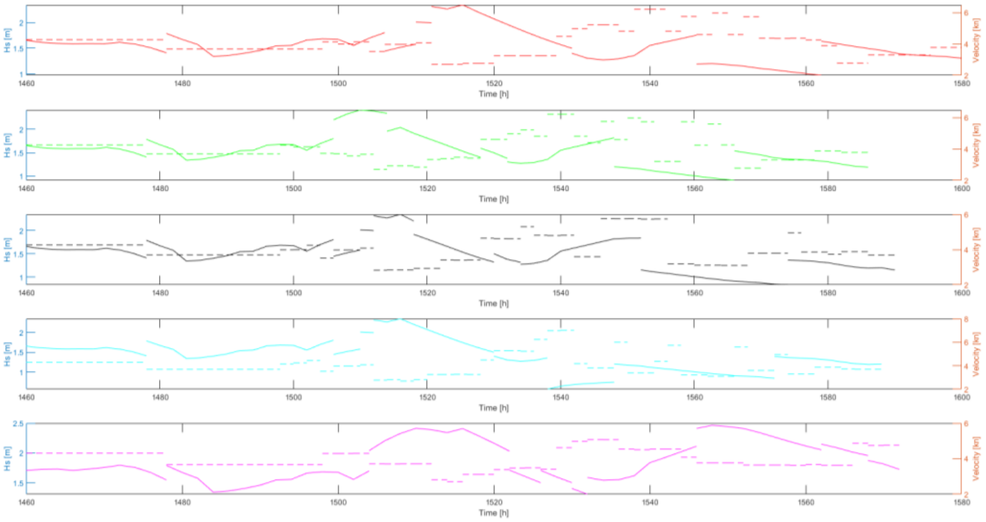

Figure B.10: Vessel velocity and H_s of May

B.6 June

Table B.6: Optimal paths of June of the different local optima optimisation models (OM)

OM	Path
$w_i = 0.25$	1 5 7 10 18 31 45 66 91 120 152 181 209 234 251 264 273 278 279 280 284 290 298 309 322 336 353 371 392 413 434 452 468 482 494 504 511 516 519 520
$w_1 = 1$	1 5 7 10 18 31 46 66 91 120 152 182 209 234 251 264 273 278 279 280 284 290 299 310 322 336 353 371 392 414 433 452 467 482 494 504 511 516 519 520
$w_2 = 1$	1 5 7 11 18 27 41 58 82 108 139 170 199 223 244 257 267 275 279 280 284 290 299 310 322 337 354 373 394 414 433 452 467 480 492 503 511 516 519 520
$w_3 = 1$	1 3 7 8 14 27 40 58 81 109 140 171 197 222 243 260 269 274 279 280 284 291 299 308 321 334 351 369 390 410 430 449 464 477 488 499 507 512 517 520
$w_4=1$	1 4 7 8 13 22 35 52 73 98 127 160 189 214 235 252 265 274 279 280 283 289 297 307 319 333 349 367 387 408 428 446 462 476 488 498 506 512 517 520

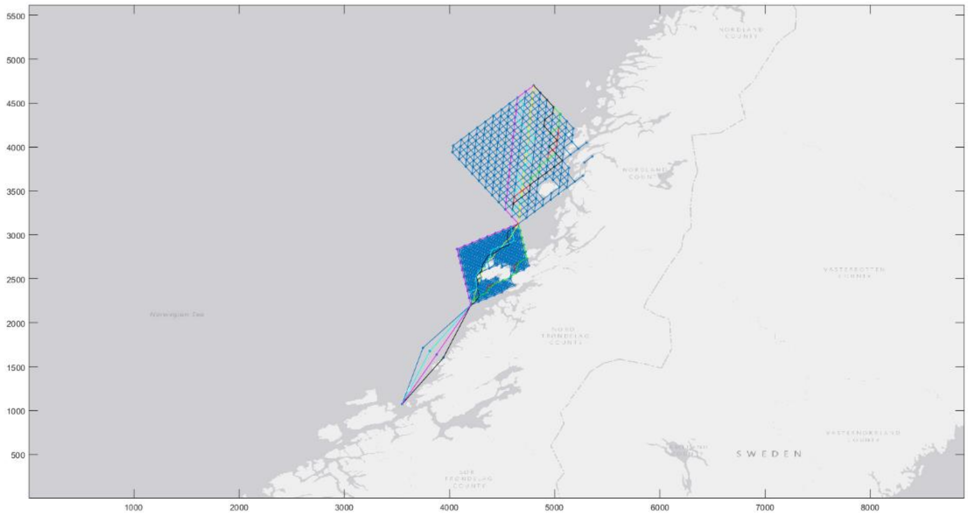

Figure B.11: Optimal paths of June

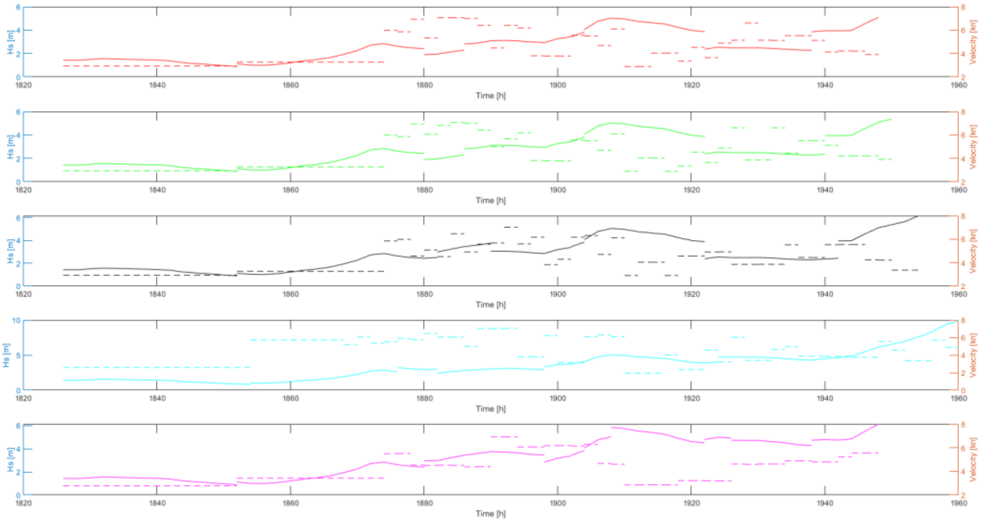

Figure B.12: Vessel velocity and H_s of June

B.7 July

Table B.7: Optimal paths of July of the different local optima optimisation models (OM)

OM	Path
$w_i = 0.25$	1 3 7 11 16 26 40 57 82 107 140 169 196 220 239 256 268 276 279 280 284 290 299 308 320 334 350 368 387 408 429 446 463 476 489 500 508 513 518 520
$w_1 = 1$	1 5 7 9 18 31 46 67 92 120 150 183 209 233 250 263 273 278 279 280 284 290 299 310 321 335 350 368 389 411 430 447 462 477 488 499 507 512 517 520
$w_2 = 1$	1 5 7 12 17 27 43 60 82 108 140 173 199 223 244 257 267 274 279 282 286 293 302 313 326 339 355 374 394 415 435 453 468 483 495 504 511 516 519 520
$w_3 = 1$	1 3 7 12 19 31 46 67 92 120 150 179 206 229 247 261 272 278 279 282 285 292 300 309 321 334 350 369 389 411 430 449 464 477 488 499 507 512 517 520
w_4=1	1 2 7 8 13 22 35 52 73 98 127 160 189 214 235 252 265 274 279 280 283 288 295 304 315 328 343 360 379 400 421 440 457 472 485 496 505 512 517 520

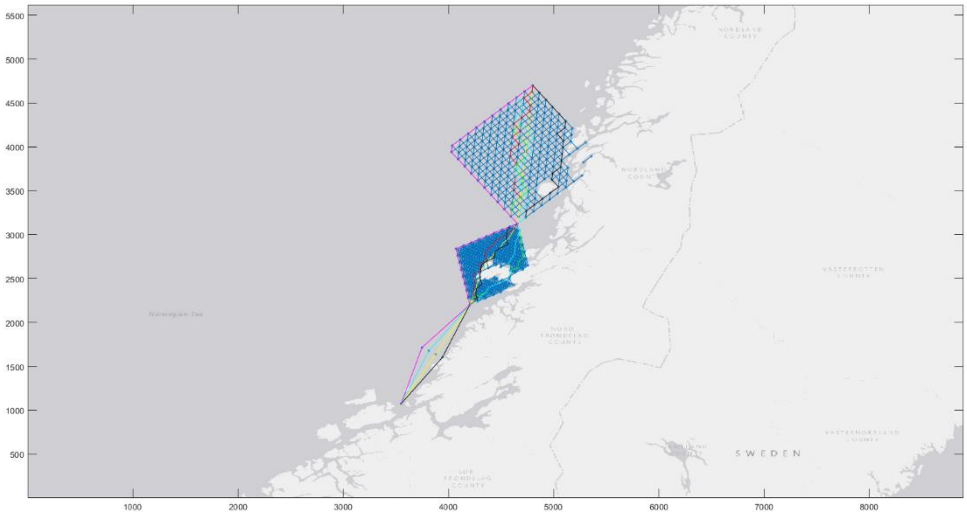

Figure B.13: Optimal paths of July

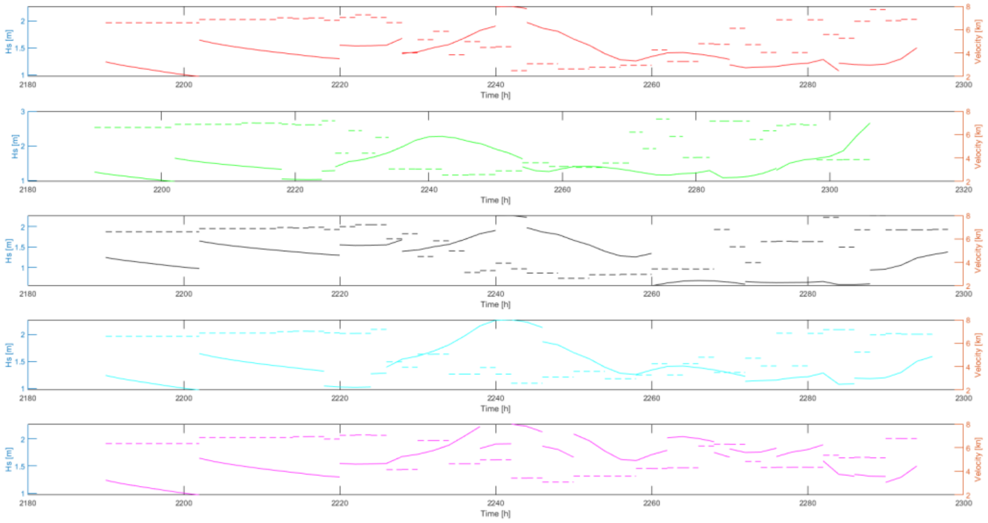

Figure B.14: Vessel velocity and H_s of July

B.8 August

Table B.8: Optimal paths of August of the different local optima optimisation models (OM)

OM	**Path**
$w_i = 0.25$	1 3 7 10 16 27 40 60 81 109 139 171 199 223 243 259 271 276 279 280 284 290 299 310 322 335 350 368 388 409 430 447 464 478 489 500 508 513 518 520

w₁ = 1	1 2 7 10 16 29 45 66 91 120 150 179 205 227 244 260 270 276 279 280 284 291 299 309 321 334 351 369 390 410 431 449 465 479 491 502 510 515 519 520
w₂ = 1	1 5 7 11 19 29 45 66 91 120 151 181 206 228 248 262 273 278 279 280 284 290 298 308 320 335 350 368 388 410 429 448 464 478 489 499 508 513 518 520
w₃ = 1	1 2 7 10 16 29 45 66 91 120 150 182 210 234 251 264 273 278 279 280 284 290 299 309 321 334 351 369 389 411 430 449 466 481 494 504 511 516 519 520
w₄=1	1 2 7 8 13 22 35 55 79 107 139 171 199 223 243 259 271 276 279 280 283 288 295 304 315 328 343 360 379 400 421 440 457 472 485 496 505 512 517 520

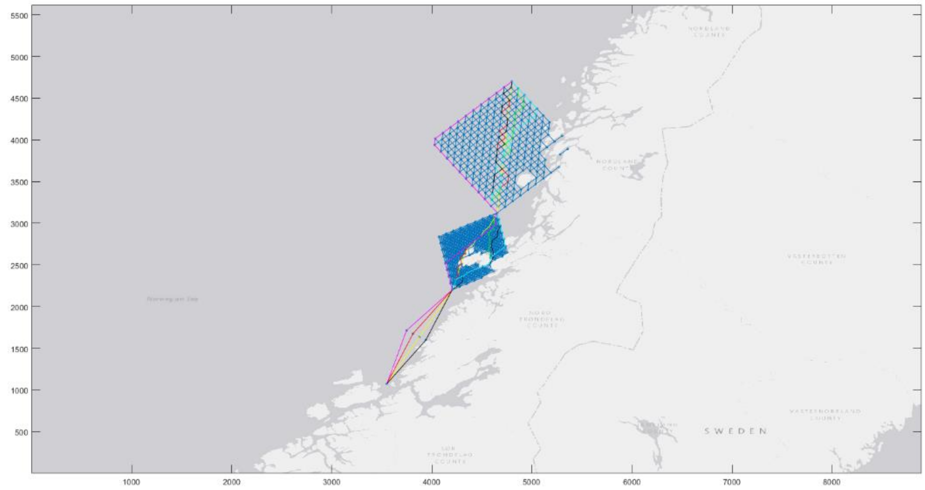

Figure B.15: Optimal paths of August

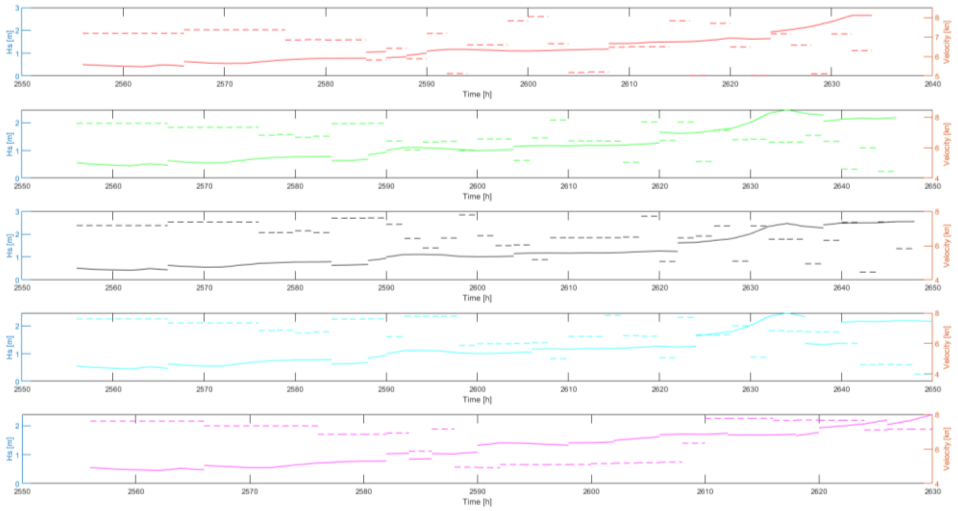

Figure B.16: Vessel velocity and H_s of August

B.9 September

Table B.9: Optimal paths of September of the different local optima optimisation models (OM)

OM	Path
$w_i = 0.25$	1 5 7 11 19 31 45 66 91 120 152 181 209 234 251 264 273 278 279 281 285 291 299 309 321 335 351 369 389 410 430 448 464 477 488 499 507 514 519 520
$w_1 = 1$	1 5 7 9 18 29 46 66 91 120 152 182 209 234 251 264 273 278 279 280 284 290 299 309 321 334 351 369 390 410 430 448 465 479 491 502 510 515 519 520
$w_2 = 1$	1 5 7 12 17 27 41 58 82 108 139 170 199 221 242 256 265 274 279 282 286 292 301 311 323 338 353 371 391 413 432 451 467 480 492 503 511 516 519 520
$w_3 = 1$	1 3 7 12 19 31 48 67 91 120 151 181 209 231 248 263 273 278 279 282 287 293 302 313 326 339 354 373 392 413 434 451 467 482 495 504 511 516 519 520
$w_4=1$	1 4 7 9 15 25 39 57 79 105 134 163 189 214 235 252 265 274 279 281 285 291 299 309 321 335 351 369 389 410 430 448 464 478 489 498 505 512 517 520

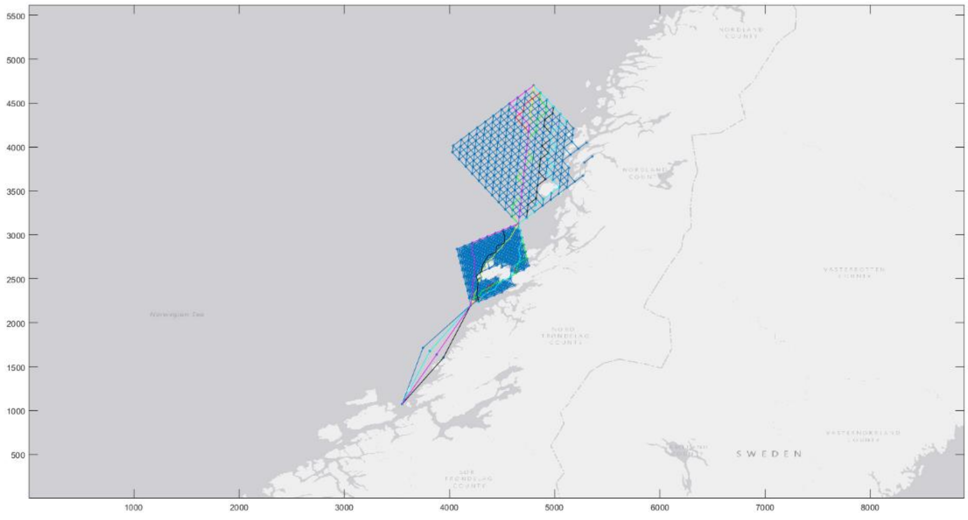

Figure B.17: Optimal paths of September

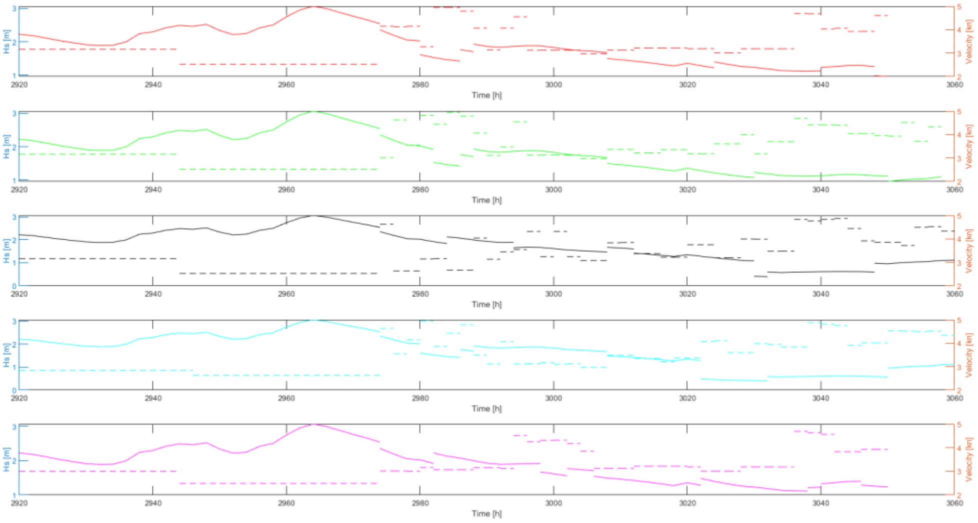

Figure B.18: Vessel velocity and H_s of September

B.10 October

Table B.10: Optimal paths of October of the different local optima optimisation models (OM)

OM	Path
$w_i = 0.25$	1 3 7 10 16 27 40 57 81 108 139 170 198 223 243 258 267 276 279 280 284 290 299 310 322 336 353 371 392 412 432 450 466 480 492 502 510 516 519 520
$w_1 = 1$	1 2 7 10 16 29 45 66 91 120 152 182 209 230 248 264 273 278 279 280 284 290 299 309 321 335 350 368 387 408 428 447 464 477 488 499 507 514 519 520
$w_2 = 1$	1 5 7 11 19 29 45 66 91 120 150 179 204 226 243 257 268 274 279 280 284 290 299 308 320 335 350 368 389 410 429 448 464 478 489 499 508 513 518 520
$w_3 = 1$	1 2 7 10 16 29 45 66 91 120 151 181 209 233 250 263 273 278 279 280 284 291 299 309 321 336 352 371 392 414 434 453 468 482 495 504 511 516 519 520
$w_4=1$	1 2 7 8 13 22 35 52 73 98 127 160 189 214 235 252 265 274 279 280 283 288 295 304 315 328 343 360 379 400 421 440 457 472 485 496 505 512 517 520

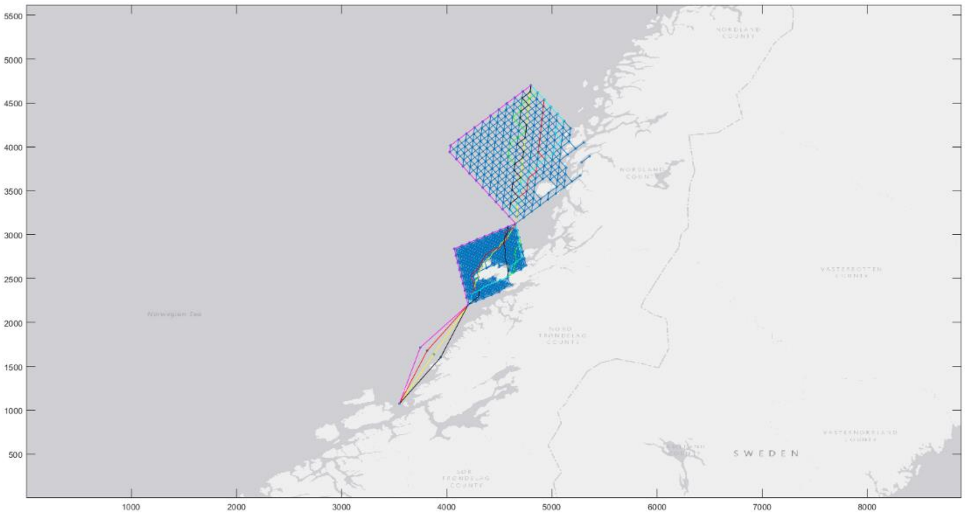

Figure B.19: Optimal paths of October

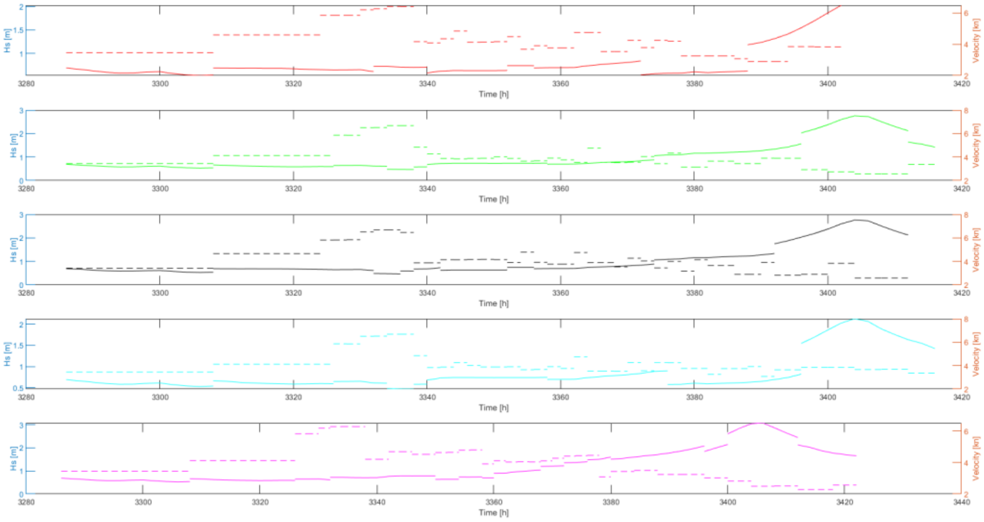

Figure B.20: Vessel velocity and H_s of October

B.11 November

Table B.11: Optimal paths of November of the different local optima optimisation models (OM)

OM	**Path**
$w_i = 0.25$	1 5 7 10 16 27 40 57 82 107 139 170 195 219 240 255 267 276 279 280 284 290 298 309 320 334 350 369 390 410 429 448 464 479 491 502 511 516 519 520

w_1 = 1	1 2 7 10 16 29 46 66 91 120 150 179 205 227 244 257 268 275 279 280 284 290 299 310 322 336 353 371 391 413 434 451 467 481 494 503 510 515 519 520
w_2 = 1	1 5 7 11 18 29 45 66 91 120 151 183 210 231 250 264 273 278 279 282 286 292 300 309 322 336 351 370 389 411 431 449 466 480 492 503 510 516 519 520
w_3 = 1	1 2 7 10 16 29 46 66 91 120 150 179 205 227 244 257 268 275 279 282 287 293 302 313 326 341 357 375 394 415 435 452 469 483 495 504 511 516 519 520
w_4=1	1 5 7 8 13 22 35 52 73 98 127 160 189 214 235 252 265 274 279 280 283 288 295 304 315 328 343 360 379 400 421 440 457 472 485 496 505 512 517 520

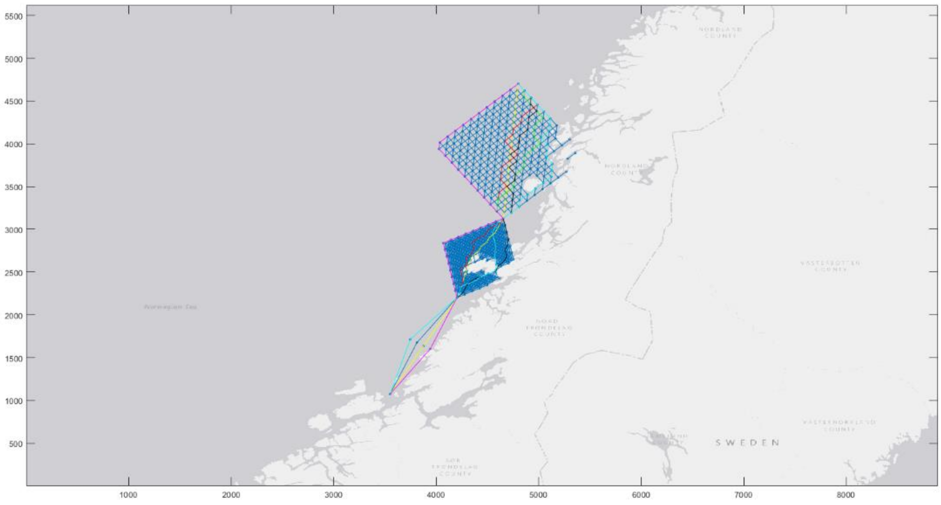

Figure B.21: Optimal paths of November

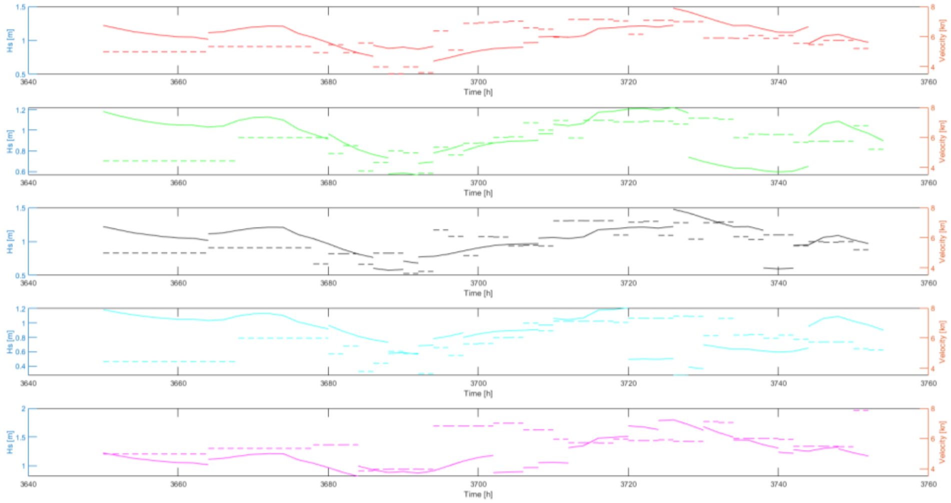

Figure B.22: Vessel velocity and H_s of November

B.12 December

Table B.12: Optimal paths of December of the different local optima optimisation models (OM)

OM	Path
$w_i = 0.25$	1 5 7 11 20 31 46 67 92 120 151 182 209 232 251 264 273 278 279 280 284 291 299 308 320 334 351 369 389 411 430 447 464 478 489 499 507 514 518 520
$w_1 = 1$	1 2 7 8 15 26 39 58 82 107 137 169 196 219 240 253 265 274 279 280 284 290 299 309 321 334 351 369 390 410 430 449 464 477 488 499 507 512 517 520
$w_2 = 1$	1 5 7 11 19 29 42 60 81 108 140 169 196 220 240 255 265 274 279 280 284 290 299 308 320 335 350 368 388 410 429 448 464 478 489 499 508 513 517 520
$w_3 = 1$	1 3 7 9 16 26 43 60 82 107 140 169 196 221 242 255 267 276 279 280 284 291 299 309 321 334 351 369 389 411 430 447 464 478 490 499 507 514 517 520
$w_4=1$	1 5 7 8 13 22 35 52 73 98 129 160 189 214 235 252 265 274 279 280 283 288 295 304 315 328 343 360 379 400 421 440 457 472 485 496 505 512 517 520

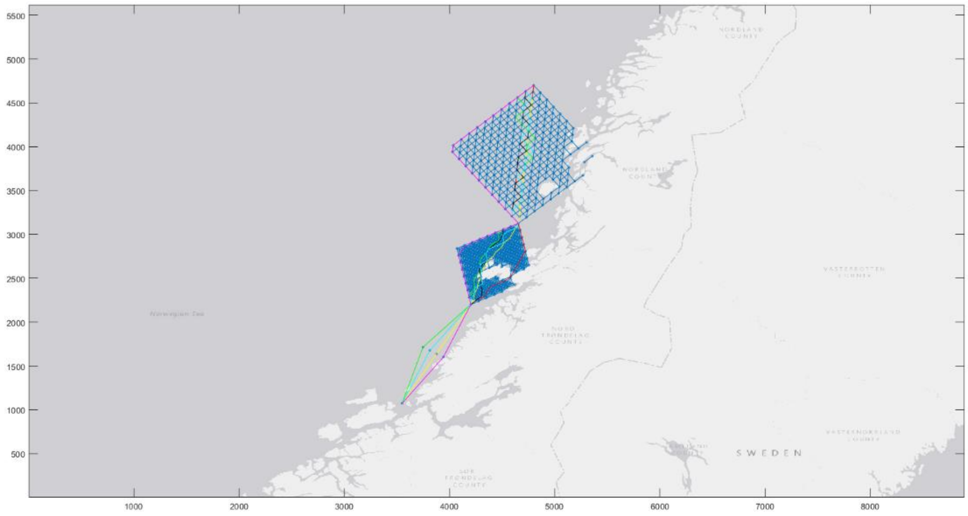

Figure B.23: Optimal paths of December

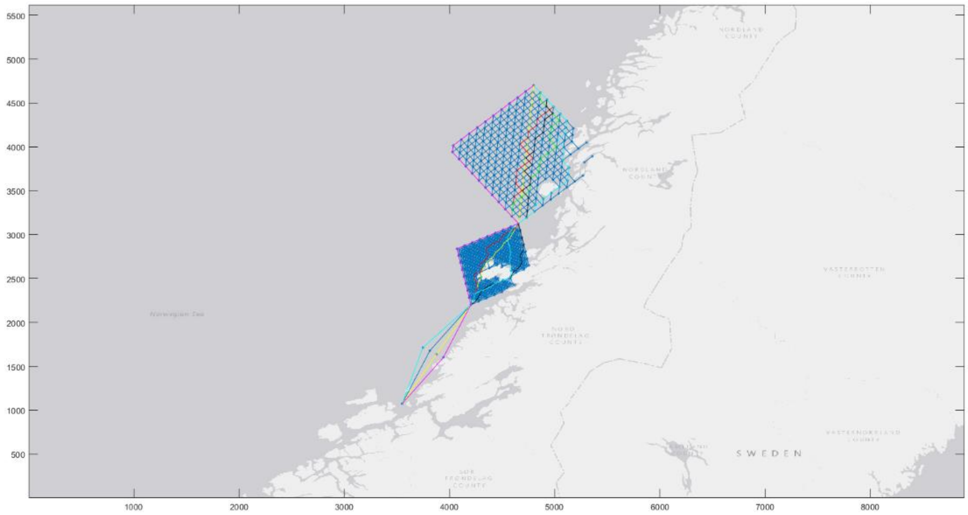

Figure B.24: Vessel velocity and H_s of December

B.13 Multiple linear regression of decision variables

This appendix presents the dataset used for multiple linear regression of (1) sailing distance, (2) vessel velocity, and (3) sailing time. In the multiple linear regression, the dependent variable is the value (sailing distance, vessel velocity, sailing time) of the optimal path when all decision variables are weighted equally. The independent variables are the values of the optimal path with sole contribution from the decision variables. For regression of vessel velocity, the results from MS Excel are also attached.

B.13.1 Sailing distance

Table B.13: Sailing distance [m] by month

Month	$w_i = 0.25$	$w_1 = 1$	$w_2 = 1$	$w_3 = 1$	$w_4=1$
Jan	912065	936959	926528	939750	875781
Feb	929710	948372	956915	963233	865651
Mar	933784	956557	948230	968362	1006506
Apr	908500	931765	933206	943011	980187
May	918073	932707	941597	963233	920041

Jun	928822	944865	946231	956262	906396
Jul	928454	958676	941798	943719	1006506
Aug	917316	917316	914670	952084	976626
Sep	904096	925294	931849	959561	897275
Oct	895542	937089	930150	948899	1006506
Nov	937680	933521	926107	944813	1001118
Dec	912740	943417	922900	933530	994020

The dataset above gave insignificant results with an R^2-value of 0.305. Additionally, none of the independent variables were significant with p-values >> 0.05.

B.13.2 Vessel average velocity

Table B.14: Vessel average velocity [kn] by month

Month	***w_i = 0.25***	***w_1 = 1***	***w_2 = 1***	***w_3 = 1***	***w_4=1***
Jan	4.25	4.29	4.03	4.23	4.22
Feb	4.18	4.06	4.1	4.19	4.17
Mar	5.25	5.38	5.22	5.45	5.66
Apr	4.63	4.75	4.75	4.8	5.09
May	4.13	4	3.91	4.06	4.44
Jun	4.11	4.11	3.99	3.85	4.01
Jul	4.82	4.46	4.71	4.81	5.23
Aug	6.35	5.52	5.37	5.47	7.13
Sep	3.76	3.62	3.59	3.7	3.73
Oct	4.17	3.89	3.99	3.94	4
Nov	4.96	4.85	4.9	4.91	5.3
Dec	5.73	5.42	5.3	5.36	5.37

Table B.15: Vessel average velocity [kn] regression analysis results

Regression Statistics	
Multiple R	0.977
R Square	0.954
Adjusted R Square	0.928
Standard Error	0.206
Observations	12

ANOVA					
	df	*SS*	*MS*	*F*	*Significant F*
Regression	4	6.186	1.547	36.463	8.92E-05
Residual	7	0.297	0.042		
Total	11	6.483			

	Coeff.	*Std. Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	0.120	0.505	0.238	0.819	-1.073	1.313	-1.073	1.313
w1	0.617	0.475	1.298	0.235	-0.507	1.741	-0.507	1.741
w2	0.538	0.703	0.765	0.469	-1.124	2.200	-1.124	2.200
w3	-0.615	0.662	-0.929	0.384	-2.182	0.951	-2.182	0.951
w4	0.447	0.158	2.833	0.025	0.074	0.821	0.074	0.821

B.13.3 Sailing time

Table B.16: Sailing time [h] by month

Month	***w_i = 0.25***	***w_1 = 1***	***w_2 = 1***	***w_3 = 1***	***w_4=1***
Jan	116	118	124	120	112
Feb	120	126	126	124	112
Mar	96	96	98	96	96
Apr	106	106	106	106	104
May	120	126	130	128	112
Jun	122	124	128	134	122
Jul	104	116	108	106	104
Aug	78	90	92	94	74

Sep	130	138	140	140	130
Oct	116	130	126	130	136
Nov	102	104	102	104	102
Dec	86	94	94	94	100

The dataset above gave insignificant results with an R^2-value of 0.941 and an $R^2_{adj} = 0.907$. Moreover, none of the independent variables showed significant correlation to the dependent variable, with all p-values > 0.05. $w_2 = 1$ showed the most correlation with a p-value of 0.132, however not within a significance interval of 95%.

Appendix C

MATLAB

This appendix presents the overview of how the digital service is constructed in MATLAB. In addition, some details of the functions and scripts created in this thesis are presented. Appendix C.1 presents the flowchart and architecture of the MATLAB service, whereas Appendix C.2 presents descriptions of scripts and functions.

C.1 Flowchart of the MATLAB service

The flowchart of the service is presented in Figure C.2. The legend of symbols is presented in Figure C.1.

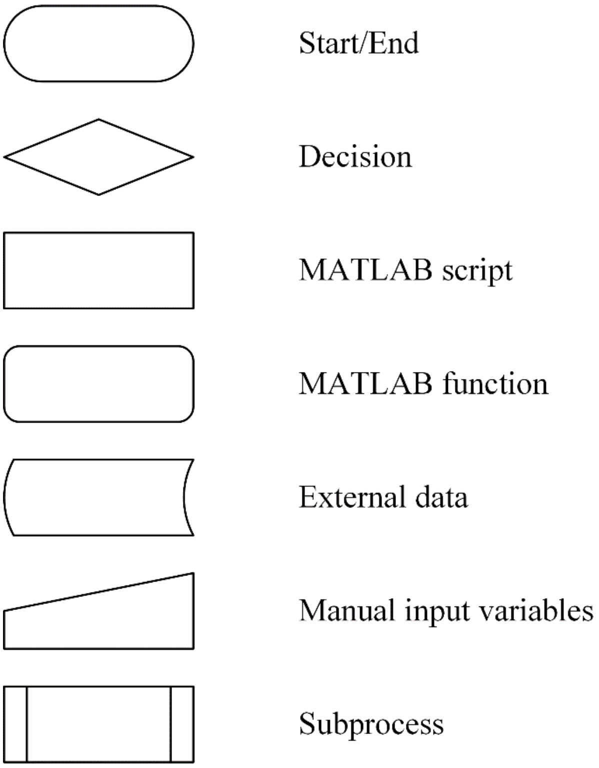

Figure C.1: Legend of symbols in the MATLAB flowchart

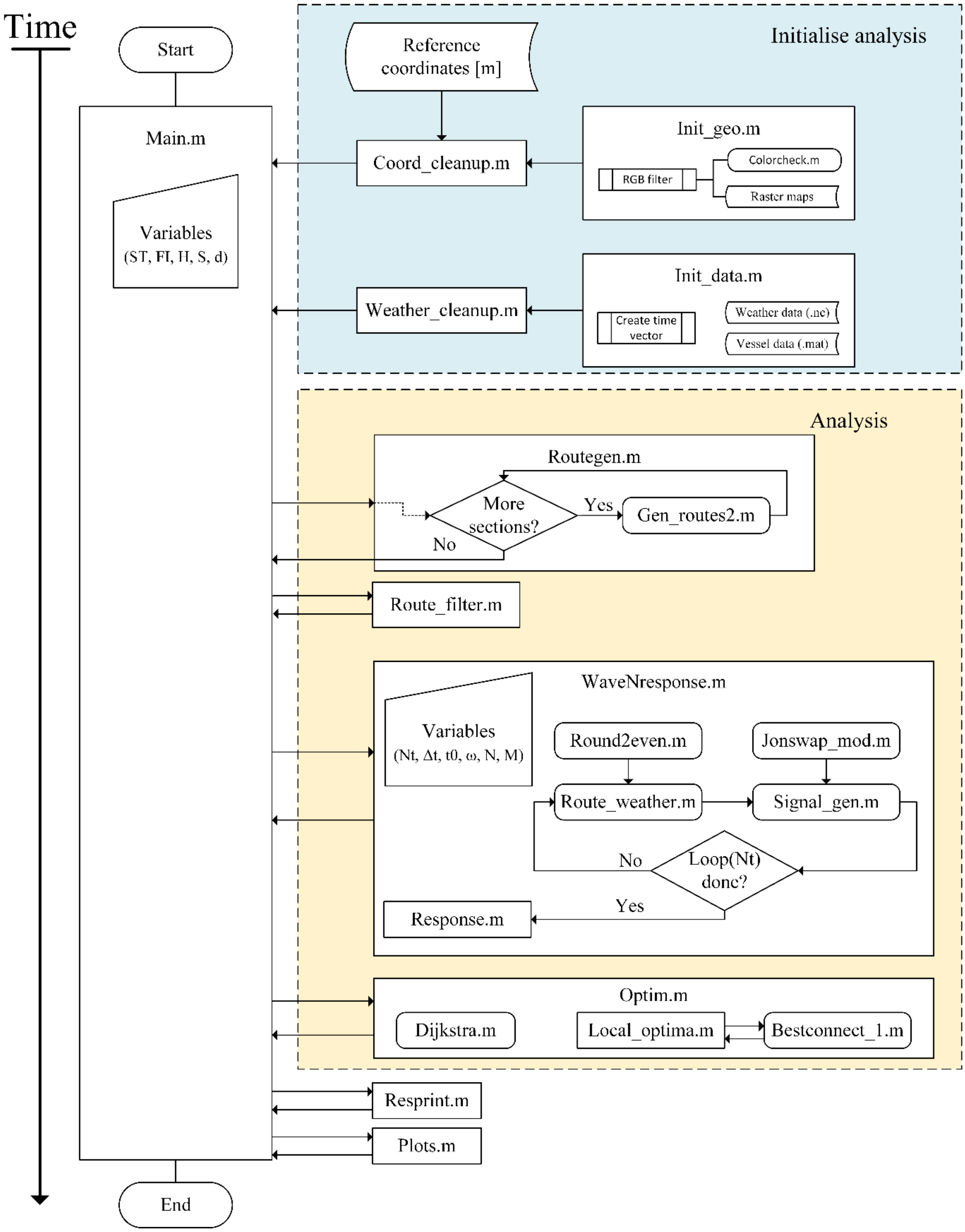

Figure C.2: Flowchart of the MATLAB service

C.2 Descriptions of scripts and functions

- *main.m*: Is the main interface of the service. This is equal to the junction in the proposed digital service platform in Section 2.3.
    - Variables ST, FI, H, S, d are stated manually before running the script. ST and FI can be 1x3 vectors or Nx3 matrices. If the latter, the network is split into smaller sections (sub-networks). H, S, d can be 1x1 or Nx1, independent of each other, i.e. H can be 5x1 with S 1x1. Note that for matrix form of H, S, d, the variables ST and FI also needs to be on matrix form.
- *init_geo.m*: Reads raster files and filters RGB of national borders, province borders and the blurry shorelines. The raster type used in this analysis is PNG. Two raster files are read: One with the cropped data frame, one with the full extent. Both in the same resolution.
    - *colorcheck.m*: Script to trace RGB values in the raster. Note that this script does not need to be ran if you know what RGB values are sea and what is land. Moreover, this is created as a script, rather than a function, as indicated in Figure C.2.
- *coord_cleanup.m*: Script ran after init_geo.m to find the data frame boundaries in the full extent raster. The procedure is to find out what x in the cropped data frame raster has RGB=White. In addition, it assigns pixel distances in meters. This script is dependent of coordinates from ArcGIS, read in ArcGIS and manually entered in the script.
- *init_data.m*: Script to read weather data and vessel data. The weather data is read from a NC-file, whereas the vessel data is read from a MAT-file. The vessel data was obtained from the model that were shared by Prof. Astrid H. Brodtkorb. Furthermore, the script creates a time vector ranging from 0, with a time step equal to the step in the NC-file, in this case: 2 hours.
- *weather_cleanup.m*: Script to clean up weather data. As mentioned, the formatting to NC gave inaccurate weather matrices. Clear NaN values and flip matrices correctly. In addition, make matrices for absolute wind speed and wind direction, and convert wave direction matrix.
- *routegen.m*: Script for the initiation and loop of the route generation function. Loop through N if ST and FI are Nx3 matrices.
    - *Gen_routes2.m*: Function to generate routes. Explained in detail in Section 3.4.

- *route_filter.m*: Script to filter out infeasible routes, i.e. routes that starts or end at land and nodes that have no connecting nodes. Explained in Section 3.4.5.
- *waveNresponse.m*: The main script of the generation of seastates and corresponding responses. Loops through $N_{\Delta t}$, i.e. the number of intervals to generate seastates and all segments. The route weather variables $t_0, \Delta t, N_{\Delta t}$ are stated in this script, as well as JONSWAP variables ω, N, M.
    - *route_weather.m*: Function to align time, segment and weather conditions. In addition, time to complete the segment is found, the vessel velocity based on the weather conditions and the segment length (constant throughout *t*).
        - *round2even.m*: Function to find the nearest even integer.
    - *signal_gen.m*: Function to generate a wave elevation, -velocity and -acceleration timeseries based on the two methods described in Section 3.3.4.
        - *jonswap_mod.m*: Modified JONSWAP spectrum function. The original by Vazirizade (2020). Generates an SDF vector (S), amplitude (A) vector and a time vector.
    - *response.m*: Script to calculates vessel responses based on the timeseries generated by *signal_gen.m*.
- *optim.m*: The optimisation script. Interface where the weight variable W is stated. Runs through the sub-functions:
    - *dijkstra.m*: Dijkstra's algorithm function by Kirk (2020).
    - *local_optima.m*: Script to find local optima as explained in Section 3.6.
        - *bestconnect_1.m*: Function to find the best connecting node for a given node, given the decision variables and their weights. Only checks 1 step ahead of a given node.
        - *bestconnect_2.m*: Extended function to find the best connecting node of a given node. Procedure explained in Section 3.6.
- *plots.m*: Script to plot relevant results.
- *resprint.m*: Script to print relevant results.

Gjermund Prestegårdshus

Route modelling in coastal navigation for digital twins and digital platforms

NTNU
Norwegian University of
Science and Technology